

**geoc**

**version 0.11**

**Jared Erickson**

September 19, 2019



# Contents

<b>geoc Commandline Application</b>	<b>1</b>
Key Features	1
Data Sources	1
Vector	1
Raster	2
Tile	2
Map Layer	3
Examples	3
Buffer a Vector Layer	4
Create Centroids from a Vector Layer	4
Create Interior Points from a Vector Layer	5
Create Convex Hulls around a Vector Layer	5
Create Envelopes around a Vector Layer	6
Create a Vector Layer of Random Points	6
Create a Voronoi Diagram from a Vector Layer	7
Create a Delaunay Diagram from a Vector Layer	7
Select Features that intersect other Features	8
Select Features contained in other Features	8
Select Features within a distance of other Features	9
Create supercircles for each Feature in a Layer	9
Create a Map Cube	10
Generate Geodetic Tiles	10
Generate Web Mercator Tiles in MBTiles	11
Reclassify a Raster	12
Create Polygons from a Raster	12
Commands	13
list	13
version	13
proj wkt	13
proj envelope	14
style create	14
style css2sld	14
style sld2ysld	15
style yslid2sld	15
style uniquevaluesfromtext	15
geometry convert	16
geometry dd2pt	16
geometry geohash encode	17
geometry geohash decode	17
geometry geohash bounds	17

geometry geohash neighbors	18
geometry greatcirclearc	18
geometry offset	18
geometry orthodromicdistance	19
geometry plot	19
geometry pt2dd	19
filter cql2xml	20
map cube	20
map draw	21
vector datastorerelist	21
vector datastoreparams	21
vector display	22
vector list layers	22
vector count	22
vector buffer	23
vector envelope	23
vector envelopes	24
vector centroid	24
vector convexhull	24
vector convexhulls	25
vector interiorpoint	25
vector join attribute	26
vector join spatial	26
vector mincircle	27
vector mincircles	27
vector minrect	28
vector minrects	28
vector octagonalenvelope	28
vector octagonalenvelopes	29
vector draw	29
vector voronoi	30
vector delaunay	30
vector geomw	31
vector geomr	31
vector coordinates	31
vector simplify	32
vector densify	32
vector filter	33
vector delete	33
vector schema	33
vector updatefield	34
vector project	34

vector copy	35
vector randompoints	35
vector create	36
vector add	36
vector grid	36
vector to	37
vector from	37
vector info	38
vector defaultstyle	38
vector uniquevaluesstyle	38
vector gradientstyle	39
vector addfields	39
vector removefields	40
vector addidfield	40
vector addareafield	41
vector addlengthfield	41
vector addxyfields	41
vector splitbyfield	42
vector splitbylayer	42
vector dissolvebyfield	43
vector dissolveintersecting	43
vector pointsalongline	43
vector append	44
vector merge	44
vector clip	45
vector union	45
vector intersection	46
vector intersects	47
vector contains	47
vector distancewithin	47
vector erase	48
vector identity	48
vector update	49
vector symdifference	49
vector validity	50
vector single2multiple	50
vector multiple2single	51
vector compareschemas	51
vector transform	51
vector pointstacker	52
vector raster	52
vector heatmap	53

vector barnessurface	53
vector raster values	54
vector subset	55
vector sort	55
vector page	55
vector translate	56
vector uniquevalues	56
vector shear	57
vector rotate	57
vector scale	58
vector reflect	58
vector smooth	59
vector arc	59
vector arcpolygon	59
vector ellipse	60
vector rectangle	61
vector sinestar	61
vector supercircle	62
vector squircle	62
vector database select	63
vector database sql	63
vector database index create	64
vector database index list	64
vector database index delete	65
vector database remove	65
vector count featuresInfeature	65
vector snap points2lines	66
vector points2lines	66
vector points2polygons	67
vector dump shapefiles	67
vector graticule square	68
vector graticule rectangle	68
vector graticule hexagon	68
vector graticule line	69
vector remove layer	69
raster abs	70
raster info	70
raster display	70
raster draw	71
raster exp	71
raster crop	72
raster crop with geometry	72

raster crop with layer	73
raster extractfootprint	73
raster log	74
raster normalize	74
raster convolve	75
raster get value	75
raster scale	76
raster project	76
raster reclassify	77
raster resample	77
raster invert	78
raster stylize	78
raster add constant	79
raster add	79
raster subtract constant	79
raster subtract	80
raster multiply constant	80
raster multiply	81
raster divide constant	81
raster divide	82
raster mosaic	82
raster envelope	83
raster point	83
raster polygon	83
raster contour	84
raster to	84
raster mapalgebra	85
raster worldfile	85
raster size	86
raster projection	86
raster shadedrelief	86
raster style default	87
raster style shadedrelief	87
raster style channel selection	88
raster style contrast enhancement	88
raster style colormap	88
raster animatedgif	89
tile generate	89
tile delete	90
tile pyramid	90
tile stitch raster	90
tile stitch vector	91

tile vector grid	91
tile get bounds	92
tile list tiles	92
Build	93
Help	93
<b>Indices and tables</b>	<b>94</b>

# geoc Commandline Application

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.

geoc is very much under development (command names may change). Originally it was developed as a complement to [geometry commands](#) and to stress test [GeoScript Groovy](#). The commands have not been optimized for large datasets.

geoc is built on the shoulders of giants: [GeoTools](#) and the [Java Topology Suite](#). geoc just provides a command line application that wraps the herculean effort that the developers of these two libraries have undertaken.

[PDF](#)

Contents:

## Key Features

1. Git style commands. One command (geoc) and many subcommands (which can be more than one word).
2. By default geoc reads and writes vector layers as CSV and raster layers as ASCII grids.
3. But geoc can read and write to any supported [GeoTools](#) DataStore (Shapefiles, PostGIS, H2) or CoverageStore (GeoTIFF, WorldImage, GTOPO).
4. Commands are looked up using Java's Service Provider Interface (SPI) so the framework is extensible.
5. Where appropriate, values are expressions (literals, properties, or CQL with functions)
6. Uses [GeoScript Groovy](#) for extremely terse code.

## Data Sources

By default, vector commands read and write CSV using WKT for geometry fields and raster commands read and write ASCII grids. But, geoc can read and write any supported GeoTools DataStore or CoverageStore by using Connection Strings. GeoTools uses connection maps to connect to DataStore's. geoc connection strings are these connection maps where the key/value pairs are separated by an '=' sign and multiple key/value pairs are separated by a white space. Values can be single quoted.

## Vector

### PostGIS

```
dbtype=postgres database=postgres host=localhost port=5432 user=postgres passwd=postgres
```

### MySQL

```
dbtype=mysql database=layers host=localhost port=5432 user=me passwd=s$cr$t
```

### H2

```
test.db
```

```
dbtype=h2 database=test.db
```

```
dbtype=h2 host=localhost port=5432 schema=public user=me password=s$cr$t
```

```
dbtype=h2 jndiReferenceName=layers schema=public
```

### Shapefile

```
url=data/states.shp
```

```
data/states.shp
```

### Memory

## geoc Commandline Application

memory

### Properties

data/states.properties

directory=data/properties

### GeoPackage

layers.gpkg

database=layers.gpkg dbtype=geopkg user=me passwd=s\$cr\$t

### Geobuf

layer.pbf

file=layers precision=6 dimension=2

### Spatialite

layers.sqlite

dbtype=spatialite database=layers.sqlite

### OGR

DatasourceName=states.shp DriverName='ESRI Shapefile' namespace=shp

### WFS

<http://geoserver.org/wfs?request=getcapabilities>

## Raster

Raster sources are currently all file based.

data/earth.tif

world.png

## Tile

### pyramid

Several tile layers can take a pyramid attribute. You can use one of several well known pyramid names:

- globalmercator
- mercator
- globalmercatorbottomleft
- globalgeodetic
- geodetic

or use a file that contains pyramid metadata in csv, xml, or json format.

### 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

## Examples

```
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=GlobalGeodetic
```

## Map Layer

Map layer strings contain a layertype, layername, layerprojection, and style properties.

### layertype

- 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

### style

A SLD or CSS File

## Examples

```
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
```

```
layertype=raster source=rasters/earth.tif
```

```
layertype=tile file=world.mbtiles
```

```
layertype=tile type=geopackage file=states.gpkg
```

## Examples

List commands:

```
>>> geoc list
```

Count features in a CSV layer:

```
>>> cat states.csv | geoc vector count
```

Buffer feature from a shapefile:

## Examples

```
>>> geoc vector buffer -i earthquakes.shp -o earthquake_buffers.shp
```

Get the envelope of a layer and then calculate the buffer:

```
>>> cat states.csv | geoc vector envelope | geoc vector buffer -d 0.1
```

Crop a raster:

```
>>> geoc raster crop -i raster.tif -b "-120,-40,120,40" -o raster_cropped.tif
```

Create 100 random points in a GeoPackage database, get's metadata of that layer, and then finally converts the layer to CSV::

```
>>> geoc vector randompoints -g "0 0 10 10" -n 100 -o test.gpkg -r points100
```

```
>>> geoc vector info -i test.gpkg -l points100
```

```
>>> geoc vector to -i test.gpkg -f csv
```

## Buffer a Vector Layer

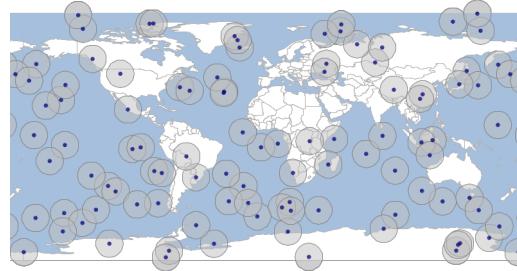
```
geoc vector randompoints -g -180,-90,180,90 -n 100 > points.csv
```

```
cat points.csv | geoc vector buffer -d 10 > polygons.csv
```

```
cat points.csv | geoc vector defaultstyle --color navy -o 0.75 > points.sld
```

```
cat polygons.csv | geoc vector defaultstyle --color silver -o 0.5 > polygons.sld
```

```
geoc map draw -f vector_buffer.png -l "layertype=layer file=naturalearth.gpkg layername=ocean"
-l "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \
-l "layertype=layer file=polygons.csv style=polygons.sld" \
-l "layertype=layer file=points.csv style=points.sld"
```



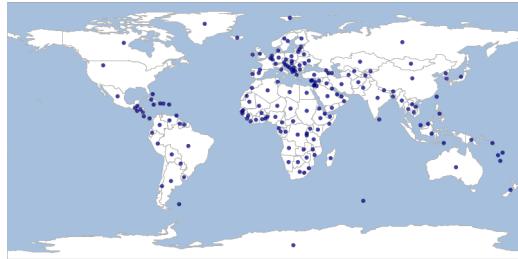
## Create Centroids from a Vector Layer

```
geoc vector centroid -i naturalearth.gpkg -l countries -o countries_centroids.shp
```

```
geoc vector defaultstyle --color navy -o 0.75 -i countries_centroids.shp > countries_centroids.sld
```

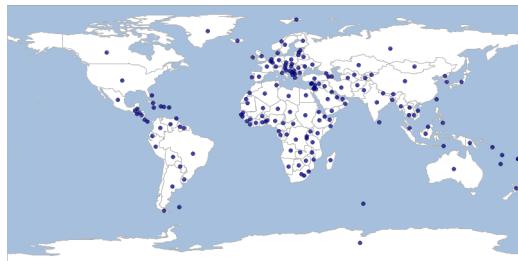
```
geoc map draw -f vector_centroid.png -l "layertype=layer file=naturalearth.gpkg layername=ocean"
-l "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \
-l "layertype=layer file=countries_centroids.shp style=countries_centroids.sld"
```

## Examples



### Create Interior Points from a Vector Layer

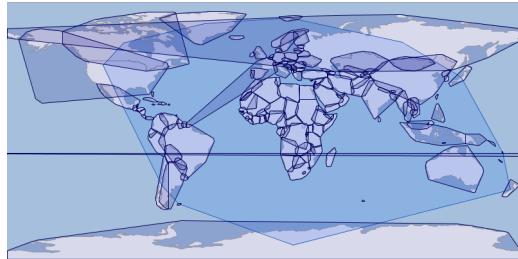
```
geoc vector interiorpoint -i naturalearth.gpkg -l countries -o countries_interiorpoints.shp  
geoc vector defaultstyle --color navy -o 0.75 -i countries_interiorpoints.shp > countries_in  
geoc map draw -f vector_interiorpoint.png -l "layertype=layer file=naturalearth.gpkg layern  
-l "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \\\  
-l "layertype=layer file=countries_interiorpoints.shp style=countries_interiorpoints.sld"
```



### Create Convex Hulls around a Vector Layer

```
#!/bin/bash  
geoc vector centroid -i naturalearth.gpkg -l countries | geoc vector convexhull -o countries  
geoc vector convexhulls -i naturalearth.gpkg -l countries -o countries_convexhulls.shp  
  
geoc vector defaultstyle --color "#0066FF" -o 0.15 -g line > countries_convexhull.sld  
geoc vector defaultstyle --color navy -o 0.15 -g polygon > countries_convexhulls.sld  
  
geoc map draw -f vector_convexhull.png -b "-180,-90,180,90" \\  
-l "layertype=layer file=naturalearth.gpkg layername=ocean style=ocean.sld" \\  
-l "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \\  
-l "layertype=layer file=countries_convexhull.shp" \\  
-l "layertype=layer file=countries_convexhulls.shp"
```

## Examples



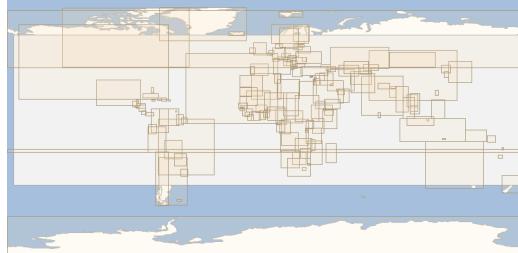
### Create Envelopes around a Vector Layer

```
#!/bin/bash

rm envelope.db.*

geoc vector envelope -i naturalearth.gpkg -l places -o envelope.db -r envelope
geoc vector envelopes -i naturalearth.gpkg -l countries -o envelope.db -r envelopes

geoc map draw -f vector_envelope.png -b "-180,-90,180,90" \
  -l "layertype=layer file=naturalearth.gpkg layername=ocean style=ocean.sld" \
  -l "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \
  -l "layertype=layer dbtype=h2 database=envelope.db layername=envelope style='stroke=#006699'" \
  -l "layertype=layer dbtype=h2 database=envelope.db layername=envelopes style='stroke=#ab9966'"
```



### Create a Vector Layer of Random Points

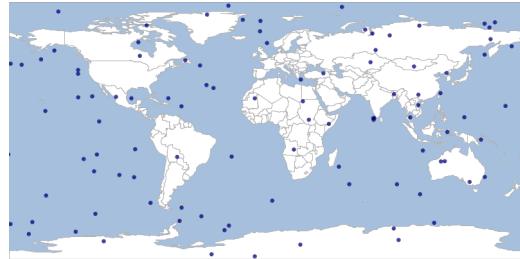
```
#!/bin/bash

geoc vector randompoints -g -180,-90,180,90 -n 100 > points.csv

cat points.csv | geoc vector defaultstyle --color navy -o 0.75 > points.sld

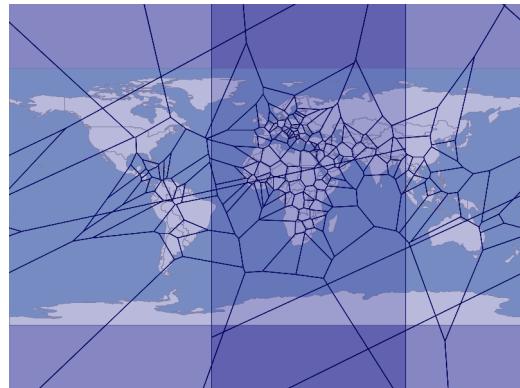
geoc map draw -f vector_random.png -l "layertype=layer file=naturalearth.gpkg layername=ocean" \
  -l "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \
  -l "layertype=layer file=points.csv style=points.sld"
```

## Examples



### Create a Voronoi Diagram from a Vector Layer

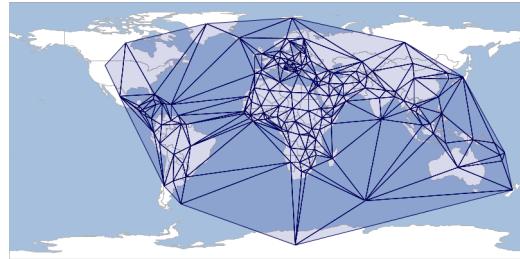
```
geoc vector centroid -i naturalearth.gpkg -l countries | geoc vector voronoi -o countries_voronoi.shp > countries_voronoi.sld  
  
geoc vector defaultstyle --color navy -o 0.15 -i countries_voronoi.shp > countries_voronoi.sld  
  
geoc map draw -f vector_voronoi.png -b "-180,-90,180,90" \  
-l "layertype=layer file=naturalearth.gpkg layername=ocean style=ocean.sld" \  
-l "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \  
-l "layertype=layer file=countries_voronoi.shp layername=countries_voronoi style=countries_voronoi.sld"
```



### Create a Delaunay Diagram from a Vector Layer

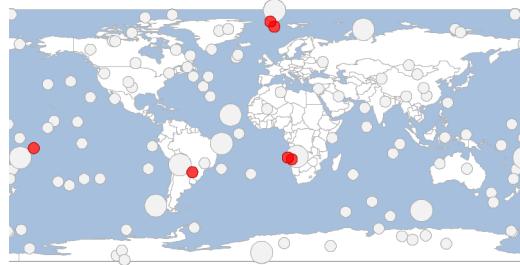
```
#!/bin/bash  
geoc vector centroid -i naturalearth.gpkg -l countries | geoc vector delaunay -o countries_delaunay.pbf > countries_delaunay.sld  
  
geoc vector defaultstyle --color navy -o 0.15 -g polygon > countries_delaunay.sld  
  
geoc map draw -f vector_delaunay.png -b "-180,-90,180,90" \  
-l "layertype=layer file=naturalearth.gpkg layername=ocean style=ocean.sld" \  
-l "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \  
-l "layertype=layer file=countries_delaunay.pbf layername=countries_delaunay style=countries_delaunay.sld"
```

## Examples



### Select Features that intersect other Features

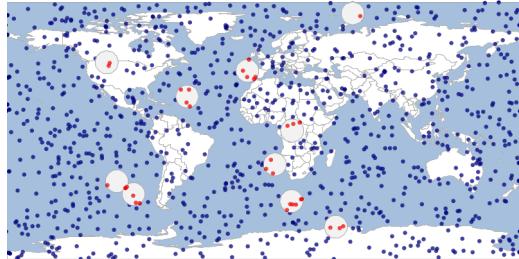
```
#!/bin/bash
geoc vector randompoints -g -180,-90,180,90 -n 100 | geoc vector buffer -d 4 -o polys1.shp
geoc vector randompoints -g -180,-90,180,90 -n 10 | geoc vector buffer -d 8 -o polys2.shp
geoc vector intersects -i polys1.shp -k polys2.shp > intersectingPolys.csv
cat intersectingPolys.csv | geoc vector defaultstyle --color red -o 0.75 > intersectingPolys.sld
geoc map draw -f vector_intersects.png -l "layertype=layer file=naturalearth.gpkg layername=countries" \
-l "layertype=layer file=polys1.shp" \
-l "layertype=layer file=polys2.shp" \
-l "layertype=layer file=intersectingPolys.csv style=intersectingPolys.sld"
```



### Select Features contained in other Features

```
#!/bin/bash
geoc vector randompoints -g -180,-90,180,90 -n 1000 -o points.shp
geoc vector randompoints -g -180,-90,180,90 -n 10 | geoc vector buffer -d 8 -o polys.shp
geoc vector contains -i points.shp -k polys.shp > pointsInPolys.csv
cat pointsInPolys.csv | geoc vector defaultstyle --color red -o 0.75 > pointsInPolys.sld
geoc map draw -f vector_contains.png -l "layertype=layer file=naturalearth.gpkg layername=countries" \
-l "layertype=layer file=points.shp" \
-l "layertype=layer file=polys.shp" \
-l "layertype=layer file=pointsInPolys.csv style=pointsInPolys.sld"
```

## Examples



### Select Features within a distance of other Features

```
#!/bin/bash

geoc vector randompoints -g -180,-90,180,90 -n 1000 -o points.shp

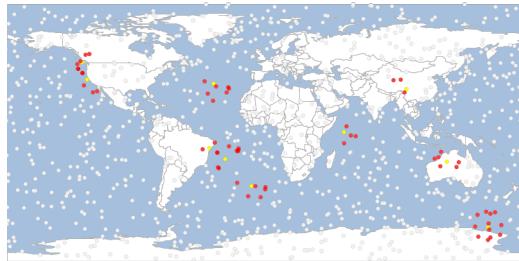
geoc vector randompoints -g -180,-90,180,90 -n 10 -o otherPoints.shp

geoc vector distancewithin -i points.shp -d 12 -k otherPoints.shp > pointsNearOthers.csv

geoc vector defaultstyle -g point --color yellow > otherPoints.sld

cat pointsNearOthers.csv | geoc vector defaultstyle --color red -o 0.75 > pointsNearOthers.sld

geoc map draw -f vector_distancewithin.png -l "layertype=layer file=naturalearth.gpkg layername=countries" \
-l "layertype=layer file=otherPoints.shp" \
-l "layertype=layer file=pointsNearOthers.sld"
```



### Create supercircles for each Feature in a Layer

```
geoc vector supercircle -i naturalearth.gpkg -l countries -e 0.45 -p 20 > supercircles.csv

geoc vector supercircle -i naturalearth.gpkg -l countries -g "centroid(the_geom)" -w 5 -h 5

geoc vector envelopes -i naturalearth.gpkg -l countries > envelopes.csv

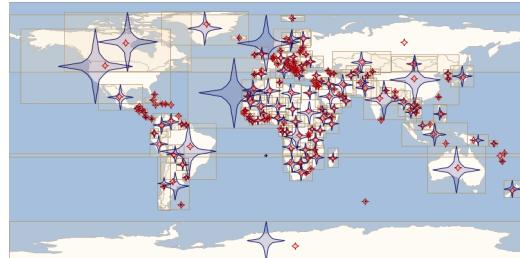
geoc vector defaultstyle --color navy -o 0.15 -g polygon > supercircles.sld

geoc vector defaultstyle --color red -o 0.15 -g polygon > supercircles2.sld

geoc vector defaultstyle --color wheat -o 0.15 -g polygon > envelopes.sld
```

## Examples

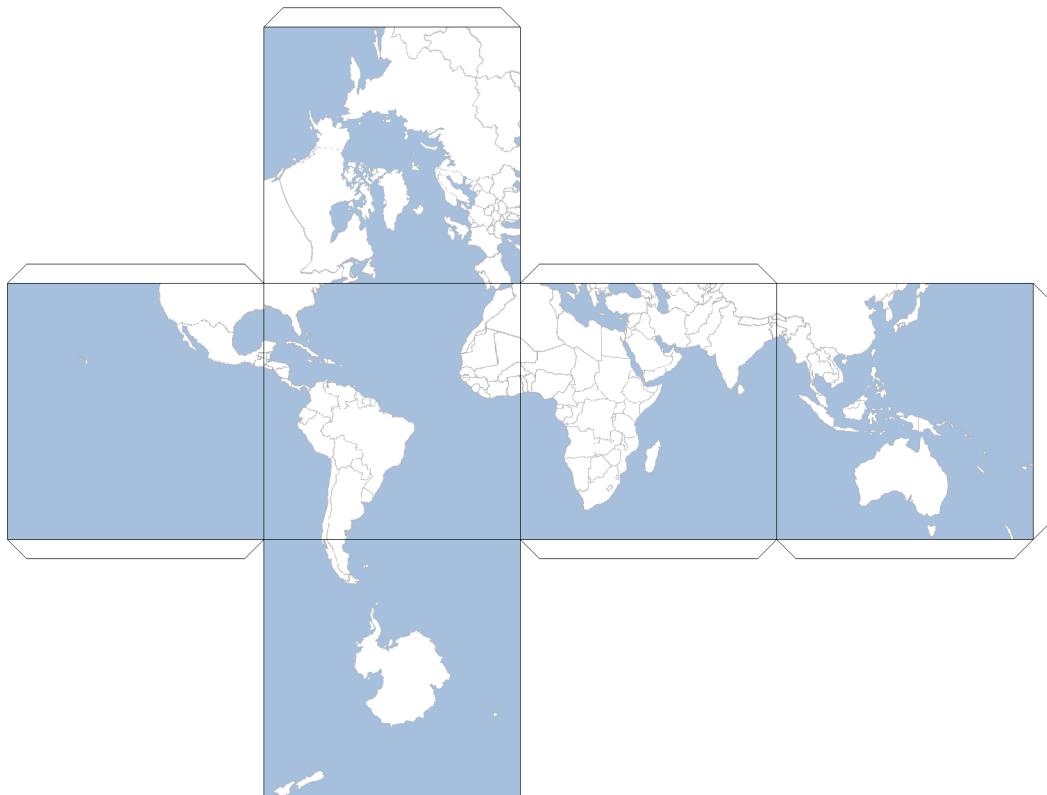
```
geoc map draw -f vector_supercircles.png -b "-180,-90,180,90" \
-l "layertype=layer file=natural-earth.gpkg layername=ocean style=ocean.sld" \
-l "layertype=layer file=natural-earth.gpkg layername=countries style=countries.sld" \
-l "layertype=layer file=envelopes.csv style=envelopes.sld" \
-l "layertype=layer file=supercircles.csv style=supercircles.sld" \
-l "layertype=layer file=supercircles2.csv style=supercircles2.sld"
```



## Create a Map Cube

```
geoc map cube -f map_cube.png -t -o -i 'Natural Earth' \
-l "layertype=layer file=natural-earth.gpkg layername=ocean style=ocean.sld" \
-l "layertype=layer file=natural-earth.gpkg layername=countries style=countries.sld" \
```

**Natural Earth**



## Generate Geodetic Tiles

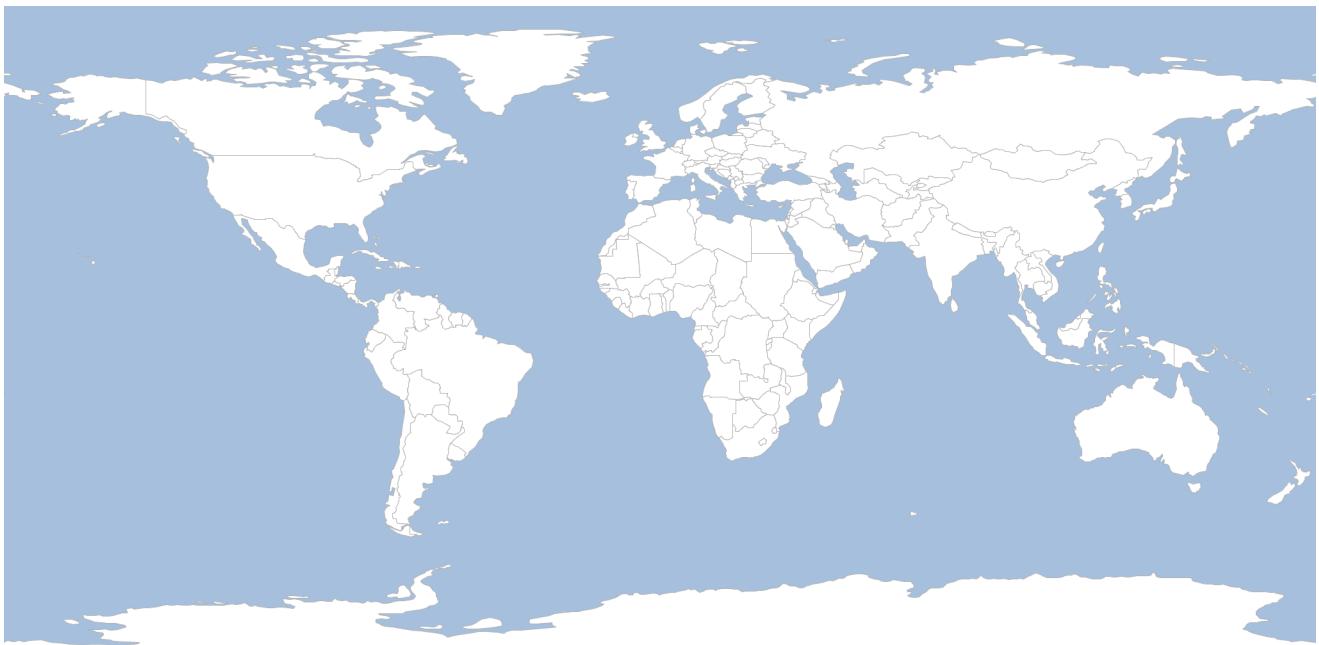
## Examples

```
#!/bin/bash
rm tiles.gpkg

geoc tile generate -l "type=geopackage file=tiles.gpkg name=world_geodetic pyramid=geodetic"
  -m "layertype=layer file=naturalearth.gpkg layername=ocean style=ocean.sld" \
  -m "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \
  -s 0 \
  -e 3 \
  -v

geoc tile pyramid -l "type=geopackage file=tiles.gpkg name=world_geodetic" -o json

geoc tile stitch raster -l "type=geopackage file=tiles.gpkg name=world_geodetic" \
  -o world_geodetic_2.png -z 2
```



## Generate Web Mercator Tiles in MBTiles

```
#!/bin/bash
rm countries.mbtiles

geoc tile generate -l countries.mbtiles \
  -m "layertype=layer file=naturalearth.gpkg layername=ocean style=ocean.sld" \
  -m "layertype=layer file=naturalearth.gpkg layername=countries style=countries.sld" \
  -s 0 \
  -e 4

geoc tile pyramid -l countries.mbtiles -o text

geoc tile stitch raster -l countries.mbtiles -o countries_1.png -z 1
```

## Examples



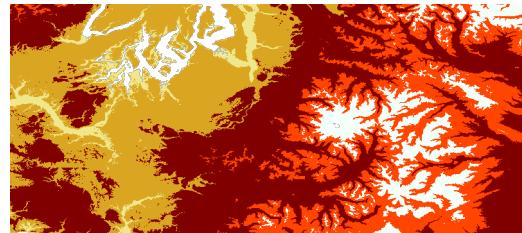
### Reclassify a Raster

```
#!/bin/sh

geoc raster reclassify -i pc.tif -o pc_reclass.tif \
    -r 0-0=1 -r 0-50=2 -r 50-200=3 \
    -r 200-1000=5 -r 1000-1500=4 -r 1500-4000=6

geoc raster style colormap \
    -v 1=#FFFACD -v 2=#F0E68C -v 3=#DAA520 \
    -v 4=#FF4500 -v 5=#800000 -v 6=#F5FFFA > pc_reclass.sld

geoc map draw -f pc_reclass.png -l "layertype=raster source=pc_reclass.tif style=pc_reclass.sld"
```



### Create Polygons from a Raster

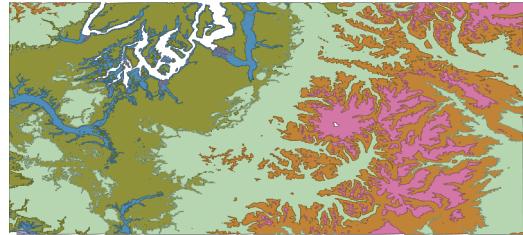
```
#!/bin/sh

geoc raster reclassify -i pc.tif -o pc_reclass.tif \
    -r 0-0=1 -r 0-50=2 -r 50-200=3 \
    -r 200-1000=5 -r 1000-1500=4 -r 1500-4000=6

geoc raster polygon -e -i pc_reclass.tif -o pc_reclass_poly.shp

geoc vector uniquevaluesstyle -i pc_reclass_poly.shp -f value \
    -c BoldLandUse > pc_reclass_poly.sld

geoc map draw -f pc_reclass_poly.png \
    -l "layertype=layer file=pc_reclass_poly.shp style=pc_reclass_poly.sld"
```



## Commands

### *list*

**Name:**

geoc list

**Description:**

List all geocommands

**Arguments:**

- -d --description: Include the description
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc list
```

### *version*

**Name:**

geoc version

**Description:**

Get the version

**Arguments:**

- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc version
```

### *proj wkt*

**Name:**

geoc proj wkt

**Description:**

Get the WKT of a Projection

**Arguments:**

## Commands

- -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

### Example:

```
geoc proj wkt -e EPSG:4326
```

## proj envelope

### Name:

geoc proj envelope

### Description:

Get a Projection's envelope

### Arguments:

- -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

### Example:

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

## style create

### Name:

geoc style create

### Description:

Create a simple style

### Arguments:

- -s --style-options: A style options
- -t --type: The output type (sld or ysl)
- -o --output: The output file
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc style create -s fill=white -s stroke=black -s stroke-width=0.1 -t ysl
```

## style css2sld

### Name:

## Commands

geoc style css2sld

### Description:

Convert CSS to SLD

### Arguments:

- -i --input: The input file or url
- -o --output: The output file
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc style css2sld -i states.css -o states.sld
```

## **style sld2ysld**

### Name:

geoc style sld2ysld

### Description:

Convert SLD to YSLD

### Arguments:

- -i --input: The input file or url
- -o --output: The output file
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc style sld2ysld -i countries.sld -o countries.yml
```

## **style ysl2sld**

### Name:

geoc style ysl2sld

### Description:

Convert YSLD to SLD

### Arguments:

- -i --input: The input file or url
- -o --output: The output file
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc style ysl2sld -i countries.yml -o countries.sld
```

## **style uniquevaluesfromtext**

### Name:

geoc style uniquevaluesfromtext

## Commands

### Description:

Create a Style from reading values in the unique values format

### Arguments:

- -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 yslsld)
- -o --output: The output file
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc style uniquevaluesfromtext -f unit -g Polygon -i units.txt -o units.sld
```

## geometry convert

### Name:

geoc geometry convert

### Description:

Convert a geometry from one format to another

### Arguments:

- -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

### Example:

```
geoc geometry convert -i "POINT (-122 48)" -f geojson
```

## geometry dd2pt

### Name:

geoc geometry dd2pt

### Description:

Convert a decimal degrees formatted string into a Point

### Arguments:

- -d --decimaldegrees: The decimal degrees
- -t --type: The output type (xy, wkt, json)
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc geometry dd2pt -d "122d 19m 59.0016s W, 47d 36m 34.9992s N"
```

## geometry geohash encode

**Name:**

geoc geometry geohash encode

**Description:**

Encode a Geometry as a GeoHash

**Arguments:**

- -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

**Example:**

```
geoc geometry geohash encode -i "POINT (45 78)"
```

## geometry geohash decode

**Name:**

geoc geometry geohash decode

**Description:**

Decode a GeoHash to a Geometry.

**Arguments:**

- -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

**Example:**

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

## geometry geohash bounds

**Name:**

geoc geometry geohash bounds

**Description:**

Calculate the geohashes for the given bounds

**Arguments:**

- -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

**Example:**

```
geoc geometry geohash bounds -b "120, 30, 120.0001, 30.0001"
```

## **geometry geohash neighbors**

**Name:**

geoc geometry geohash neighbors

**Description:**

Get a geohash's neighbors

**Arguments:**

- -i --input: The input geometry
- -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

**Example:**

```
geoc geometry geohash neighbors -i uf8vk6wjr
```

## **geometry greatcirclearc**

**Name:**

geoc geometry greatcirclearc

**Description:**

Create a great circle arc.

**Arguments:**

- -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

**Example:**

```
geoc geometry greatcirclearc -p "POINT (-122 48)" -t "POINT (-77 39)"
```

## **geometry offset**

**Name:**

geoc geometry offset

**Description:**

Create a Geometry offset from the input Geometry

**Arguments:**

- -i --input: The input geometry
- -d --offset: The offset distance

## Commands

- -s --quadrant-segments: The number of quadrant segments (defaults to 8)
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc geometry offset -i "LINESTRING (10 0, 10 10)" -d 5 -s 8
```

## geometry orthodromicdistance

### Name:

geoc geometry orthodromicdistance

### Description:

Calculate the orthodromic distance between two points.

### Arguments:

- -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

### Example:

```
geoc geometry orthodromicdistance -e wgs84 -p "-86.67 36.12" -t "-118.40 33.94"
```

## geometry plot

### Name:

geoc geometry plot

### Description:

Draw a geometry to a plot

### Arguments:

- -i --input: The input geometry
- -f --file: The output file
- -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

### Example:

```
geoc geometry plot -d -i "POLYGON ((80 80, 80 120, 120 120, 120 80, 80 80))"
```

## geometry pt2dd

## Commands

### Name:

geoc geometry pt2dd

### Description:

Format a Point in Decimal Degrees

### Arguments:

- -p --point: The Point
- -t --type: The output type (dms, dms\_char, ddm, ddm\_char)
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc geometry pt2dd -p "POINT (-122.5256194 47.212022222)" -t dms
```

## filter cql2xml

### Name:

geoc filter cql2xml

### Description:

Convert a CQL statement to an OCG XML Filter

### Arguments:

- -c --cql: The CQL statement
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc filter cql2xml -c STATE_ABBR=WA
```

## map cube

### Name:

geoc map cube

### Description:

Create a map cube

### Arguments:

- -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

### Example:

## Commands

```
geoc map cube -m map.groovy -f mapcube.png
```

### **map draw**

#### **Name:**

geoc map draw

#### **Description:**

Draw a Map

#### **Arguments:**

- -l --layer: The map layer
- -i --layer-file: The input layer file
- -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

#### **Example:**

```
geoc map draw -l "layertype=layer dbtype=geopkg database=data/countries.gpkg layername=country
```

### **vector datastорelist**

#### **Name:**

geoc vector datastорelist

#### **Description:**

List all available DataStores

#### **Arguments:**

- --help : Print the help message
- --web-help : Open help in a browser

#### **Example:**

```
geoc vector datastорelist
```

### **vector datastoreparams**

#### **Name:**

geoc vector datastoreparams

#### **Description:**

List all parameters for the given DataStore

#### **Arguments:**

- -n --name: The DataStore name

## Commands

- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector datastoreparams -n PostGIS
```

## **vector display**

### Name:

geoc vector display

### Description:

Display a Layer in a simple viewer

### Arguments:

- -w --width: The width
- -h --height: The height
- -s --sld-file: The sld file
- -b --bounds: The bounds
- -m --layer: The map layer
- -g --background-color: The background color
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector display -i points.shp -w 400 -h 400
```

## **vector list layers**

### Name:

geoc vector list layers

### Description:

List Layers in a DataStore

### Arguments:

- -i --input-workspace: The input workspace
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector list layers -i "dbtype=postgis database=naturalearth host=localhost port=5432 user=naturalearth password=naturalearth"
```

## **vector count**

### Name:

geoc vector count

### Description:

## Commands

Count the features, geometries, or points in a Layer

### Arguments:

- -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

### Example:

```
geoc vector count -i states.shp
```

## vector buffer

### Name:

geoc vector buffer

### Description:

Buffer the features of the input Layer and save them to the output Layer

### Arguments:

- -d --distance: The buffer distance
- -q --quadrantsegments: The number of quadrant segments
- -s --singlesided: 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

### Example:

```
geoc vector randompoints -n 10 -g "1,1,10,10" | geoc vector buffer -d 10
```

## vector envelope

### Name:

geoc vector envelope

### Description:

Get the bounding envelope of all the features of the input Layer and save it to the output Layer

### Arguments:

## Commands

- -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

### Example:

```
geoc vector envelope -i states.shp -o states_envelope.shp
```

## vector envelopes

### Name:

geoc vector envelopes

### Description:

Calculate the envelope of each feature in the input Layer and save them to the output Layer

### Arguments:

- -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

### Example:

```
geoc vector envelopes -i states.shp -o state_envelopes.shp
```

## vector centroid

### Name:

geoc vector centroid

### Description:

Get the centroid of each feature in the input Layer and save them to the output Layer

### Arguments:

- -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

### Example:

```
geoc vector centroid -i states.shp -o state_centroids.shp
```

## vector convexhull

## Commands

### Name:

geoc vector convexhull

### Description:

Calculate the convex hull of the input Layer and save it to the output Layer

### Arguments:

- -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

### Example:

```
geoc vector convexhull -i states.shp -o state_convexhull.shp
```

## vector convexhulls

### Name:

geoc vector convexhulls

### Description:

Calculate the convex hull of each feature in the input Layer and save them to the output Layer

### Arguments:

- -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

### Example:

```
geoc vector convexhulls -i states.shp -o state_convexhulls.shp
```

## vector interiorpoint

### Name:

geoc vector interiorpoint

### Description:

Get the interior point of each feature in the input Layer and save them to the output Layer

### Arguments:

- -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

**Example:**

```
geoc vector interiorpoint -i states.shp -o state_interiorpoints.shp
```

## **vector join attribute**

**Name:**

geoc vector join attribute

**Description:**

Perform a attribute join between a Layers and a table.

**Arguments:**

- -s --table-source: The table source
- -t --table-name: The table name
- -y --layer-field: The input layer field name
- -j --table-field: The other layer field name
- -n --field: The join field names to include in the output layer
- -m --only-include-matching: The flag to whether only include matching rows
- -p --options: The options (for csv separator and quote, for dbf encoding)
- -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

**Example:**

```
geoc vector join attribute -i polygons.shp -s table.csv -o polygons_table.shp -y id -j key -
```

## **vector join spatial**

**Name:**

geoc vector join spatial

**Description:**

Spatially join two layers to create the output Layer.

**Arguments:**

- -f --field: A Field name
- -t --spatial-type: The spatial type (intersects, contains). Defaults to intersects.
- -m --multiple-type: The multiple type (first, closest, largest). Defaults to first.

## Commands

- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector join spatial -i points.shp -k polygons.shp -o points_joined.shp -f name -f descr
```

## **vector mincircle**

### Name:

geoc vector mincircle

### Description:

Calculate the minimum bounding circle of the input Layer and save them to the output Layer

### Arguments:

- -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

### Example:

```
geoc vector mincircle -i states.shp -o state_mincircle.shp
```

## **vector mincircles**

### Name:

geoc vector mincircles

### Description:

Calculate the minimum bounding circles of each feature in the input Layer and save them to the output Layer

### Arguments:

- -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

### Example:

```
geoc vector mincircles -i states.shp -o state_mincircles.shp
```

## **vector minrect**

**Name:**

geoc vector minrect

**Description:**

Calculate the minimum rectangle of the input Layer and save it to the output Layer

**Arguments:**

- -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

**Example:**

```
geoc vector minrect -i states.shp -o state_minrect.shp
```

## **vector minrects**

**Name:**

geoc vector minrects

**Description:**

Calculate the minimum rectangle of each feature in the input Layer and save them to the output Layer

**Arguments:**

- -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

**Example:**

```
geoc vector minrects -i states.shp -o state_minrects.shp
```

## **vector octagonalenvelope**

**Name:**

geoc vector octagonalenvelope

**Description:**

Calculate the octagonal envelope of the input Layer and save it to the output Layer

**Arguments:**

## Commands

- -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

### Example:

```
geoc vector octagonalenvelope -i states.shp -o state_octagonalenvelope.shp
```

## **vector octagonalenvelopes**

### Name:

geoc vector octagonalenvelopes

### Description:

Calculate the octagonal envelope of each feature in the input Layer and save them to the output Layer

### Arguments:

- -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

### Example:

```
geoc vector octagonalenvelopes -i states.shp -o state_octagonalenvelopes.shp
```

## **vector draw**

### Name:

geoc vector draw

### Description:

Draw a Layer to an Image, PDF, or SVG Document

### Arguments:

- -f --file: The output file
- -t --type: The type of document
- -w --width: The width
- -h --height: The height
- -s --sld-file: The sld file
- -b --bounds: The bounds
- -m --layer: The map layer
- -g --background-color: The background color
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector draw -i states.shp -f image.png -w 600 -h 400 && open image.png
```

## **vector voronoi**

**Name:**

geoc vector voronoi

**Description:**

Calculate a voronoi diagram of the input Layer and save it to the output Layer

**Arguments:**

- -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

**Example:**

```
geoc vector voronoi -i states.shp -o states_voronoi.shp
```

## **vector delaunay**

**Name:**

geoc vector delaunay

**Description:**

Calculate a delaunay triangle diagram of the input Layer and save it to the output Layer

**Arguments:**

## Commands

- -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

### Example:

```
geoc vector delaunay -i states.shp -o states_delaunay.shp
```

## **vector geomw**

### Name:

geoc vector geomw

### Description:

Convert the input layer to a text stream of WKT geometries that can be read by the geom commands

### Arguments:

- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector geomw -i states.shp | geom combine | geom draw && open image.png
```

## **vector geomr**

### Name:

geoc vector geomr

### Description:

Convert a text stream of WKT geometries to a Layer

### Arguments:

- -t --text: The text
- -o --output-workspace: The output workspace
- -r --output-layer: The output layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
echo "POINT (1 1)" | geom buffer -d 100 | geom random -n 100 | geom dump | geoc vector geomr
```

## **vector coordinates**

### Name:

geoc vector coordinates

### Description:

## Commands

Extract coordinates from the input Layer and save them to the output Layer.

### Arguments:

- -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

### Example:

```
geoc vector coordinates -i states.shp -o states_coordinates.shp
```

## **vector simplify**

### Name:

geoc vector simplify

### Description:

Simplify the features of the input Layer and save them to the output Layer

### Arguments:

- -a --algorithm: The simplify algorithm (DouglasPeucker - dp or TopologyPreserving - tp)
- -d --distance: The distance tolerance
- -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

### Example:

```
geoc vector simplify -i states.shp -o states_simplified.shp -a DouglasPeucker -d 100
```

## **vector densify**

### Name:

geoc vector densify

### Description:

Densify the features of the input Layer and save them to the output Layer

### Arguments:

## Commands

- -d --distance: The distance tolerance
- -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

### Example:

```
geoc vector densify -i states.shp -o states_densified -d 10
```

## vector filter

### Name:

geoc vector filter

### Description:

Filter features from the input Layer and save them to the output Layer

### Arguments:

- -f --filter: The CQL Filter
- -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

### Example:

```
geoc vector filter -i states.shp -f "STATE_POP > 1000000" -o states_largepop.shp
```

## vector delete

### Name:

geoc vector delete

### Description:

Delete features from a Layer in place

### Arguments:

- -f --filter: The CQL Filter
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector delete -i states.shp -f "STATE_NAME = 'Washington'" -o states_no_wash.shp
```

## vector schema

## Commands

### Name:

geoc vector schema

### Description:

Get the Layer's schema

### Arguments:

- -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

### Example:

```
geoc vector schema -i states.shp -p
```

## **vector updatefield**

### Name:

geoc vector updatefield

### Description:

Update the values of a Layer's Field

### Arguments:

- -d --field: The Field name
- -v --value: The value
- -f --filter: The CQL Filter
- -s --script: Whether the value is a script or not
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector updatefield -i states_xy.shp -f INCLUDE -s -v "return f.geom.centroid.x" -d x
```

## **vector project**

### Name:

geoc vector project

### Description:

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

### Arguments:

- -s --source-projection: The source projection
- -t --target-projection: The target projection
- -o --output-workspace: The output workspace
- -r --output-layer: The output layer

## Commands

- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector project -i states.shp -o states_2927.shp -t "EPSG:2927"
```

## vector copy

### Name:

geoc vector copy

### Description:

Copy the input Layer to the output Layer

### Arguments:

- -f --filter: The CQL Filter
- -s --sort: The sort Field
- -t --start: The start index
- -m --max: The max number of Features
- -d --field: The sub 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

### Example:

```
geoc vector copy -i states.shp -o states.gpkg -r states
```

## vector randompoints

### Name:

geoc vector randompoints

### Description:

Create a new Layer with randomly placed points

### Arguments:

- -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
- -u --id-fieldname: The id field name

## Commands

- -o --output-workspace: The output workspace
- -r --output-layer: The output layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector randompoints -n 10 -g "0,0,10,10" -o "dbtype=h2 database=h2.db" -r points
```

## vector create

### Name:

geoc vector create

### Description:

Create a new Layer

### Arguments:

- -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
- --web-help : Open help in a browser

### Example:

```
geoc vector create -o mypoints.shp -f "the_geom=POINT EPSG:4326" -f "id=int" -f "name=string"
```

## vector add

### Name:

geoc vector add

### Description:

Add a Feature to an existing Layer

### Arguments:

- -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

### Example:

```
geoc vector add -i mypoints.shp -v "id=1" -v "the_geom=POINT(1 1)" -v "name=House"
```

## vector grid

### Name:

geoc vector grid

### Description:

Create a vector grid

### Arguments:

## Commands

- -y --rows: The number of rows
- -x --columns: The number of columns
- -w --cell-width: The cell width
- -h --cell-height: The cell height
- -t --type: The grid cell type
- -g --geometry: The geometry
- -p --projection: The projection
- -o --output-workspace: The output workspace
- -r --output-layer: The output layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector grid -g "0,0,10,10" -x 4 -y 4
```

## **vector to**

### Name:

geoc vector to

### Description:

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

### Arguments:

- -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

### Example:

```
geoc vector to -i alki_points.properties -f csv > alki_points.csv
```

## **vector from**

### Name:

geoc vector from

### Description:

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

### Arguments:

- -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

**Example:**

```
curl -s http://earthquake.usgs.gov/earthquakes/feed/v1.0/summary/all_hour.geojson | geoc vec
```

## **vector info**

**Name:**

geoc vector info

**Description:**

Get basic information about the Layer

**Arguments:**

- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector info -i zones.shp
```

## **vector defaultstyle**

**Name:**

geoc vector defaultstyle

**Description:**

Get the default style for the Layer

**Arguments:**

- -g --geometry-type: The geometry type
- -c --color: The base color
- -o --opacity: The opacity (defaults to 1.0)
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector defaultstyle zones.shp
```

## **vector uniquevaluesstyle**

**Name:**

geoc vector uniquevaluesstyle

**Description:**

Create an SLD document where each unique value in the Layer is a rule.

**Arguments:**

- -f --field: The field name
- -c --colors: The color brewer palette name or a list of colors (space delimited)
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector uniquevaluesstyle -i states.shp -f STATE_ABBR -c "Greens"
```

## vector gradientstyle

**Name:**

geoc vector gradientstyle

**Description:**

Create a gradient SLD for the Layer.

**Arguments:**

- -f --field: The field name
- -n --number: The number of categories
- -c --colors: The color brewer palette name or a list of colors (space delimited)
- -m --method: The classification method (Quantile or EqualInterval)
- -e --else-mode: The else mode (ignore, min, max)
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector gradientstyle -i states.shp -f SAMP_POP -n 6 -c greens
```

## vector addfields

**Name:**

geoc vector addfields

**Description:**

Add one or more Fields to the input Layer to create the output Layer

**Arguments:**

- -f --field: A Field in the format 'name=type'
- -o --output-workspace: The output workspace

## Commands

- -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

### Example:

```
geoc vector addfields -i states.shp -o states_xy.shp -f x=double -f y=double
```

## vector removefields

### Name:

geoc vector removefields

### Description:

Remove one or more Fields from the input Layer to create the output Layer

### Arguments:

- -f --field: A Field name
- -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

### Example:

```
geoc vector removefields -i states.shp -o states_temp.shp -f description -f name -f boundedBy
```

## vector addidfield

### Name:

geoc vector addidfield

### Description:

Add an ID Field

### Arguments:

- -f --id-fieldname: The name for the ID Field
- -s --start: The number of start at
- -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

### Example:

```
geoc vector addidfield -i states.shp -o states_id.shp -f ID -s 1
```

## **vector addareafield**

**Name:**

geoc vector addareafield

**Description:**

Add an area Field

**Arguments:**

- -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

**Example:**

```
geoc vector addareafield -i states.shp -o states_area.shp -f area
```

## **vector addlengthfield**

**Name:**

geoc vector addlengthfield

**Description:**

Add an length/perimeter Field

**Arguments:**

- -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

**Example:**

```
geoc vector addlengthfield -i roads.shp -o roads_len.shp -f LENGTH
```

## **vector addxyfields**

**Name:**

geoc vector addxyfields

**Description:**

Add a XY Fields

**Arguments:**

- -x --x-fieldname: The name for the X Field
- -y --y-fieldname: The name for the Y Field

## Commands

- -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

### Example:

```
geoc vector addxyfields -i points.shp -o points_xy.shp -x X_COL -y Y_COL
```

## **vector splitbyfield**

### Name:

geoc vector splitbyfield

### Description:

Split a Layer into separate Layers based on values from a Field

### Arguments:

- -f --field: The field name
- -o --output-workspace: The output workspace
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector splitbyfield -i states.shp -o state_regions -f SUB_REGION
```

## **vector splitbylayer**

### Name:

geoc vector splitbylayer

### Description:

Split a Layer into separate Layers based on the Feature from another Layer

### Arguments:

- -s --split-workspace: The input workspace
- -p --split-layer: The input layer
- -f --field: The field name
- -o --output-workspace: The output workspace
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector splitbylayer -i states.shp -s states_grid.shp -o statesgrid -f col_row
```

## ***vector dissolvebyfield***

**Name:**

geoc vector dissolvebyfield

**Description:**

Dissolve the Features of a Layer by a Field.

**Arguments:**

- -f --field: The field name
- -d --id-field: The id field name
- -c --count-field: The count field name
- -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

**Example:**

```
geoc vector dissolvebyfield -i states.shp -o states_subregions.shp -f SUB_REGION
```

## ***vector dissolveintersecting***

**Name:**

geoc vector dissolveintersecting

**Description:**

Dissolve the intersecting Features of a Layer.

**Arguments:**

- -d --id-field: The id field name
- -c --count-field: The count field name
- -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

**Example:**

```
geoc vector dissolveintersecting -i polys -o polys_dissolved -d ID -c COUNT
```

## ***vector pointsalongline***

**Name:**

geoc vector pointsalongline

**Description:**

Create points along lines

**Arguments:**

- -d --distance: The distance between points
- -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

**Example:**

```
geoc vector pointsalongline -i rivers.shp -o sample_locations.shp -d 10.0
```

## **vector append**

**Name:**

geoc vector append

**Description:**

Append the Features from an other Layer to the input Layer

**Arguments:**

- -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

**Example:**

```
geoc vector append -i points.shp -k locations.shp
```

## **vector merge**

**Name:**

geoc vector merge

**Description:**

Merge two Layers together to create a new Layer

**Arguments:**

- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

**Example:**

```
geoc vector merge -i states_SUB_REGION_Pacific.shp -k states_SUB_REGION_Mtn.shp -o states_we
```

## **vector clip**

**Name:**

geoc vector clip

**Description:**

Clip the input Layer by the other Layer to produce the output Layer.

**Arguments:**

- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

**Example:**

```
geoc vector clip -i states.shp -k area_of_interest.shp -o states_clipped.shp
```

## **vector union**

**Name:**

geoc vector union

**Description:**

Union one Layer with another Layer

**Arguments:**

- -p --postfix-all: Whether to postfix all field names (true) or not (false). If true, all Fields from the this current Schema will have '1' at the end of their name while the other Schema's Fields will have '2'.
- -d --include-duplicates: Whether or not to include duplicate fields names. Defaults to false. If a duplicate is found a '2' will be added.
- -m --maxfieldname-length: The maximum new Field name length (mostly to support shapefiles where Field names can't be longer than 10 characters)
- -f --first-postfix: The postfix for fields from the first Layer
- -s --second-postfix: The postfix for fields from the second Layer
- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

**Example:**

```
geoc vector union -i states.shp -k clip_layer.shp -o states_union.shp
```

## ***vector intersection***

**Name:**

geoc vector intersection

**Description:**

Calculate the intersection between two Layers

**Arguments:**

- -p --postfix-all: Whether to postfix all field names (true) or not (false). If true, all Fields from the this current Schema will have '1' at the end of their name while the other Schema's Fields will have '2'.
- -d --include-duplicates: Whether or not to include duplicate fields names. Defaults to false. If a duplicate is found a '2' will be added.
- -m --maxfieldname-length: The maximum new Field name length (mostly to support shapefiles where Field names can't be longer than 10 characters)
- -f --first-postfix: The postfix for fields from the first Layer
- -s --second-postfix: The postfix for fields from the second Layer
- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

**Example:**

```
geoc vector intersection -i states.shp -k clip_layer.shp -o states_intersection.shp
```

## **vector intersects**

**Name:**

geoc vector intersects

**Description:**

Only include Features from the Input Layer that Intersect with Features from the Other Layer in the Output Layer.

**Arguments:**

- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

**Example:**

```
geoc vector intersects -i points.shp -k polygons.shp -o pointsIntersectingPolygons.shp
```

## **vector contains**

**Name:**

geoc vector contains

**Description:**

Only include Features from the Input Layer that are contained by Features from the Other Layer in the Output Layer.

**Arguments:**

- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

**Example:**

```
geoc vector contains -i points.shp -k polygons.shp -o pointsInPolygons.shp
```

## **vector distancewithin**

**Name:**

geoc vector distancewithin

**Description:**

## Commands

Only include Features from the Input Layer that are within a given distance of Features from the Other Layer in the Output Layer.

### Arguments:

- -d --distance: The distance
- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector distancewithin -i points.shp -k polygons.shp -o pointsInPolygons.shp -d 4.56
```

## vector erase

### Name:

geoc vector erase

### Description:

Erase features from one Layer based on another Layer

### Arguments:

- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector erase -i states.shp -k clip_layer.shp -o states_erase.shp
```

## vector identity

### Name:

geoc vector identity

### Description:

Calculate the identity between one Layer and another Layer.

### Arguments:

- -p --postfix-all: Whether to postfix all field names (true) or not (false). If true, all Fields from the this current Schema will have '1' at the end of their name while the other Schema's Fields will have '2'.

## Commands

- -d --include-duplicates: Whether or not to include duplicate fields names. Defaults to false. If a duplicate is found a '2' will be added.
- -m --maxfieldname-length: The maximum new Field name length (mostly to support shapefiles where Field names can't be longer than 10 characters)
- -f --first-postfix: The postfix for fields from the first Layer
- -s --second-postfix: The postfix for fields from the second Layer
- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector identity -i states.shp -k clip_layer.shp -o states_identity.shp
```

## ***vector update***

### Name:

geoc vector update

### Description:

Update one Layer with another Layer

### Arguments:

- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector update -i states.shp -k clip_layer.shp -o states_update.shp
```

## ***vector symdifference***

### Name:

geoc vector symdifference

### Description:

Calculate the symmetric difference between two Layers.

### Arguments:

## Commands

- -p --postfix-all: Whether to postfix all field names (true) or not (false). If true, all Fields from the this current Schema will have '1' at the end of their name while the other Schema's Fields will have '2'.
- -d --include-duplicates: Whether or not to include duplicate fields names. Defaults to false. If a duplicate is found a '2' will be added.
- -m --maxfieldname-length: The maximum new Field name length (mostly to support shapefiles where Field names can't be longer than 10 characters)
- -f --first-postfix: The postfix for fields from the first Layer
- -s --second-postfix: The postfix for fields from the second Layer
- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector symdifference -i states.shp -k clip_layer.shp -o states_symdifference.shp
```

## vector validity

### Name:

geoc vector validity

### Description:

Check whether geometry in a Layer is valid or not

### Arguments:

- -f --field: A Field to include when reporting an invalid Geometry
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
cat invalid.csv | geoc vector validity
```

## vector single2multiple

### Name:

geoc vector single2multiple

### Description:

Combine all of the geometries in the input Layer into one multipart geometry in the output Layer

### Arguments:

- -o --output-workspace: The output workspace
- -r --output-layer: The output layer

## Commands

- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
cat points.csv | geoc vector single2multiple -o multi.properties
```

## **vector multiple2single**

### Name:

geoc vector multiple2single

### Description:

Convert multipart geometries in the input Layer into single geometries in the output Layer.

### Arguments:

- -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

### Example:

```
geoc vector multiple2single -i multi.properties
```

## **vector compareschemas**

### Name:

geoc vector compareschemas

### Description:

Compare Schemas from two Layers

### Arguments:

- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector compareschemas -i states.shp -k states_xy.shp
```

## **vector transform**

### Name:

## Commands

geoc vector transform

### Description:

Transform the values of the input Layer using Expression and Functions

### Arguments:

- -d --definition: A transform definition 'field=expression'
- -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

### Example:

```
geoc vector transform -i point.properties -i points_transformed.shp -d "the_geom=buffer(the_
```

## vector pointstacker

### Name:

geoc vector pointstacker

### Description:

Group nearby points together

### Arguments:

- -c --cell-size: The cell size in pixels which aggregates points
- -b --bounds: The bounds
- -w --width: The output width
- -h --height: The output height
- -s --source-projection: The source 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
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector pointstacker -i earthquakes.shp -o stacked_quakes.shp -c 5 -w 800 -h 600
```

## vector raster

### Name:

geoc vector raster

### Description:

Convert a vector Layer to a Raster

### Arguments:

## Commands

- -d --field: The field name with value
- -s --grid-size: The grid size
- -b --bounds: The bounds
- -n --raster-name: The raster name
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- -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

### Example:

```
geoc vector raster -i states.shp -o states.tif -d population -s 600,600
```

## vector heatmap

### Name:

geoc vector heatmap

### Description:

Create a heatmap of the input layer

### Arguments:

- -r --radius-pixels: The radius of the density kernel in pixels
- -a --weight-field: The name of the weight field
- -p --pixels-per-cell: The resolution of the computed grid
- -b --bounds: The output bounds
- -w --width: The output width
- -h --height: The output height
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- -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

### Example:

```
geoc vector heatmap -i earthquakes.properties -o heatmap.tif -r 50 -w 800 -h 800
```

## vector barnessurface

### Name:

geoc vector barnessurface

### Description:

Create a barnes surface of the features from the input layer

### Arguments:

- -v --value-field: The name of the value field
- -d --data-limit: The maximum number of features to process
- -s --scale: The interpolation length
- -c --convergence: The refinement factor
- -p --passes: The number of passes
- -m --min-observations: The minimum number of observations to be a grid cell
- -x --max-observation-distance: The max distance for an observation to be a grid cell
- -n --no-data: The no data value
- -e --pixels-per-cell: The resolution of the computed grid
- -q --query-buffer: The query buffer
- -b --bounds: The output bounds
- -w --width: The output width
- -h --height: The output height
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- -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

**Example:**

```
geoc vector barnessurface -i point_grid.properties -o barnes.tif -s 1 -v value -m 1 -w 800 -
```

## **vector raster values**

**Name:**

geoc vector raster values

**Description:**

Get value from a Raster for each Feature's geometry

**Arguments:**

- -n --field-name: The new value field name (defaults to value)
- -t --field-type: The new value field type (defaults to double)
- -b --band: The band to get values from (defaults to 0)
- -s --input-raster: The input raster
- -e --input-raster-name: The input raster name
- -p --input-projection: The input 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
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector raster values -i points.shp -s raster.tif -o points_values.shp
```

## vector subset

**Name:**

geoc vector subset

**Description:**

Extract a subset of Features from the input Layer

**Arguments:**

- -f --filter: The CQL Filter
- -s --sort: The sort field
- -m --max: The maximum number of Features to include
- -t --start: The index of the Feature to start at
- -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

**Example:**

```
geoc vector subset -i earthquakes.shp -s "date ASC" -s "title ASC" -t 5 -m 10 -o ten_earthquakes.shp
```

## vector sort

**Name:**

geoc vector sort

**Description:**

Sort the Features in the input Layer.

**Arguments:**

- -s --sort: The sort 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

**Example:**

```
cat states.csv | geoc vector sort -s "STATE_NAME ASC"
```

## vector page

**Name:**

## Commands

geoc vector page

### Description:

Page through Feature in the input Layer

### Arguments:

- -m --max: The maximum number of Features to include
- -t --start: The index of the Feature to start at
- -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

### Example:

```
cat states.csv | geoc vector page -t 0 -m 2
```

## **vector translate**

### Name:

geoc vector translate

### Description:

Translate or move Feature in a Layer

### Arguments:

- -x --x-distance: The x distance
- -y --y-distance: The y distance
- -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

### Example:

```
geoc vector translate -i points.properties -i points_transolatate.shp -x 5 -y 10
```

## **vector uniquevalues**

### Name:

geoc vector uniquevalues

### Description:

List the unique values in a Layer's Field

### Arguments:

- -f --field: The field name
- -i --input-workspace: The input workspace

## Commands

- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
$ cat states.csv | geoc vector uniquevalues -f SUB_REGION
```

## vector shear

### Name:

geoc vector shear

### Description:

Shear Features in a Layer

### Arguments:

- -x --x-distance: The x distance
- -y --y-distance: The y distance
- -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

### Example:

```
geoc vector shear -i points.properties -o points_sheared.shp -x 5 -y 10
```

## vector rotate

### Name:

geoc vector rotate

### Description:

Rotate Features in a Layer

### Arguments:

- -t --theta: The angle of rotation in radians
- -s --sine: The sine of the angle of rotation in radians
- -c --cosine: The cosine of the angle of rotation in radians
- -x --x-coord: The x coordinate of the rotation point
- -y --y-coord: The y coordinate of the rotation point
- -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

**Example:**

```
geoc vector rotate -i polys.properties -o rotated_polys.shp -t 0.785 -x "getX(centroid(geom))
```

## vector scale

**Name:**

geoc vector scale

**Description:**

Scale Feature in a Layer

**Arguments:**

- -x --x-distance: The x distance
- -y --y-distance: The y distance
- -c --x-coord: The x coordinate
- -d --y-coord: The y coordinate
- -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

**Example:**

```
geoc vector scale -i polys.properties -o scaled_polys.shp -x 5 -y 5 -c "getX(centroid(geom))
```

## vector reflect

**Name:**

geoc vector reflect

**Description:**

Reflect Features in a Layer

**Arguments:**

- -x --x1-distance: The x1 distance
- -y --y1-distance: The y1 distance
- -c --x2-distance: The x2 distance
- -d --y2-distance: The y2 distance
- -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

**Example:**

```
geoc vector reflect -i polys.properties -o reflected_polys.shp -x 5 -y 5
```

## **vector smooth**

**Name:**

geoc vector smooth

**Description:**

Smooth the features of the input Layer and save them to the output Layer

**Arguments:**

- -f --fit: The amount of smoothing (between 0 - more and 1 - less)
- -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

**Example:**

```
geoc vector smooth -i jagged.shp -o smooth.shp -f 0.25
```

## **vector arc**

**Name:**

geoc vector arc

**Description:**

Create a arc shape around each feature of the input Layer

**Arguments:**

- -s --start-angle: The start angle
- -e --end-angle: The end angle
- -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

**Example:**

```
geoc vector arc -i states.shp -o states_arc.shp -p 100 -s 45 -e 90
```

## **vector arcpolygon**

**Name:**

geoc vector arcpolygon

**Description:**

Create a arc polygon shape around each feature of the input Layer

**Arguments:**

- -s --start-angle: The start angle
- -e --end-angle: The end angle
- -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

**Example:**

```
geoc vector arcpolygon -i states.shp -o states_arcpoly.shp -p 100 -s 45 -e 90
```

## **vector ellipse**

**Name:**

geoc vector ellipse

**Description:**

Create a ellipse shape around each feature of the input Layer

**Arguments:**

- -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

**Example:**

```
geoc vector ellipse -i states.shp -o states_ellipse.shp -g "centroid(geom)" -w 10000 -h 2000
```

## **vector rectangle**

**Name:**

geoc vector rectangle

**Description:**

Create a rectangle shape around each feature of the input Layer

**Arguments:**

- -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

**Example:**

```
geoc vector rectangle -i states.shp -o states_rects.shp -g "centroid(geom)" -w 10000 -h 2000
```

## **vector sinestar**

**Name:**

geoc vector sinestar

**Description:**

Create a sinestar shape around each feature of the input Layer

**Arguments:**

## Commands

- -n --number-of-arms: The number of arms
- -e --arm-length-ratio: The arm length ratio
- -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

### Example:

```
geoc vector sinestar -i states.shp -o states_sinestar.shp -n 10 -e 0.75 -p 100
```

## vector supercircle

### Name:

geoc vector supercircle

### Description:

Create a super circle shape around each feature of the input Layer

### Arguments:

- -e --power: The power
- -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

### Example:

```
geoc vector supercircle -i polys.properties -o supercircles.shp -p 40 -e 0.4
```

## vector squircle

## Commands

### Name:

geoc vector squircle

### Description:

Create a squircle shape around each feature of the input Layer

### Arguments:

- -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

### Example:

```
geoc vector squircle -i polys.properties -o squircles.shp -p 40
```

## vector database select

### Name:

geoc vector database select

### Description:

Get a Layer from a Database using a SELECT statement

### Arguments:

- -w --database-workspace: The input workspace
- -l --layer-name: The input layer
- -s --sql: The input layer
- -g --geometry-field: The geometry field (name|type|projection)
- -p --primary-key-field: The primary key field names
- -o --output-workspace: The output workspace
- -r --output-layer: The output layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector database select -w h2.db -l centroids -o centroids.properties -s "SELECT ST_CENT
```

## vector database sql

### Name:

## Commands

```
geoc vector database sql
```

### Description:

Execute SQL commands against a Database Workspace

### Arguments:

- -w --database-workspace: The input workspace
- -s --sql: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector database sql -w h2.db -s "insert into "points" ("id", "the_geom", "name") values
```

## **vector database index create**

### Name:

```
geoc vector database index create
```

### Description:

Create a database index

### Arguments:

- -w --database-workspace: The input workspace
- -l --layer-name: The input workspace
- -i --index-name: The input workspace
- -f --field: The input workspace
- -u --unique: The input workspace
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector database index create -w points.db -i geom_index -l points50 -f the_geom
```

## **vector database index list**

### Name:

```
geoc vector database index list
```

### Description:

List database indices

### Arguments:

- -w --database-workspace: The input workspace
- -l --layer-name: The input workspace
- -p --pretty-print: Whether to pretty print the output
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc vector database index list -w points.db -l points50 -p
```

## **vector database index delete**

**Name:**

geoc vector database index delete

**Description:**

Delete a database index

**Arguments:**

- -w --database-workspace: The input workspace
- -l --layer-name: The input workspace
- -i --index-name: The input workspace
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector database index delete -w points.db -l points50 -i geom_index
```

## **vector database remove**

**Name:**

geoc vector database remove

**Description:**

Remove a Layer from a Database

**Arguments:**

- -w --database-workspace: The input workspace
- -l --layer-name: The input workspace
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector database remove -w points.db -l points50
```

## **vector count featuresInfeature**

**Name:**

geoc vector count featuresInfeature

**Description:**

Count the number of features in a feature

**Arguments:**

## Commands

- -f --count-fieldname: The name for the count Field
- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector grid -g "0,0,10,10" -x 10 -y 10 -o grid.shp
```

```
geoc vector randompoints -g "0 0 10 10" -n 100 -o points.shp
```

```
geoc vector count featuresinfeature -i grid.shp -k points.shp -o grid_count.shp
```

## vector snap points2lines

### Name:

```
geoc vector snap points2lines
```

### Description:

Snap points to their nearest line

### Arguments:

- -d --search-distance: The distance to search for the closest line
- -s --snapped-fieldname: The name for the snapped Field
- -k --other-workspace: The other workspace
- -y --other-layer: The other layer
- -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

### Example:

```
geoc vector snap points2lines -i points.shp -k lines.shp -o snapped.shp -d 2
```

## vector points2lines

### Name:

```
geoc vector points2lines
```

### Description:

Convert points to lines

### Arguments:

- -s --sort-field: The Field to sort the field

## Commands

- -g --group-field: The Field used create separate Lines
- -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

### Example:

```
geoc vector points2lines -i points.shp -o lines.shp -s id -g group
```

## **vector points2polygons**

### Name:

geoc vector points2polygons

### Description:

Convert points to polygons

### Arguments:

- -s --sort-field: The Field to sort the field
- -g --group-field: The Field used create separate Lines
- -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

### Example:

```
geoc vector points2polygons -i points.shp -o polygons.shp -s id -g group
```

## **vector dump shapefiles**

### Name:

geoc vector dump shapefiles

### Description:

Create shapefiles from the input Layer

### Arguments:

- -o --output-directory: The output directory
- -s --max-shp-size: The maximum shp size
- -d --max-dbf-size: The maximum dbf size
- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
cat pointsAndPolygons.csv | geoc vector dump shapfiles -o shapefiles
```

## **vector graticule square**

**Name:**

geoc vector graticule square

**Description:**

Create square graticules

**Arguments:**

- -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

**Example:**

```
geoc vector graticule square -l 10 -g -180,-90,180,90
```

## **vector graticule rectangle**

**Name:**

geoc vector graticule rectangle

**Description:**

Create rectangle graticules

**Arguments:**

- -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

**Example:**

```
geoc vector graticule rectangle -w 10 -h 15 -g -180,-90,180,90
```

## **vector graticule hexagon**

**Name:**

geoc vector graticule hexagon

**Description:**

## Commands

Create hexagon graticules

### Arguments:

- -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

### Example:

```
geoc vector graticule hexagon -l 10 -g -180,-90,180,90
```

## **vector graticule line**

### Name:

geoc vector graticule line

### Description:

Create line graticules

### Arguments:

- -g --geometry: The geometry
- -s --spacing: The spacing (defaults to -1)
- -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

### Example:

```
geoc vector graticule line -g -180,-90,180,90 -l "vertical,2,10" -l "vertical,1,2" -l "horiz
```

## **vector remove layer**

### Name:

geoc vector remove layer

### Description:

Remove a Layer from a Workspace

### Arguments:

- -i --input-workspace: The input workspace
- -l --input-layer: The input layer
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc vector remove layer -i layers.gpkg -l points
```

## raster abs

**Name:**

geoc raster abs

**Description:**

Calculate the absolute value for each cell.

**Arguments:**

- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

**Example:**

```
geoc raster abs -i raster.tif -o raster_abs.tif
```

## raster info

**Name:**

geoc raster info

**Description:**

Get information about a Raster

**Arguments:**

- -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

**Example:**

```
geoc raster info -i raster.tif
```

## raster display

**Name:**

geoc raster display

**Description:**

Display a Raster in a simple viewer

**Arguments:**

- -w --width: The width

## Commands

- -h --height: The height
- -s --sld-file: The sld file
- -b --bounds: The bounds
- -m --layer: The map layer
- -g --background-color: The background color
- -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

### Example:

```
geoc raster display -i raster.tif -w 400 -h 400
```

## raster draw

### Name:

geoc raster draw

### Description:

Draw a Raster to an image

### Arguments:

- -f --file: The output file
- -t --type: The type of document
- -w --width: The width
- -h --height: The height
- -s --sld-file: The sld file
- -b --bounds: The bounds
- -m --layer: The map 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

### Example:

```
geoc raster draw -i raster.tif -f map.png
```

## raster exp

### Name:

geoc raster exp

### Description:

Calculate the exponent for each cell.

### Arguments:

- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

**Example:**

```
geoc raster exp -i raster.tif -o raster_exp.tif
```

## raster crop

**Name:**

geoc raster crop

**Description:**

Crop a Raster

**Arguments:**

- -b --bound: The Bounds
- -x --pixel: Whether the Bounds is pixel or geographic
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

**Example:**

```
geoc raster crop -i raster.tif -b "-120,-40,120,40" -o raster_cropped.tif
```

## raster crop with geometry

**Name:**

geoc raster crop with geometry

**Description:**

Crop a Raster with a Geometry

**Arguments:**

- -g --geometry: The Geometry
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

**Example:**

```
geoc raster crop with geometry -i alki.gif -o alki_cropped.tif -g "geom buffer -g "POINT (1
```

## raster crop with layer

**Name:**

geoc raster crop with layer

**Description:**

Crop a Raster using the geometry from a Layer

**Arguments:**

- -w --input-workspace: The input workspace
- -y --input-layer: The input layer
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

**Example:**

```
geoc raster crop with layer -i alki.tif -o alki_cropped.tif -w poly.shp
```

## raster extractfootprint

**Name:**

geoc raster extractfootprint

**Description:**

Extract the footprint of the Raster as a Vector Layer

**Arguments:**

- -e --exclusion-range: A comma delimited range of values to exclude from the search.
- -t --threshold-area: A number used to exclude small Polygons. The default is 5.
- -f --compute-simplified-footprint: Whether to compute a simplified footprint or not. The default is false.
- -s --simplifier-factor: A number used to simplify the geometry. The default is 2.
- -c --remove-collinear: Whether to remove collinear coordinates. The default is true.

## Commands

- -v --force-valid: Whether to force creation of valid polygons. The default is true.
- -y --loading-type: The image loading type (Deferred or Immediate). Immediate is the default.
- -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

### Example:

```
geoc raster extractfootprint -i raster.tif -o footprint.shp
```

## raster log

### Name:

geoc raster log

### Description:

Calculate the log for each cell.

### Arguments:

- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster log -i raster.tif -o raster_log.tif
```

## raster normalize

### Name:

geoc raster normalize

### Description:

Normalize the values of a Raster

### Arguments:

## Commands

- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster normalize -i raster.tif -o raster_normalized.tif
```

## raster convolve

### Name:

geoc raster convolve

### Description:

Convolve the values of a Raster

### Arguments:

- -w --width: The kernel width
- -h --height: The kernel height
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster convolve -i raster.tif -o raster_convolved.tif -w 2 -h 3
```

## raster get value

### Name:

geoc raster get value

### Description:

Get the cell value from a Raster

### Arguments:

- -x --x-coordinate: The x coordinate
- -y --y-coordinate: The y coordinate
- -t --type: The type can be point or pixel
- -b --band: The band to get a value from
- -i --input-raster: The input raster
- -l --input-raster-name: The input raster name

## Commands

- -p --input-projection: The input projection
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc raster get value -i alki.tif -x 5 -y 5 -t pixel
```

## raster scale

### Name:

geoc raster scale

### Description:

Scale a Raster

### Arguments:

- -x --x-scale: The scale factor along the x axis
- -y --y-scale: The scale factor along the y axis
- -t --x-translate: The x translation
- -r --y-translate: The y translation
- -n --interpolation: The interpolation method (bicubic, bicubic2, bilinear, nearest)
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster scale -i raster.tif -x 2 -y 3 -o raster_scaled.tif
```

## raster project

### Name:

geoc raster project

### Description:

Project a Raster

### Arguments:

## Commands

- -t --target-projection: The target projection
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster project -i raster.tif -o raster_4326.tif -t EPSG:4326
```

## raster reclassify

### Name:

geoc raster reclassify

### Description:

Reclassify a Raster

### Arguments:

- -b --band: The band
- -n --nodata: The NODATA value
- -r --range: A range: from-to=value or 1-10=5
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster reclassify -i raster.tif -o raster_reclass.tif -r 49-100=1 -r 100-256=255
```

## raster resample

### Name:

geoc raster resample

### Description:

Resample a Raster

### Arguments:

- -b --bounds: The bounding box
- -s --size: The output size
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format

## Commands

- -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

### Example:

```
geoc raster resample -i alki.tif -s "200,400" -o alki_resized.tif
```

## raster invert

### Name:

geoc raster invert

### Description:

Invert the values of a Raster

### Arguments:

- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster invert -i raster.tif -o raster_inv.tif
```

## raster stylize

### Name:

geoc raster stylize

### Description:

Create a new Raster by baking the style into an existing Raster

### Arguments:

- -s --style: The SLD style file
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster stylize -i raster.tif -o raster_stylized.tif -s raster_colormap.sld
```

## raster add constant

**Name:**

geoc raster add constant

**Description:**

Add a constant value to a Raster

**Arguments:**

- -v --value: The value
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

**Example:**

```
geoc raster add constant -i raster.tif -v 100 -o raster_add_100.tif
```

## raster add

**Name:**

geoc raster add

**Description:**

Add two Rasters together

**Arguments:**

- -k --other-raster: The other raster
- -y --other-raster-name: The other raster name
- -j --other-projection: The other projection
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

**Example:**

```
geoc raster add -i raster1.acs -p "EPSG:4326" -k raster2.acs -j "EPSG:4326" -o raster_add.tif
```

## raster subtract constant

**Name:**

geoc raster subtract constant

**Description:**

## Commands

Subtract a constant value to a Raster

### Arguments:

- -v --value: The value
- -m --from: Whether to subtract the Raster from the constant or vice versa
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster subtract constant -i raster.tif -v 50 -o raster_minus_50.tif
```

## raster subtract

### Name:

geoc raster subtract

### Description:

Subtract one Raster from another Raster

### Arguments:

- -k --other-raster: The other raster
- -y --other-raster-name: The other raster name
- -j --other-projection: The other projection
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster subtract -i raster1.acs -p "EPSG:4326" -k raster2.acs -j "EPSG:4326" -o raster...
```

## raster multiply constant

### Name:

geoc raster multiply constant

### Description:

Multiply a constant value to a Raster

### Arguments:

- -v --value: The value
- -o --output-raster: The output raster

## Commands

- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster multiply constant -i raster.tif -o raster_mul_2.tif -v 2
```

## raster multiply

### Name:

geoc raster multiply

### Description:

Multiply two Rasters together

### Arguments:

- -k --other-raster: The other raster
- -y --other-raster-name: The other raster name
- -j --other-projection: The other projection
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster multiply -i raster1.acs -p "EPSG:4326" -k raster2.acs -j "EPSG:4326" -o raster_m
```

## raster divide constant

### Name:

geoc raster divide constant

### Description:

Divide a constant value to a Raster

### Arguments:

## Commands

- -v --value: The value
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster divide constant -i raster.tif -o raster_div_2.tif -v 2
```

## raster divide

### Name:

geoc raster divide

### Description:

Divide one Raster by another Raster

### Arguments:

- -k --other-raster: The other raster
- -y --other-raster-name: The other raster name
- -j --other-projection: The other projection
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster divide -i raster1.acs -p "EPSG:4326" -k raster2.acs -j "EPSG:4326" -o raster_div
```

## raster mosaic

### Name:

geoc raster mosaic

### Description:

Mosaic a list of rasters together

### Arguments:

- -r --raster: An input Raster
- -b --bounds: The bounds
- -z --size: The size
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format

## Commands

- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc raster mosaic -r raster1.tif -r raster2.tif -o raster.tif
```

## raster envelope

### Name:

geoc raster envelope

### Description:

Get the Envelope of a Raster as a Vector Layer

### Arguments:

- -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

### Example:

```
geoc raster envelope -i raster.tif
```

## raster point

### Name:

geoc raster point

### Description:

Convert a Raster to a Point Vector Layer

### Arguments:

- -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

### Example:

```
geoc raster point -i raster.tif -o points.shp
```

## raster polygon

### Name:

geoc raster polygon

**Description:**

Convert a Raster to a Polygon Vector Layer

**Arguments:**

- -b --band: The band
- -e --inside-edges: Whether to include inside edges
- -g --region-of-interest: The region of interest
- -n --no-data: A no data value
- -a --range: A range (min,minIncluded,max,maxIncluded)
- -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

**Example:**

```
geoc raster polygon -i raster.tif -o polygons.shp
```

## raster contour

**Name:**

geoc raster contour

**Description:**

Create contours from a Raster

**Arguments:**

- -b --band: The band
- -v --level: A level or interval
- -s --simplify: Whether to simplify
- -m --smooth: Whether to smooth
- -n --bounds: The bounds
- -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

**Example:**

```
geoc raster contour -i raster.tif -o contours.shp -l 10 -s -m
```

## raster to

## Commands

### Name:

geoc raster to

### Description:

Convert a Raster from one format to another

### Arguments:

- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

### Example:

```
geoc raster to -i raster.tif -o raster.png -f worldimage
```

## raster mapalgebra

### Name:

geoc raster mapalgebra

### Description:

Perform map algebra

### Arguments:

- -s --script: The map algebra (jiffle) script
- -r --raster: An input Raster
- -b --bounds: The bounds
- -z --size: The size
- -n --output-name: The output name
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -p --output-raster-projection: The output raster projection
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc raster mapalgebra -s "dest = r1 * r2" -r "r1=raster1.acs" -r "r2=raster2.acs" -o raster
```

## raster worldfile

### Name:

geoc raster worldfile

### Description:

Create a Raster world file

### Arguments:

## Commands

- -b --bounds: The bounds
- -s --size: The size
- -f --file: The world file
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc raster worldfile -b 0,0,10,10 -s 5,5 -f test.pgw
```

## raster size

### Name:

geoc raster size

### Description:

Get the Raster size (width,height)

### Arguments:

- -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

### Example:

```
geoc raster size -i raster.tif
```

## raster projection

### Name:

geoc raster projection

### Description:

Get the Raster Projection

### Arguments:

- -t --type: The output type (epsg, id, srs, wkt)
- -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

### Example:

```
geoc raster projection -i raster.tif -t epsg
```

## raster shadedrelief

### Name:

geoc raster shadedrelief

**Description:**

Create a shaded relief raster

**Arguments:**

- -s --scale: The scale
- -a --altitude: The altitude
- -m --azimuth: The azimuth
- -x --resx: The x resolution
- -y --resy: The y resolution
- -z --zeta-factory: The zeta factory
- -g --algorithm: The algorithm
- -o --output-raster: The output raster
- -f --output-raster-format: The output raster format
- -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

**Example:**

```
geoc raster shadedrelief -i elev.tif -o shadedrelief -s 1.0 -a 45.0 -m 15.0
```

## raster style default

**Name:**

geoc raster style default

**Description:**

Create a simple Raster SLD

**Arguments:**

- -o --opacity: The opacity
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc raster style default
```

## raster style shadedrelief

**Name:**

geoc raster style shadedrelief

**Description:**

Create a shaded relief Raster SLD

**Arguments:**

- -b --brightness-only: The brightness only flag
- -r --relief-factor: The relief factor

## Commands

- -o --opacity: The opacity
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc raster style shadedrelief -r 65 -b -o 0.85
```

## raster style channel selection

### Name:

geoc raster style channel selection

### Description:

Create a channel selection Raster SLD

### Arguments:

- -r --red: The red channel name
- -g --green: The green channel name
- -b --blue: The blue channel name
- -y --gray: The gray channel name
- -o --opacity: The opacity
- --help : Print the help message
- --web-help : Open help in a browser

### Example:

```
geoc raster style channel selection -r "red,histogram,0.5" -g "green,normalize,0.25" -b "gre
```

## raster style contrast enhancement

### Name:

geoc raster style contrast enhancement

### Description:

Create a contrast enhancement Raster SLD

### Arguments:

- -m --method: The method (normalize or histogram)
- -g --gamma-value: The gamma value
- -o --opacity: The opacity
- --help : Print the help message
- --web-help : Open help in a browser

## raster style colormap

### Name:

geoc raster style colormap

### Description:

Create a color map Raster style

### Arguments:

- -v --value: A value
- -t --type: The type (intervals, values, ramp)
- -e --extended: Whether to use extended colors or not
- -o --opacity: The opacity
- --help : Print the help message
- --web-help : Open help in a browser

## raster animatedgif

**Name:**

geoc raster animatedgif

**Description:**

Create an animated GIF from a list of GIFs.

**Arguments:**

- -f --file: The GIF file
- -o --output-file: The output animated GIF file
- -d --delay: The delay between images
- -r --repeat: Whether to repeat the animation or not
- --help : Print the help message
- --web-help : Open help in a browser

**Example:**

```
geoc raster animatedgif -f image1.gif -f image2.gif -f image2.gif -o animated.gif -d 450 -r
```

## tile generate

**Name:**

geoc tile generate

**Description:**

Generate tiles

**Arguments:**

- -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

**Example:**

```
geoc tile generate -l earthquakes.mbtiles -m layerFile -s 0 -e 2 -v false
```

## tile delete

**Name:**

geoc tile delete

**Description:**

Delete tiles from a tile layer

**Arguments:**

- -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

**Example:**

```
geoc tile delete -l earthquakes.mbtiles -z 2
```

## tile pyramid

**Name:**

geoc tile pyramid

**Description:**

Get a Pyramid from a TileLayer

**Arguments:**

- -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

**Example:**

```
geoc tile pyramid -l src/test/resources/earthquakes.mbtiles
```

## tile stitch raster

**Name:**

geoc tile stitch raster

**Description:**

Stitch image tiles together to create a Raster

**Arguments:**

- -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

**Example:**

```
geoc tile stitch raster -l earthquakes.mbtiles -z 1 -o earthquakes.tif
```

## tile stitch vector

**Name:**

geoc tile stitch vector

**Description:**

Stitch vector tiles together to create a one or more Layers

**Arguments:**

- -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

**Example:**

```
geoc tile stitch vector -l "type=vectortiles format=mvt file=earthquakes/mvt name=earthquake
```

## tile vector grid

**Name:**

geoc tile vector grid

**Description:**

Create a vector grid of a tile layers cells.

**Arguments:**

- -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

**Example:**

```
geoc tile vector grid -l earthquakes.mbtiles -z 1 -o grid.shp
```

## ***tile get bounds***

**Name:**

geoc tile get bounds

**Description:**

Get the Bounds of a tile

**Arguments:**

- -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

**Example:**

```
geoc tile get bounds -p mercator -z 10 -x 245 -y 310
```

## ***tile list tiles***

**Name:**

geoc tile list tiles

**Description:**

## Build

Get a list of tiles for a given geometry

### Arguments:

- -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

### Example:

```
geoc tile list tiles -p mercator -z 10 -b "-13731759.2574,5981350.3374,-13512843.6084,609554"
```

## Build

Building geoc is very easy but you will need Java 8 and Maven 3.

Check it out:

```
git checkout https://github.com/jericks/geoc.git
```

Build it:

```
cd geoc  
mvn clean install
```

## Help

Each command contains a --help option:

```
>>> geoc vector buffer --help  
geoc vector buffer: Buffer the features of the input Layer and save them to the output Layer  
--help : Print the help message  
-c (--capstyle) VAL : The cap style  
-d (--distance) VAL : The buffer distance  
-i (--input-workspace) VAL : The input workspace  
-l (--input-layer) VAL : The input layer  
-o (--output-workspace) VAL : The output workspace  
-q (--quadrantsegments) N : The number of quadrant segments  
-r (--output-layer) VAL : The output layer  
-s (--singlesided) : Whether buffer should be single sided or not
```

There is also a man page for each subcommand:

```
>>> man geoc-vector-buffer  
geoc-vector-buffer(1)                                geoc-vector-buffer(1)  
  
NAME  
      geoc vector buffer  
  
DESCRIPTION  
      Buffer the features of the input Layer and save them to the output Layer  
  
USAGE  
      geoc vector randompoints -n 10 -g "1,1,10,10" | geoc vector buffer -d 10  
  
OPTIONS  
      -d --distance: The buffer distance
```

```
-q --quadrantsegments: The number of quadrant segments  
-s --singlesided: Whether buffer should be single sided or not  
-c --capstyle: The cap style  
-o --output-workspace: The output workspace
```

Finally, there is a bash completion script which makes using geoc with bash much easier.

Install it in your .bash\_profile:

```
source /Users/You/geoc/shell/geoc_bash_comp
```

## Indices and tables

- *genindex*
- *modindex*
- *search*