

```
Jared-Ericksons-MacBook-Pro:geo-shell jericks$ geo-shell

Welcome to the Geo Shell!
geo-shell>workspace open --name mem --params memory
Workspace mem opened!
geo-shell>layer create --workspace mem --name points --fields "the_geom=Point EPSG:4326|fid=Integer|name=String"
Created Layer points!
geo-shell>layer add --name points --values "the_geom=POINT (-122.333056 47.609722)|fid=1|name=Seattle"
Added Feature to points
geo-shell>layer add --name points --values "the_geom=POINT (-122.459444 47.241389)|fid=2|name=Tacoma"
Added Feature to points
geo-shell>layer count --name points
2
geo-shell>[]
```

Geo Shell

Jared Erickson

Version 0.7.0

Table of Contents

Introduction	1
Modules	1
Use	2
Workspace	2
Basics.....	2
Layers	3
Layer	3
Basics.....	3
Geoprocessing	25
Graticule	86
Format.....	94
Open	94
List	94
Close	95
Rasters	95
Raster.....	96
Open	96
Close	96
List	97
Info	97
Value	98
Envelope.....	99
Get Style	100
Set Style	101
Add Raster	103
Add Constant	106
Subtract Raster	108
Subtract Constant	112
Multiply Raster	114
Multiply Constant	118
Divide Raster	120
Divide Constant	124
Contours	126
Crop	128
Mosaic	129
Reclassify	131
Reproject	132
Scale	134

Shaded Relief	135
Stylize	137
Polygon.....	138
Tile	140
Open.....	140
Close	140
List.....	141
Info	141
Delete	142
Generate.....	143
Stitch Raster	145
Tiles.....	146
Vector Grid	147
Style	149
Create	149
Vector Default	152
Vector Gradient	156
Vector Unique Values	159
Vector Unique Values From Text File	161
Raster Default	164
Raster Color Map	166
Raster Palette Color Map	169
Map	172
Open	172
Close	172
List	173
Add Layer	173
Add Raster	174
Add Tile Layer	176
Remove Layer	177
Reorder	178
Layers	180
Draw	181
Display	182
Built in	184
Exit / Quit	184
Help	184
Run OS Command	185
Date	186
Script	186
System Properties	197

Version	198
Download	198
Unzip	199
Open	200

Introduction

Geo Shell is an interactive shell for geospatial analysis.



```
1. geo shell 0.0.1 (java)
Jared-Ericksons-MacBook-Pro:geo-shell jericks$ geo-shell

Welcome to the Geo Shell!
geo-shell>workspace open --name mem --params memory
Workspace mem opened!
geo-shell>layer create --workspace mem --name points --fields "the_geom=Point EPSG:4326|fid=Integer|name=String"
Created Layer points!
geo-shell>layer add --name points --values "the_geom=POINT (-122.333056 47.609722)|fid=1|name=Seattle"
Added Feature to points
geo-shell>layer add --name points --values "the_geom=POINT (-122.459444 47.241389)|fid=2|name=Tacoma"
Added Feature to points
geo-shell>layer count --name points
2
geo-shell>[]
```

Modules

Geo Shell has modules for dealing with **vectors**, **rasters**, **tiles**, **maps**, and **styles**.

For **vector** layers, you can use **workspace** commands access layers of spatial data in datasets like shapefiles, geopackages, or postgis databases. With **layer** commands you can perform geoprocessing functions like calculating centroids or buffer features.

For **raster** layers, you can use **format** commands access individual rasters from geotiffs or world images. With **raster** commands you can perform mosaic, raster algebra, or crop functions.

The **tile** commands let you create tile layers, get tiles, and get rasters from tiles.

The **style** commands let you create styles for vector layers and raster.

The **map** commands allow you to visualize vector, raster, and tile layers.

Use

You can use geo-shell interactively by typing **geo-shell** at the command line.

Or you can write scripts and then execute them from the command line by typing **geo-shell -cmdfile script.txt**

Or by using the **script --file script.txt** command within a geo-shell session.

Workspace

Workspaces hold vector layers. A Workspace can be a GeoPackage database, a directory of Shapefiles, or a PostGIS database.

Basics

You can open, close, and list Workspaces. The earliest Workspace to open is an in memory Workspace.

Open

Open a Workspace.

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Workspace name	true		
params	The connection parameters	true		

```
geo-shell> workspace open --name mem --params memory
Workspace mem opened!
```

You can open a Workspace with --params or connection parameters. You can give it a name with --name flag.

List

List open Workspaces. NOTE: No parameters

```
geo-shell> workspace list
mem = Memory
```

Listing open Workspaces give you the name and the type Workspace.

Close

Close a Workspace.

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Workspace name	true		

geo-shell> **workspace close** --name mem

Workspace mem closed!

Once you close a Workspace by name it will no longer appear with the list command.

Layers

List the Layer in a Workspaces.

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Workspace name	true		

In this example, we will open a GeoPackage database filled with data from Natural Earth.

Open a Workspace

geo-shell> **workspace open** --name naturalearth --params src/test/resources/naturalearth.gpkg
Workspace naturalearth opened!

List open Workspaces

geo-shell> **workspace layers** --name naturalearth
countries
ocean
places
states

Close a Workspace

geo-shell> **workspace close** --name naturalearth
Workspace naturalearth closed!

Layer

Basics

Open

Open a Layer.

geo-shell> **layer open** --workspace naturalearth --layer countries --name countries

Name	Description	Mandatory	Specified Default	Unspecified Default
workspace	The Workspace name	true		
layer	The Layer name	true		
name	The name	false		

geo-shell> **workspace open** --name naturalearth --params src/test/resources/naturalearth.gpkg
 Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer countries --name countries
 Opened Workspace naturalearth Layer countries as countries

geo-shell> **workspace close** --name naturalearth
 Workspace naturalearth closed!

Close

Close a Layer.

geo-shell> **layer close** --name countries

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		

geo-shell> **workspace open** --name naturalearth --params src/test/resources/naturalearth.gpkg
 Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer countries --name countries
 Opened Workspace naturalearth Layer countries as countries

geo-shell> **layer close** --name countries
 Layer countries closed!

geo-shell> **workspace close** --name naturalearth
 Workspace naturalearth closed!

List

List open Layers.

geo-shell> **layer list**



No parameters

geo-shell> **workspace open** --name naturalearth --params src/test/resources/naturalearth.gpkg
 Workspace naturalearth opened!

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer open --workspace naturalearth --layer states --name states
Opened Workspace naturalearth Layer states as states
```

```
geo-shell> layer list
countries = GeoPackage
ocean = GeoPackage
states = GeoPackage
```

```
geo-shell> workspace close --name naturalearth
Workspace naturalearth closed!
```

Schema

Inspect a Layer's Schema.

```
geo-shell> layer schema --name countries
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer schema --name countries
```

Name Type

```
the_geom MultiPolygon
ScaleRank Long
FeatureCla String
SOVEREIGNT String
SOVISO String
SOV_A3 String
LEVEL Double
TYPE String
NAME String
SORTNAME String
ADM0_A3 String
NAME_SM String
NAME_LNG String
```

TERR_String
PARENTHETI String
NAME_ALT String
LOCAL_LNG String
LOCAL_SM String
FORMER String
ABBREV_String
MAP_COLOR Double
PEOPLE Double
GDP_USDM Double
FIPS_10 String
ISO_A2 String
ISO_A3 String
ISO_N3 Double
ITU String
IOC String
FIFA String
DS String
WMO String
GAUL Double
MARC String
STANAG1059 String
GW_ID Double
DIAL Double
INTERNET_String
COG String
ACTUAL String
CAPAY String
CRPAY String
ANI String
LIBENR String
ANCNOM String
PAYS_R_GIO String
COMMENT String

```
geo-shell> workspace close --name naturalearth
Workspace naturalearth closed!
```

Count

Count the Feature in a Layer.

```
geo-shell> layer count --name countries
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer count --name countries  
177
```

```
geo-shell> workspace close --name naturalearth  
Workspace naturalearth closed!
```

Projection

Get the Projection of a Layer.

```
geo-shell> layer projection --name countries
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer projection --name countries  
EPSG:4326
```

```
geo-shell> workspace close --name naturalearth  
Workspace naturalearth closed!
```

Features

Display the Features of a Layer.

```
geo-shell> layer features --name states --filter "NAME_1='North Dakota'"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		
filter	The CQL Filter	false		
sort	A Sort parameter (fld dir)	false		
start	The start index	false		-1
max	The maximum number of records	false		-1

field	A subfield to include	false		
-------	-----------------------	-------	--	--

geo-shell> **workspace open** --name naturalearth --params src/test/resources/naturalearth.gpkg
 Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer states --name states
 Opened Workspace naturalearth Layer states as states

geo-shell> **layer features** --name states --filter "NAME_1='North Dakota'"

Feature (states.3)

the_geom = MULTIPOLYGON
 FID_1 = 31
 ScaleRank = 2
 FeatureCla = 1st Order Admin Polys
 OBJECTID = 22
 VertexCou = 223.0
 ISO = USA
 NAME_0 = United States
 NAME_1 = North Dakota
 VARNAME_1 = ND | N.D.
 NL_NAME_1 =
 HASC_1 = US.ND
 TYPE_1 = State
 ENGTTYPE_1 = State
 VALIDFR_1 = 18891102
 VALIDTO_1 = Present
 REMARKS_1 =
 Region =
 RegionVar =
 ProvNumber = 23
 NEV_Countr = United States
 FIRST_FIPS =
 FIRST_HASC =
 FIPS_1 = US38
 gadm_level = 1.0
 CheckMe = 0
 Region_Cod =
 Region_C_1 =
 ScaleRan_1 = 1
 Region_C_2 =
 Region_C_3 =
 Country_Pr =

geo-shell> **workspace close** --name naturalearth
 Workspace naturalearth closed!

Get Style

Get the Layer's style.

```
geo-shell> layer style get --name states --style target/states.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		
style	The SLD File	false		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth -layer states --name states  
Opened Workspace naturalearth Layer states as states
```

```
geo-shell> style vector default --layer states --color #1E90FF --file examples/states_simple.sld  
Default Vector Style for states written to /home/travis/build/jericks/geo-  
shell/examples/states_simple.sld!
```

```
geo-shell> layer style get --name states --style target/states.sld  
states style written to /home/travis/build/jericks/geo-shell/target/states.sld
```

```
geo-shell> workspace close --name naturalearth  
Workspace naturalearth closed!
```

```

<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:PolygonSymbolizer>
            <sld:Fill>
              <sld:CssParameter name="fill">#f2f2f2</sld:CssParameter>
            </sld:Fill>
          </sld:PolygonSymbolizer>
          <sld:LineSymbolizer>
            <sld:Stroke>
              <sld:CssParameter name="stroke">#a9a9a9</sld:CssParameter>
              <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
            </sld:Stroke>
          </sld:LineSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```

Set Style

Set a Layer's style

```
geo-shell> layer style get --name states --style target/states_simple.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		
style	The SLD or CSS File	true		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer states --name states
Opened Workspace naturalearth Layer states as states
```

```
geo-shell> style vector default --layer states --color #1E90FF --file examples/states_simple.sld
```

```
Default Vector Style for states written to /home/travis/build/jericks/geo-shell/examples/states_simple.sld!
```

```
geo-shell> layer style get --name states --style target/states_simple.sld  
states style written to /home/travis/build/jericks/geo-shell/target/states_simple.sld
```

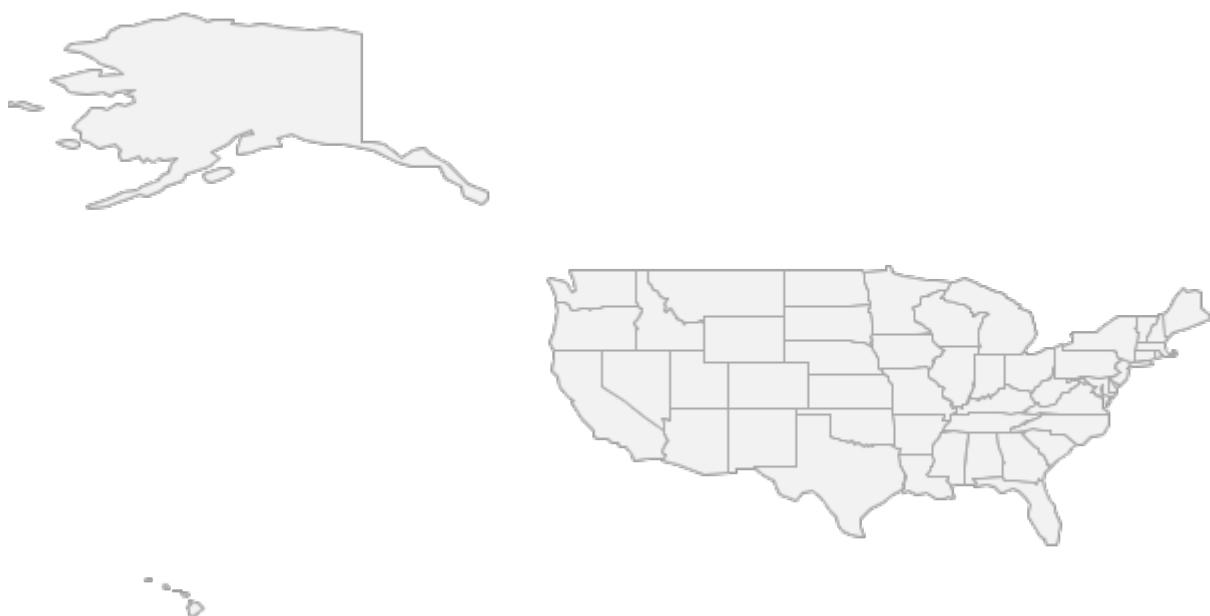
```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add layer --name map --layer states  
Added states layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_set_style.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_set_style.png!
```

```
geo-shell> map close --name map  
Map map closed!
```

```
geo-shell> workspace close --name naturalearth  
Workspace naturalearth closed!
```



Copy

Copy one Layer to another Workspace.

```
geo-shell> layer copy --input-name states_gpkg --output-workspace shapefiles --output-name states
```

Name	Description	Mandatory	Specified Default	Unspecified Default

input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
filter	The CQL Filter	false		
sort	A Sort parameter (fld dir)	false		
start	The start index	false		-1
max	The maximum number of records	false		-1
field	A subfield to include	false		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer states --name states_gpkg
Opened Workspace naturalearth Layer states as states_gpkg
```

```
geo-shell> workspace open --name shapefiles --params target/
Workspace shapefiles opened!
```

```
geo-shell> layer copy --input-name states_gpkg --output-workspace shapefiles --output-name states
Done!
```

```
geo-shell> layer count --name states
52
```

```
geo-shell> workspace close --name shapefiles
Workspace shapefiles closed!
```

```
geo-shell> workspace close --name naturalearth
Workspace naturalearth closed!
```

Create

Create a new Layer.

```
geo-shell> layer create --workspace mem --name points --fields "the_geom=Point
EPSG:4326|fid=Int|name=String"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
workspace	The Workspace name	true		

name	The new Layer name	true		
fields	The pipe delimited list of fields (name=type)	true		

```
geo-shell> workspace open --name mem --params memory
Workspace mem opened!
```

```
geo-shell> layer create --workspace mem --name points --fields "the_geom=Point
EPSG:4326|fid=Int|name=String"
Created Layer points!
```

```
geo-shell> layer schema --name points
Name Type
-----
```

```
the_geom Point
fid Integer
name String
```

Add

Add a new Feature to a Layer.

```
geo-shell> layer add --name points --values "the_geom=POINT (-122.333056
47.609722)|fid=1|name=Seattle"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		
values	The pipe delimited list of values (field=value)	true		

```
geo-shell> workspace open --name mem --params memory
Workspace mem opened!
```

```
geo-shell> layer create --workspace mem --name points --fields "the_geom=Point
EPSG:4326|fid=Int|name=String"
Created Layer points!
```

```
geo-shell> layer add --name points --values "the_geom=POINT (-122.333056
47.609722)|fid=1|name=Seattle"
Added Feature to points
```

```
geo-shell> layer add --name points --values "the_geom=POINT (-122.459444
47.241389)|fid=2|name=Tacoma"
Added Feature to points
```

```
geo-shell> layer count --name points  
2
```

Delete

Delete features from the Layer

```
geo-shell> layer delete --name points --filter "fid=2"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		
filter	The CQL Filter	true		

```
geo-shell> workspace open --name mem --params memory  
Workspace mem opened!
```

```
geo-shell> layer create --workspace mem --name points --fields "the_geom=Point  
EPSG:4326|fid=Int|name=String"  
Created Layer points!
```

```
geo-shell> layer add --name points --values "the_geom=POINT (-122.333056  
47.609722)|fid=1|name=Seattle"  
Added Feature to points
```

```
geo-shell> layer add --name points --values "the_geom=POINT (-122.459444  
47.241389)|fid=2|name=Tacoma"  
Added Feature to points
```

```
geo-shell> layer count --name points  
2
```

```
geo-shell> layer delete --name points --filter "fid=2"  
Deleted fid=2 Features from points
```

```
geo-shell> layer count --name points  
1
```

Remove

Remove a Layer from a Workspace.

```
geo-shell> layer remove --layer polygons --workspace mem
```

Name	Description	Mandatory	Specified Default	Unspecified Default
workspace	The Workspace name	true		
layer	The Layer name	true		

```

geo-shell> workspace open --name mem --params memory
Workspace mem opened!

geo-shell> layer create --workspace mem --name points --fields "the_geom=Point
EPSG:4326|fid=Int|name=String"
Created Layer points!

geo-shell> layer create --workspace mem --name lines --fields "the_geom=LineString
EPSG:4326|fid=Int|name=String"
Created Layer lines!

geo-shell> layer create --workspace mem --name polygons --fields "the_geom=Polygon
EPSG:4326|fid=Int|name=String"
Created Layer polygons!

```

geo-shell> **workspace layers** --name mem
lines
points
polygons

geo-shell> **layer remove** --layer polygons --workspace mem
Layer polygons removed from Workspace mem

geo-shell> **workspace layers** --name mem
lines
points

Update Field

Update the values of a field

```
geo-shell> layer updatefield --name points --field state --value WA
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		
field	The field name	true		
value	The value	true		
filter	The CQL Filter	false	INCLUDE	INCLUDE
script	Whether the value is a script or not	false	false	false

```

geo-shell> workspace open --name mem --params memory
Workspace mem opened!

```

```

geo-shell> layer create --workspace mem --name points --fields "the_geom=Point
EPSG:4326|fid=Int|name=String|state=String"
Created Layer points!

```

```
geo-shell> layer add --name points --values "the_geom=POINT (-122.333056  
47.609722)|fid=1|name=Seattle"
```

Added Feature to points

```
geo-shell> layer add --name points --values "the_geom=POINT (-122.459444  
47.241389)|fid=2|name=Tacoma"
```

Added Feature to points

```
geo-shell> layer updatefield --name points --field state --value WA  
Done updating state with WA!
```

```
geo-shell> layer features --name points
```

```
Feature (fid—638365e8_1750fb4acf9_-7890)
```

```
-----  
the_geom = POINT (-122.333056 47.609722)  
fid = 1  
name = Seattle  
state = WA
```

```
Feature (fid—638365e8_1750fb4acf9_-788e)
```

```
-----  
the_geom = POINT (-122.459444 47.241389)  
fid = 2  
name = Tacoma  
state = WA
```

Add Fields

Add Fields to the input Layer and save the result to the output Layer

```
geo-shell> layer addfields --input-name points --output-workspace mem --output-name points2  
--fields "name=String,state=String"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
fields	The Fields (name=type proj)	true		

```
geo-shell> workspace open --name mem --params memory  
Workspace mem opened!
```

```
geo-shell> layer create --workspace mem --name points --fields "the_geom=Point EPSG:4326"  
Created Layer points!
```

```
geo-shell> layer addfields --input-name points --output-workspace mem --output-name points2  
--fields "name=String,state=String"  
Done!
```

```
geo-shell> layer schema --name points2  
Name Type
```

```
-----  
the_geom Point  
name String  
state String
```

Add Area Field

Add area Field to the input Layer and save the result to the output Layer

```
geo-shell> layer addareafield --input-name states --output-workspace mem --output-name  
states_area --area-fieldname AREA
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
area-fieldname	The area field name	true	area	area

```
geo-shell> workspace open --name mem --params memory  
Workspace mem opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer states --name states  
Opened Workspace naturalearth Layer states as states
```

```
geo-shell> layer addareafield --input-name states --output-workspace mem --output-name  
states_area --area-fieldname AREA  
Done!
```

```
geo-shell> layer schema --name states_area  
Name Type
```

```
-----  
the_geom MultiPolygon  
FID_1 Long  
ScaleRank Long  
FeatureCla String  
OBJECTID Long
```

```
VertexCou Double
ISO String
NAME_0 String
NAME_1 String
VARNAME_1 String
NL_NAME_1 String
HASC_1 String
TYPE_1 String
ENGTYPY_1 String
VALIDFR_1 String
VALIDTO_1 String
REMARKS_1 String
Region String
RegionVar String
ProvNumber Long
NEV_Countr String
FIRST_FIPS String
FIRST_HASC String
FIPS_1 String
gadm_level Double
CheckMe Long
Region_Cod String
Region_C_1 String
ScaleRan_1 Long
Region_C_2 String
Region_C_3 String
Country_Pr String
AREA Double
```

```
geo-shell> layer features --name states_area --filter "NAME_1='North Dakota'" --field "NAME_0,AREA"
```

```
Feature (fid—638365e8_1750fb4acf9_-7889)
```

```
-----  
NAME_0 = United States  
AREA = 21.804544852979944
```

Add ID Field

Add area ID to the input Layer and save the result to the output Layer

```
geo-shell> layer addidfield --input-name places --output-workspace mem --output-name places_id  
--id-fieldname ID --start-value 1
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		

output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
id-fieldname	The id field name	true	id	id
start-value	The value to start at	true	1	1

geo-shell> **workspace open** --name mem --params memory
 Workspace mem opened!

geo-shell> **workspace open** --name naturalearth --params examples/naturalearth.gpkg
 Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer places --name places
 Opened Workspace naturalearth Layer places as places

geo-shell> **layer addidfield** --input-name places --output-workspace mem --output-name places_id
 --id-fieldname ID --start-value 1
 Done!

geo-shell> **layer schema** --name places_id
 Name Type

the_geom	Point
SCALERANK	Long
NATSCALE	Long
LABELRANK	Long
FEATURECLA	String
NAME	String
NAMEPAR	String
NAMEALT	String
DIFFASCII	Long
NAMEASCII	String
ADM0CAP	Double
CAPALT	Double
CAPIN	String
WORLDCITY	Double
MEGACITY	Long
SOV0NAME	String
SOV_A3	String
ADM0NAME	String
ADM0_A3	String
ADM1NAME	String
ISO_A2	String
NOTE	String
LATITUDE	Double
LONGITUDE	Double

CHANGED Double
NAMEDIFF Long
DIFFNOTE String
POP_MAX Long
POP_MIN Long
POP_OTHER Long
GEONAMEID Double
MEGANAME String
LS_NAME String
LS_MATCH Long
CHECKME Long
MAX_POP10 Long
MAX_POP20 Long
MAX_POP50 Long
MAX_POP300 Long
MAX_POP310 Long
MAX_NATSCA Long
MIN_AREAKM Long
MAX_AREAKM Double
MIN_AREAMI Double
MAX_AREAMI Double
MIN_PERKM Double
MAX_PERKM Double
MIN_PERMI Double
MAX_PERMI Double
MIN_BBXMIN Double
MAX_BBXMIN Double
MIN_BBXMAX Double
MAX_BBXMAX Double
MIN_BBYMIN Double
MAX_BBYMIN Double
MIN_BBymax Double
MAX_BBymax Double
MEAN_BBXC Double
MEAN_BBYC Double
COMPARE Long
GN_ASCII String
FEATURE_CL String
FEATURE_CO String
ADMIN1_COD Double
GN_POP Long
ELEVATION Double
GTOPO30 Double
TIMEZONE String
GEONAMESNO String
UN_FID Long
UN ADM0 String
UN_LAT Double

```
UN_LONG Double
POP1950 Double
POP1955 Double
POP1960 Double
POP1965 Double
POP1970 Double
POP1975 Double
POP1980 Double
POP1985 Double
POP1990 Double
POP1995 Double
POP2000 Double
POP2005 Double
POP2010 Double
POP2015 Double
POP2020 Double
POP2025 Double
POP2050 Double
CITYALT String
popDiff Long
popPerc Double
ls_gross Long
ID Integer
```

```
geo-shell> layer features --name places_id --filter "NAME='Seattle'" --field "NAME,ID"
```

```
Feature (fid—638365e8_1750fb4acf9_-79ce)
```

```
-----
```

```
NAME = Seattle
```

```
ID = 10
```

Add XY Fields

Add x and y coordinate Fields to the input Layer and save the result to the output Layer

```
geo-shell> layer addxyfields --input-name places --output-workspace mem --output-name places_xy --x-fieldname X --y-fieldname Y
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
x-fieldname	The x field name	true	x	x
y-fieldname	The y field name	true	y	y

```
geo-shell> workspace open --name mem --params memory
```

```
Workspace mem opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

```
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer places --name places
```

```
Opened Workspace naturalearth Layer places as places
```

```
geo-shell> layer addxyfields --input-name places --output-workspace mem --output-name places_xy --x-fieldname X --y-fieldname Y
```

```
Done!
```

```
geo-shell> layer schema --name places_xy
```

```
Name Type
```

```
-----  
the_geom Point  
SCALERANK Long  
NATSCALE Long  
LABELRANK Long  
FEATURECLA String  
NAME String  
NAMEPAR String  
NAMEALT String  
DIFFASCII Long  
NAMEASCII String  
ADM0CAP Double  
CAPALT Double  
CAPIN String  
WORLDCITY Double  
MEGACITY Long  
SOV0NAME String  
SOV_A3 String  
ADM0NAME String  
ADM0_A3 String  
ADM1NAME String  
ISO_A2 String  
NOTE String  
LATITUDE Double  
LONGITUDE Double  
CHANGED Double  
NAMEDIFF Long  
DIFFNOTE String  
POP_MAX Long  
POP_MIN Long  
POP_OTHER Long  
GEONAMEID Double  
MEGANAME String  
LS_NAME String
```

LS_MATCH Long
CHECKME Long
MAX_POP10 Long
MAX_POP20 Long
MAX_POP50 Long
MAX_POP300 Long
MAX_POP310 Long
MAX_NATSCA Long
MIN_AREAKM Long
MAX_AREAKM Double
MIN_AREAMI Double
MAX_AREAMI Double
MIN_PERKM Double
MAX_PERKM Double
MIN_PERMI Double
MAX_PERMI Double
MIN_BBXMIN Double
MAX_BBXMIN Double
MIN_BBXMAX Double
MAX_BBXMAX Double
MIN_BBYMIN Double
MAX_BBYMIN Double
MIN_BBymax Double
MAX_BBymax Double
MEAN_BBXC Double
MEAN_BBYC Double
COMPARE Long
GN_ASCII String
FEATURE_CL String
FEATURE_CO String
ADMIN1_COD Double
GN_POP Long
ELEVATION Double
GTOPO30 Double
TIMEZONE String
GEONAMESNO String
UN_FID Long
UN_ADMIN0 String
UN_LAT Double
UN_LONG Double
POP1950 Double
POP1955 Double
POP1960 Double
POP1965 Double
POP1970 Double
POP1975 Double
POP1980 Double
POP1985 Double

```
POP1990 Double  
POP1995 Double  
POP2000 Double  
POP2005 Double  
POP2010 Double  
POP2015 Double  
POP2020 Double  
POP2025 Double  
POP2050 Double  
CITYALT String  
popDiff Long  
popPerc Double  
ls_gross Long  
X Double  
Y Double
```

```
geo-shell> layer features --name places_xy --filter "NAME='Seattle'" --field "NAME,X,Y"
```

```
Feature (fid—638365e8_1750fb4acf9_-6084)
```

```
-----  
NAME = Seattle  
X = -122.34193084586849  
Y = 47.57194791253073
```

Validity

Check for invalid geometries in the Layer.

```
geo-shell> layer validity --name areas
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Layer name	true		
fields	A comma delimited list of Fields to include	false		

```
geo-shell> workspace open --name areas --params src/test/resources/invalid.properties  
Workspace areas opened!
```

```
geo-shell> layer open --workspace areas --layer invalid --name areas  
Opened Workspace areas Layer invalid as areas
```

```
geo-shell> layer validity --name areas  
Values Reason
```

```
-----  
invalid.1360815594529 Self-intersection
```

Geoprocessing

Clip

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

```
geo-shell> layer clip --input-name a --clip-name b --output-workspace results --output-name results
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
clip-name	The clip Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params src/test/resources/layeralgebra.gpkg
```

Workspace layers opened!

```
geo-shell> workspace open --name results --params memory
```

Workspace results opened!

```
geo-shell> layer open --workspace layers --layer a --name a
```

Opened Workspace layers Layer a as a

```
geo-shell> layer open --workspace layers --layer b --name b
```

Opened Workspace layers Layer b as b

```
geo-shell> layer clip --input-name a --clip-name b --output-workspace results --output-name results
```

Done clipping a to b to create results!

```
geo-shell> style vector default --layer a --color red --opacity 0.75 --file examples/red.sld
```

Default Vector Style for a written to /home/travis/build/jericks/geo-shell/examples/red.sld!

```
geo-shell> style vector default --layer b --color green --opacity 0.75 --file examples/green.sld
```

Default Vector Style for b written to /home/travis/build/jericks/geo-shell/examples/green.sld!

```
geo-shell> style vector default --layer results --color blue --opacity 0.75 --file examples/blue.sld
```

Default Vector Style for results written to /home/travis/build/jericks/geo-shell/examples/blue.sld!

```
geo-shell> layer style set --name a --style examples/red.sld
```

Style /home/travis/build/jericks/geo-shell/examples/red.sld set on a

```
geo-shell> layer style set --name b --style examples/green.sld
```

Style /home/travis/build/jericks/geo-shell/examples/green.sld set on b

```
geo-shell> layer style set --name results --style examples/blue.sld
```

Style /home/travis/build/jericks/geo-shell/examples/blue.sld set on results

```
geo-shell> map open --name map
```

Map map opened!

```
geo-shell> map add layer --name map --layer a
```

Added a layer to map map

```
geo-shell> map add layer --name map --layer b
```

Added b layer to map map

```
geo-shell> map add layer --name map --layer results
```

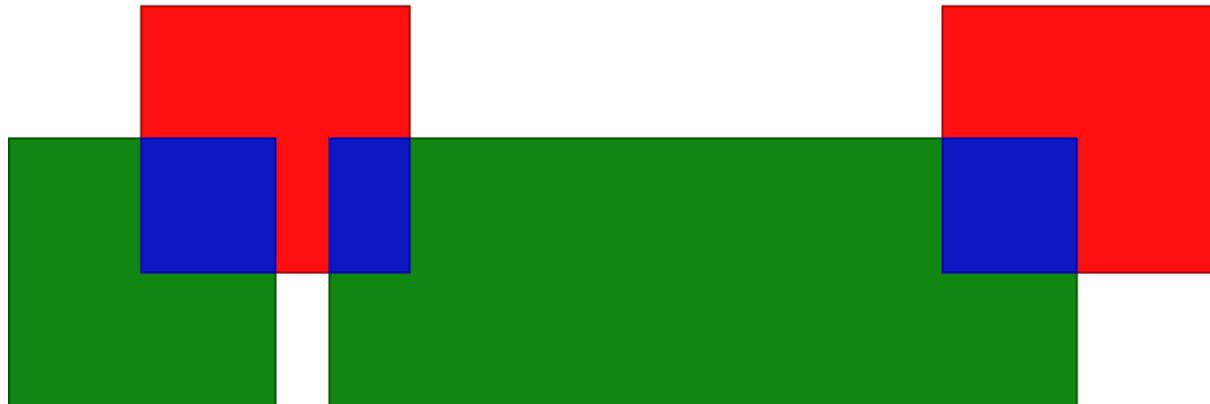
Added results layer to map map

```
geo-shell> map draw --name map --file examples/layer_clip.png
```

Done drawing /home/travis/build/jericks/geo-shell/examples/layer_clip.png!

```
geo-shell> map close --name map
```

Map map closed!



Convex Hull

Calculate the convexhull of the input Layer and save it to the output Layer.

```
geo-shell> layer convexhull --input-name countries --output-workspace layers --output-name convexhull
```

Name	Description	Mandatory	Specified Default	Unspecified Default

input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
geometry-field	The geometry field name	false	the_geom	the_geom

geo-shell> **workspace open** --name layers --params memory

Workspace layers opened!

geo-shell> **workspace open** --name naturalearth --params examples/naturalearth.gpkg

Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer countries --name countries

Opened Workspace naturalearth Layer countries as countries

geo-shell> **layer style set** --name countries --style examples/countries.sld

Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> **layer open** --workspace naturalearth --layer ocean --name ocean

Opened Workspace naturalearth Layer ocean as ocean

geo-shell> **layer style set** --name ocean --style examples/ocean.sld

Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> **layer convexhull** --input-name countries --output-workspace layers --output-name convexhull

Done!

geo-shell> **style vector default** --layer convexhull --color #1E90FF --opacity 0.25 --file examples/convexhull.sld

Default Vector Style for convexhull written to /home/travis/build/jericks/geo-shell/examples/convexhull.sld!

geo-shell> **layer style set** --name convexhull --style examples/convexhull.sld

Style /home/travis/build/jericks/geo-shell/examples/convexhull.sld set on convexhull

geo-shell> **map open** --name map

Map map opened!

geo-shell> **map add layer** --name map --layer ocean

Added ocean layer to map map

geo-shell> **map add layer** --name map --layer countries

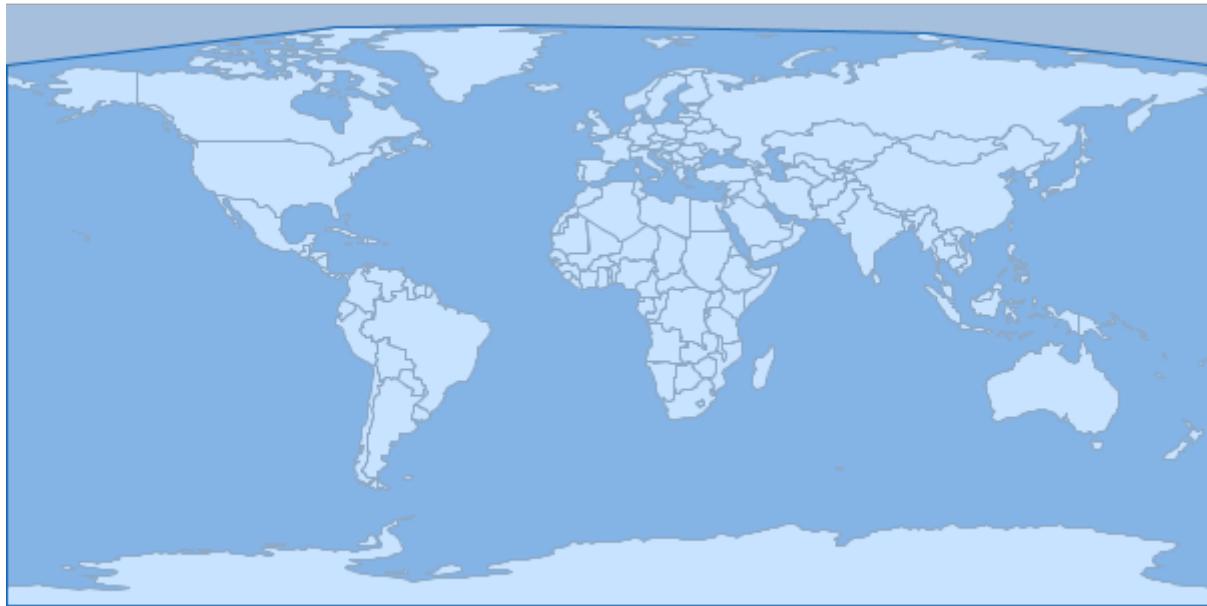
Added countries layer to map map

geo-shell> **map add layer** --name map --layer convexhull

Added convexhull layer to map map

```
geo-shell> map draw --name map --file examples/layer_convexhull.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_convexhull.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Convex Hulls

Calculate the convexhull of each Feature in the input Layer and save them to the output Layer.

```
geo-shell> layer convexhulls --input-name countries --output-workspace layers --output-name convexhulls
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> layer convexhulls --input-name countries --output-workspace layers --output-name
convexhulls
Done!

geo-shell> style vector default --layer convexhulls --color #1E90FF --opacity 0.25 --file
examples/convexhulls.sld
Default Vector Style for convexhulls written to /home/travis/build/jericks/geo-
shell/examples/convexhulls.sld!

geo-shell> layer style set --name convexhulls --style examples/convexhulls.sld
Style /home/travis/build/jericks/geo-shell/examples/convexhulls.sld set on convexhulls

geo-shell> map open --name map
Map map opened!

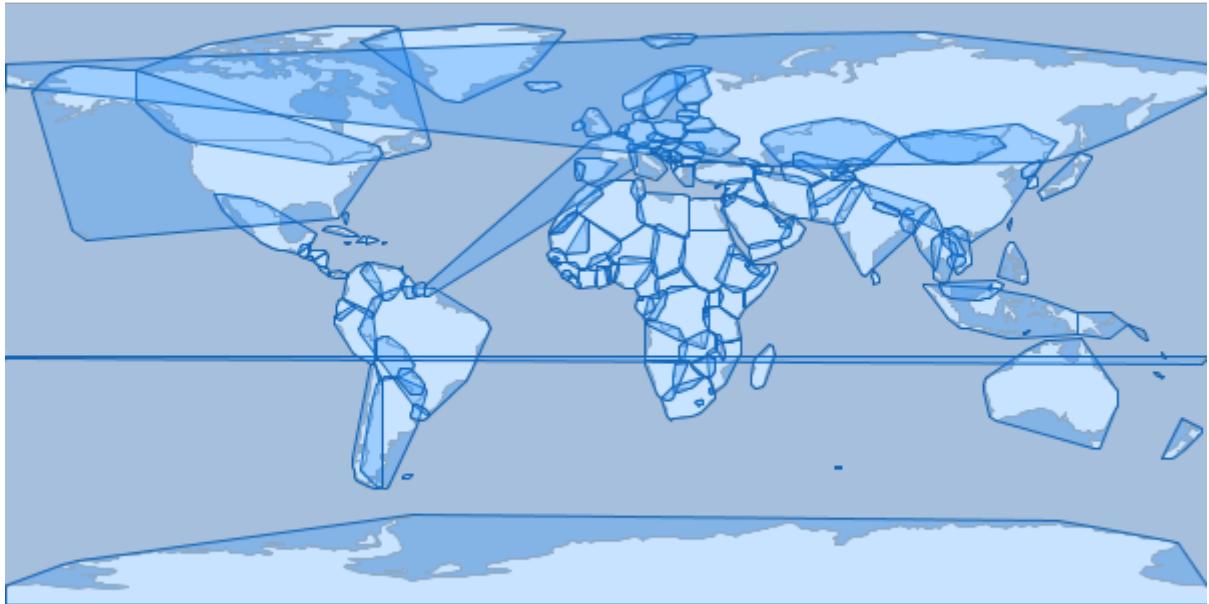
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map

geo-shell> map add layer --name map --layer countries
Added countries layer to map map

geo-shell> map add layer --name map --layer convexhulls
Added convexhulls layer to map map

geo-shell> map draw --name map --file examples/layer_convexhulls.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_convexhulls.png!

geo-shell> map close --name map
Map map closed!
```



Coordinates

Extract the coordinates each Feature in the input Layer and save them to the output Layer.

```
geo-shell> layer coordinates --input-name states --output-workspace layers --output-name coordinates
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer states --name states
Opened Workspace naturalearth Layer states as states
```

```
geo-shell> layer coordinates --input-name states --output-workspace layers --output-name coordinates
Done!
```

```
geo-shell> style vector default --layer coordinates --color #1E90FF --opacity 0.75 --file
```

```
examples/coordinates.sld  
Default Vector Style for coordinates written to /home/travis/build/jericks/geo-  
shell/examples/coordinates.sld!
```

```
geo-shell> layer style set --name coordinates --style examples/coordinates.sld  
Style /home/travis/build/jericks/geo-shell/examples/coordinates.sld set on coordinates
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map  
Map map opened!
```

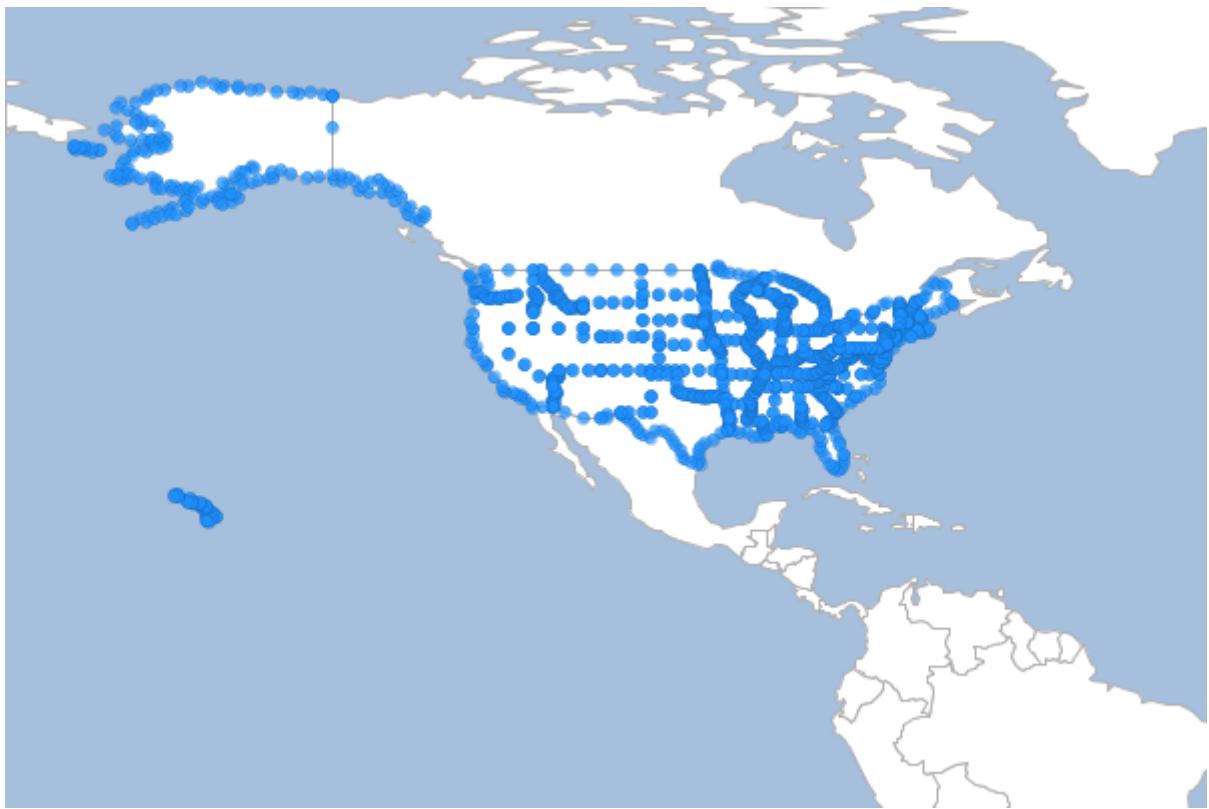
```
geo-shell> map add layer --name map --layer ocean  
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries  
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer coordinates  
Added coordinates layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_coordinates.png --bounds "-180,-8.233,-  
36.738,73.378"  
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_coordinates.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Delaunay

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

```
geo-shell> layer delaunay --input-name places --output-workspace layers --output-name delaunay
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
geometry-field	The geometry field name	false	the_geom	the_geom

```
geo-shell> workspace open --name layers --params memory
```

Workspace layers opened!

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

Workspace naturalearth opened!

```
geo-shell> layer open --workspace naturalearth --layer places --name places
```

Opened Workspace naturalearth Layer places as places

```
geo-shell> layer delaunay --input-name places --output-workspace layers --output-name delaunay
```

Done!

```
geo-shell> style vector default --layer delaunay --color #1E90FF --opacity 0.25 --file examples/delaunay.sld
Default Vector Style for delaunay written to /home/travis/build/jericks/geo-shell/examples/delaunay.sld!

geo-shell> layer style set --name delaunay --style examples/delaunay.sld
Style /home/travis/build/jericks/geo-shell/examples/delaunay.sld set on delaunay

geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries

geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> map open --name map
Map map opened!

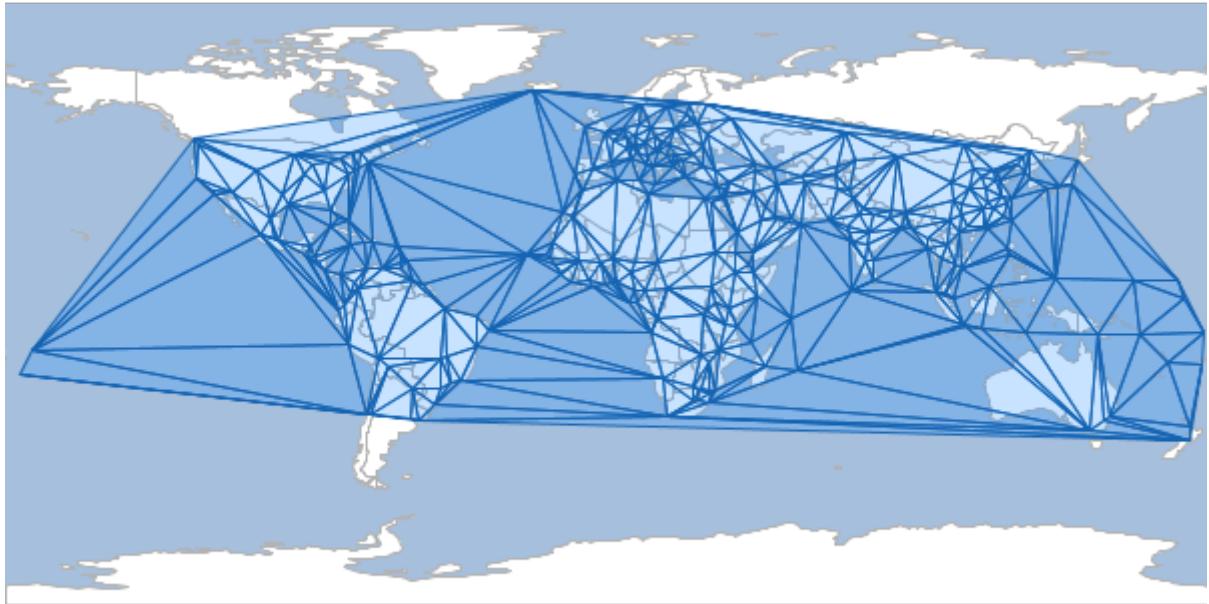
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map

geo-shell> map add layer --name map --layer countries
Added countries layer to map map

geo-shell> map add layer --name map --layer delaunay
Added delaunay layer to map map

Done drawing /home/travis/build/jericks/geo-shell/examples/layer_delaunay.png!
geo-shell> map draw --name map --file examples/layer_delaunay.png

Map map closed!
geo-shell> map close --name map
```



Densify

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

```
geo-shell> layer densify --input-name states --output-workspace layers --output-name states_densified --distance 0.1
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
distance	The distance tolerance	true		

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer states --name states
Opened Workspace naturalearth Layer states as states
```

```
geo-shell> layer densify --input-name states --output-workspace layers --output-name states_densified --distance 0.1
```

Done!

```
geo-shell> layer coordinates --input-name states_densified --output-workspace layers --output-name coordinates
```

Done!

```
geo-shell> style vector default --layer coordinates --color #1E90FF --opacity 0.75 --file examples/coordinates.sld
```

```
Default Vector Style for coordinates written to /home/travis/build/jericks/geo-shell/examples/coordinates.sld!
```

```
geo-shell> layer style set --name coordinates --style examples/coordinates.sld  
Style /home/travis/build/jericks/geo-shell/examples/coordinates.sld set on coordinates
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean  
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries  
Added countries layer to map map
```

```
Added coordinates layer to map map
```

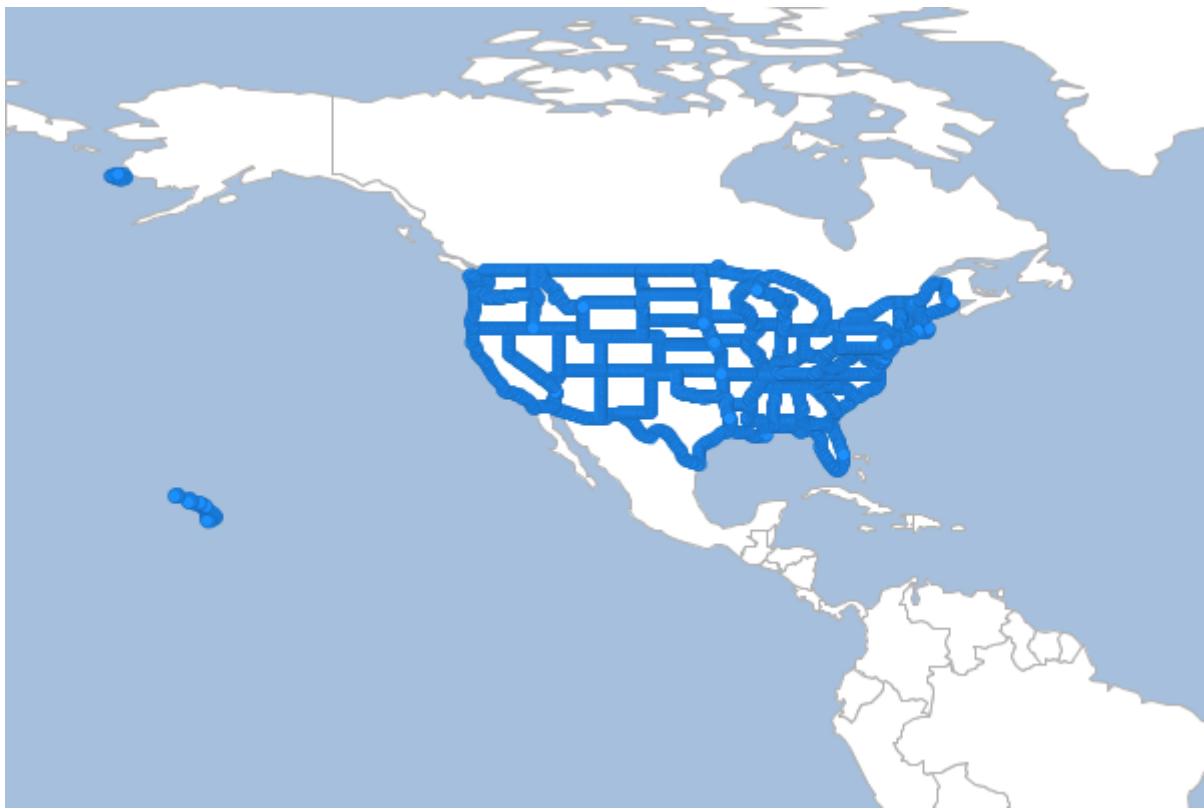
```
geo-shell> map add layer --name map --layer coordinates
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_densify.png!
```

```
geo-shell> map draw --name map --file examples/layer_densify.png --bounds "-180,-8.233,-36.738,73.378"
```

```
Map map closed!
```

```
geo-shell> map close --name map
```



Dissolve

Dissolve the Features of a Layer by a Field.

```
geo-shell> layer dissolve --input-name states --output-workspace layers --output-name regions  
--field SUB_REGION
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
field	The field to use to dissolve features	true		
idField	The name of the id field	false	id	id
countField	The name of the count field	false	count	count

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> workspace open --name shapefiles --params examples/states/states.shp  
Workspace shapefiles opened!
```

```
geo-shell> layer open --workspace shapefiles --layer states --name states
Opened Workspace shapefiles Layer states as states

geo-shell> layer dissolve --input-name states --output-workspace layers --output-name regions
--field SUB_REGION
Done dissolving states to regions by SUB_REGION!

geo-shell> style vector uniquevalues --layer regions --field SUB_REGION --colors MutedTerrain
--file [silver] examples/regions.sld
Unique Values Vector Style for regions's SUB_REGION Field written to
/home/travis/build/jericks/geo-shell/examples/regions.sld!

geo-shell> layer style set --name regions --style examples/regions.sld
Style /home/travis/build/jericks/geo-shell/examples/regions.sld set on regions

geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!

geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries

geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> map open --name map
Map map opened!

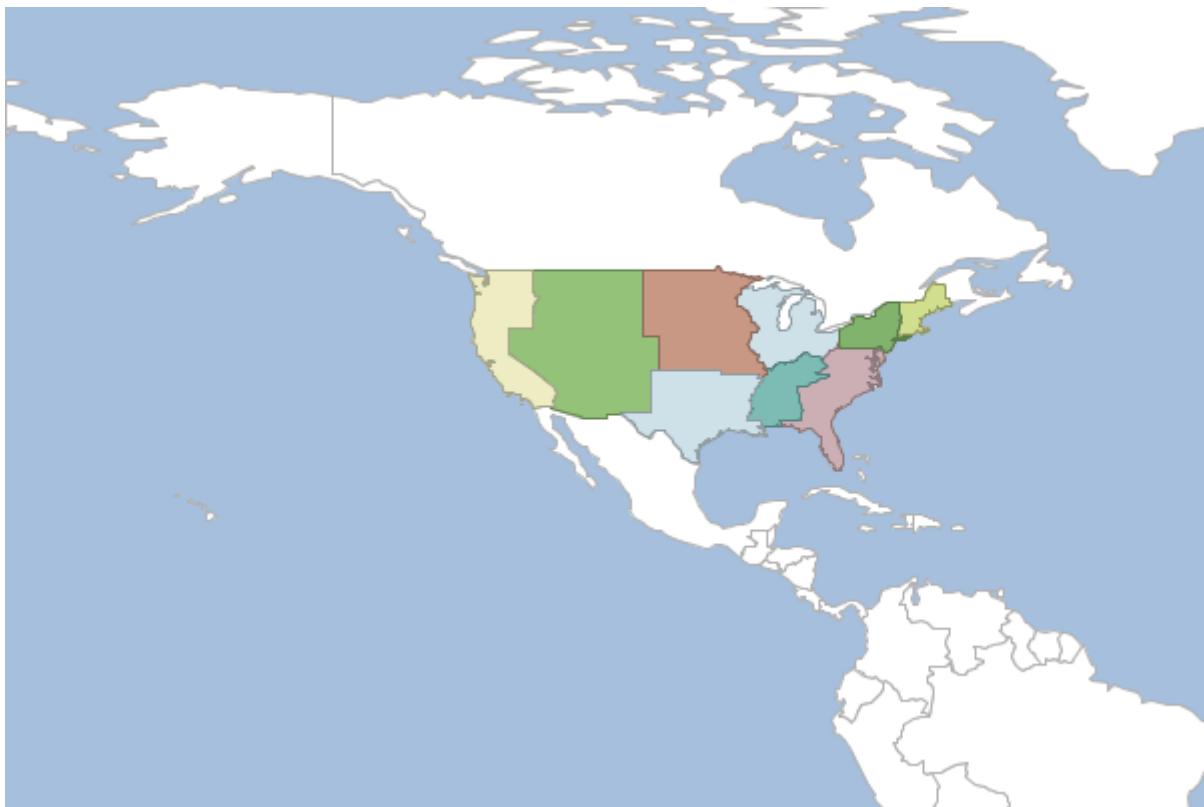
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map

geo-shell> map add layer --name map --layer countries
Added countries layer to map map

Added regions layer to map map
geo-shell> map add layer --name map --layer regions

Done drawing /home/travis/build/jericks/geo-shell/examples/layer_dissolve.png!
geo-shell> map draw --name map --file examples/layer_dissolve.png --bounds "-180,-8.233,-36.738,73.378"

Map map closed!
geo-shell> map close --name map
```



Erase

Erase one Layer from another Layer

```
geo-shell> layer erase --input-name a --other-name b --output-workspace results --output-name results
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
other-name	The other Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params src/test/resources/layeralgebra.gpkg  
Workspace layers opened!
```

```
geo-shell> workspace open --name results --params memory  
Workspace results opened!
```

```
geo-shell> layer open --workspace layers --layer a --name a  
Opened Workspace layers Layer a as a
```

```
geo-shell> layer open --workspace layers --layer b --name b  
Opened Workspace layers Layer b as b
```

```
geo-shell> layer erase --input-name a --other-name b --output-workspace results --output-name results
```

Done erasing a from b to create results!

```
geo-shell> style vector default --layer a --color red --opacity 0.75 --file examples/red.sld  
Default Vector Style for a written to /home/travis/build/jericks/geo-shell/examples/red.sld!
```

```
geo-shell> style vector default --layer b --color green --opacity 0.75 --file examples/green.sld  
Default Vector Style for b written to /home/travis/build/jericks/geo-shell/examples/green.sld!
```

```
geo-shell> style vector default --layer results --color blue --opacity 0.75 --file examples/blue.sld  
Default Vector Style for results written to /home/travis/build/jericks/geo-shell/examples/blue.sld!
```

```
geo-shell> layer style set --name a --style examples/red.sld  
Style /home/travis/build/jericks/geo-shell/examples/red.sld set on a
```

```
geo-shell> layer style set --name b --style examples/green.sld  
Style /home/travis/build/jericks/geo-shell/examples/green.sld set on b
```

```
geo-shell> layer style set --name results --style examples/blue.sld  
Style /home/travis/build/jericks/geo-shell/examples/blue.sld set on results
```

```
geo-shell> map open --name map  
Map map opened!
```

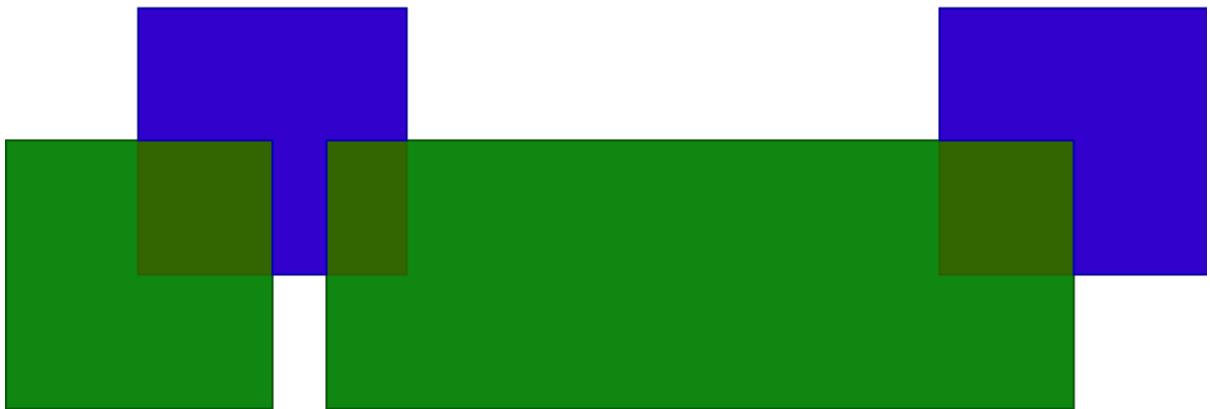
```
geo-shell> map add layer --name map --layer a  
Added a layer to map map
```

```
geo-shell> map add layer --name map --layer b  
Added b layer to map map
```

```
geo-shell> map add layer --name map --layer results  
Added results layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_erase.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_erase.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Grid Row / Column

Create a grid Layer with rows and columns

```
geo-shell> layer grid rowcol --output-workspace layers --output-name rowcol --geometry -180,-90,180,90 --rows 10 --columns 8
```

Name	Description	Mandatory	Specified Default	Unspecified Default
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
rows	The number of rows	true		
columns	The number of columns	true		
geometry	The constraining geometry	true		
type	The geometry type (point or polygon)	false	polygon	polygon
projection	The projection	false	EPSG:4326	EPSG:4326
geometry-field	The geometry field name	false	the_geom	the_geom

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> layer grid rowcol --output-workspace layers --output-name rowcol --geometry -180,-90,180,90 --rows 10 --columns 8
Done!
```

```
geo-shell> style vector default --layer rowcol --color #1E90FF --opacity 0.30 --file examples/rowcol.sld
```

```
Default Vector Style for rowcol written to /home/travis/build/jericks/geo-shell/examples/rowcol.sld!
```

```
geo-shell> layer style set --name rowcol --style examples/rowcol.sld
Style /home/travis/build/jericks/geo-shell/examples/rowcol.sld set on rowcol
```

```
geo-shell> workspace open --name naturalearth --params examples/naturelearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer rowcol
Added rowcol layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_grid_rowcol.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_grid_rowcol.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Grid Width / Height

Create a grid Layer with cell width and height

```
geo-shell> layer grid widthheight --output-workspace layers --output-name widthheight --geometry -180,-90,180,90 --cell-width 8 --cell-height 7
```

Name	Description	Mandatory	Specified Default	Unspecified Default
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
cell-width	The width of each cell	true		
cell-height	The height of each cell	true		
geometry	The constraining geometry	true		
type	The geometry type (point or polygon)	false	polygon	polygon
projection	The projection	false	EPSG:4326	EPSG:4326
geometry-field	The geometry field name	false	the_geom	the_geom

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> layer grid widthheight --output-workspace layers --output-name widthheight --geometry -180,-90,180,90 --cell-width 8 --cell-height 7  
Done!
```

```
geo-shell> style vector default --layer widthheight --color #1E90FF --opacity 0.30 --file examples/widthheight.sld  
Default Vector Style for widthheight written to /home/travis/build/jericks/geo-shell/examples/widthheight.sld!
```

```
geo-shell> layer style set --name widthheight --style examples/widthheight.sld  
Style /home/travis/build/jericks/geo-shell/examples/widthheight.sld set on widthheight
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map  
Map map opened!
```

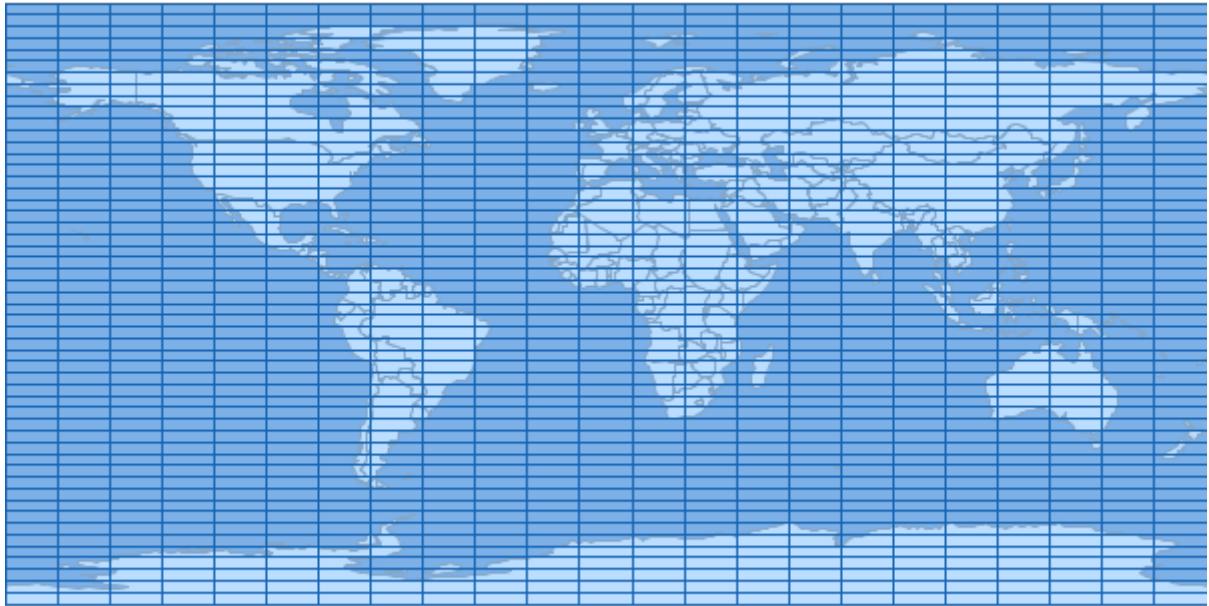
```
geo-shell> map add layer --name map --layer ocean  
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries  
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer widthheight  
Added widthheight layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_grid_widthheight.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_grid_widthheight.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Identity

Calculate the intersection between a Layer with another Layer

```
geo-shell> layer identity --input-name a --other-name b --output-workspace results --output-name results
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
other-name	The other Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
postfix-all	Whether to postfix all field names when combining schemas	false	false	false
include-duplicates	Whether to include duplicate field names	false	true	true

```
geo-shell> workspace open --name layers --params src/test/resources/layeralgebra.gpkg  
Workspace layers opened!
```

```
geo-shell> workspace open --name results --params memory
Workspace results opened!

geo-shell> layer open --workspace layers --layer a --name a
Opened Workspace layers Layer a as a

geo-shell> layer open --workspace layers --layer b --name b
Opened Workspace layers Layer b as b

geo-shell> layer identity --input-name a --other-name b --output-workspace results --output-name
results
Done calculating the identity between a and b to create results!

geo-shell> style vector default --layer a --color red --opacity 0.75 --file examples/red.sld
Default Vector Style for a written to /home/travis/build/jericks/geo-shell/examples/red.sld!

geo-shell> style vector default --layer b --color green --opacity 0.75 --file examples/green.sld
Default Vector Style for b written to /home/travis/build/jericks/geo-shell/examples/green.sld!

geo-shell> style vector default --layer results --color blue --opacity 0.75 --file examples/blue.sld
Default Vector Style for results written to /home/travis/build/jericks/geo-shell/examples/blue.sld!

geo-shell> layer style set --name a --style examples/red.sld
Style /home/travis/build/jericks/geo-shell/examples/red.sld set on a

geo-shell> layer style set --name b --style examples/green.sld
Style /home/travis/build/jericks/geo-shell/examples/green.sld set on b

geo-shell> layer style set --name results --style examples/blue.sld
Style /home/travis/build/jericks/geo-shell/examples/blue.sld set on results

geo-shell> map open --name map
Map map opened!

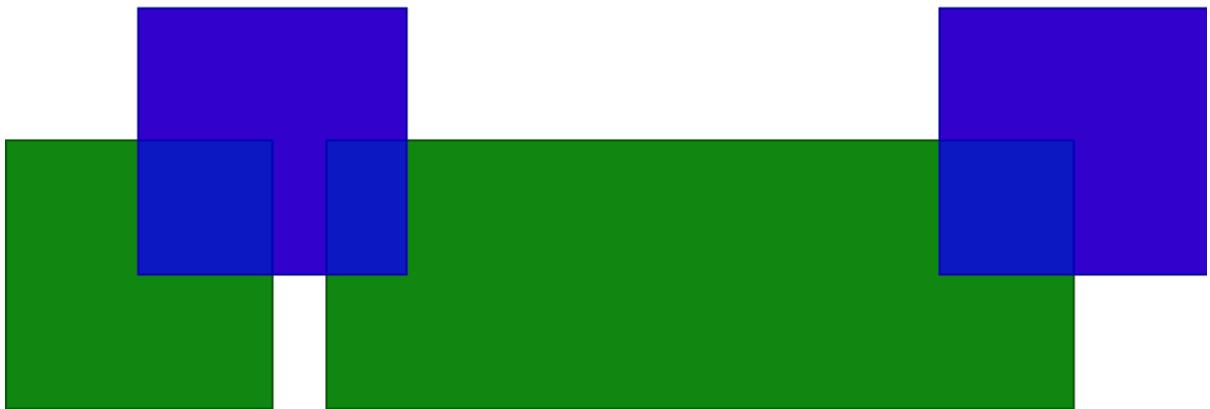
geo-shell> map add layer --name map --layer a
Added a layer to map map

geo-shell> map add layer --name map --layer b
Added b layer to map map

geo-shell> map add layer --name map --layer results
Added results layer to map map

geo-shell> map draw --name map --file examples/layer_identity.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_identity.png!

geo-shell> map close --name map
Map map closed!
```



Intersection

Calculate the intersection between a Layer with another Layer

```
geo-shell> layer intersection --input-name a --other-name b --output-workspace results --output-name results
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
other-name	The other Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
postfix-all	Whether to postfix all field names when combining schemas	false	false	false
include-duplicates	Whether to include duplicate field names	false	true	true

```
geo-shell> workspace open --name layers --params src/test/resources/layeralgebra.gpkg  
Workspace layers opened!
```

```
geo-shell> workspace open --name results --params memory
Workspace results opened!

geo-shell> layer open --workspace layers --layer a --name a
Opened Workspace layers Layer a as a

geo-shell> layer open --workspace layers --layer b --name b
Opened Workspace layers Layer b as b

geo-shell> layer intersection --input-name a --other-name b --output-workspace results --output
-name results
Done calculating the intersection between a and b to create results!

geo-shell> style vector default --layer a --color red --opacity 0.75 --file examples/red.sld
Default Vector Style for a written to /home/travis/build/jericks/geo-shell/examples/red.sld!

geo-shell> style vector default --layer b --color green --opacity 0.75 --file examples/green.sld
Default Vector Style for b written to /home/travis/build/jericks/geo-shell/examples/green.sld!

geo-shell> style vector default --layer results --color blue --opacity 0.75 --file examples/blue.sld
Default Vector Style for results written to /home/travis/build/jericks/geo-shell/examples/blue.sld!

geo-shell> layer style set --name a --style examples/red.sld
Style /home/travis/build/jericks/geo-shell/examples/red.sld set on a

geo-shell> layer style set --name b --style examples/green.sld
Style /home/travis/build/jericks/geo-shell/examples/green.sld set on b

geo-shell> layer style set --name results --style examples/blue.sld
Style /home/travis/build/jericks/geo-shell/examples/blue.sld set on results

geo-shell> map open --name map
Map map opened!

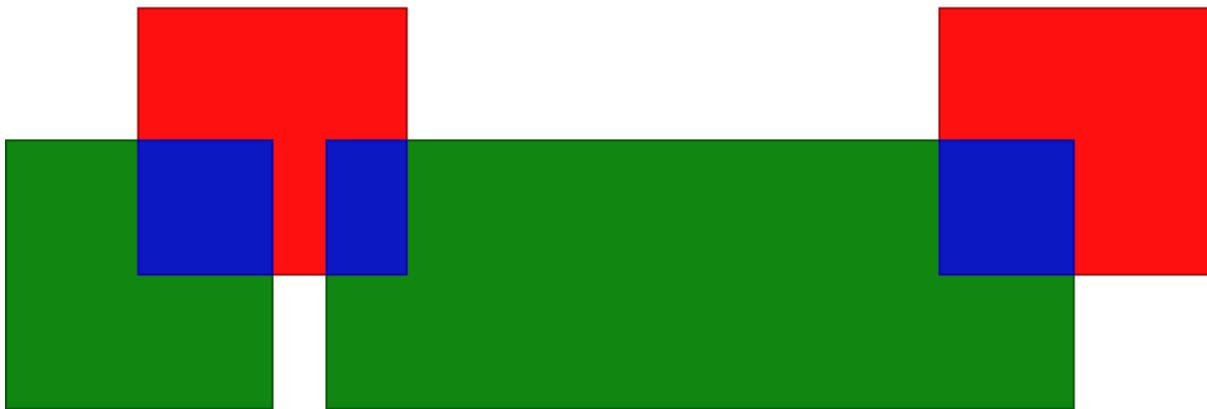
geo-shell> map add layer --name map --layer a
Added a layer to map map

geo-shell> map add layer --name map --layer b
Added b layer to map map

geo-shell> map add layer --name map --layer results
Added results layer to map map

geo-shell> map draw --name map --file examples/layer_intersection.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_intersection.png!

geo-shell> map close --name map
Map map closed!
```



Minimum Circle

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

```
geo-shell> layer mincircle --input-name countries --output-workspace layers --output-name mincircle
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
geometry-field	The geometry field name	false	the_geom	the_geom

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> layer mincircle --input-name countries --output-workspace layers --output-name
mincircle
Done!

geo-shell> style vector default --layer mincircle --color #1E90FF --opacity 0.25 --file
examples/mincircle.sld
Default Vector Style for mincircle written to /home/travis/build/jericks/geo-
shell/examples/mincircle.sld!

geo-shell> layer style set --name mincircle --style examples/mincircle.sld
Style /home/travis/build/jericks/geo-shell/examples/mincircle.sld set on mincircle

geo-shell> map open --name map
Map map opened!

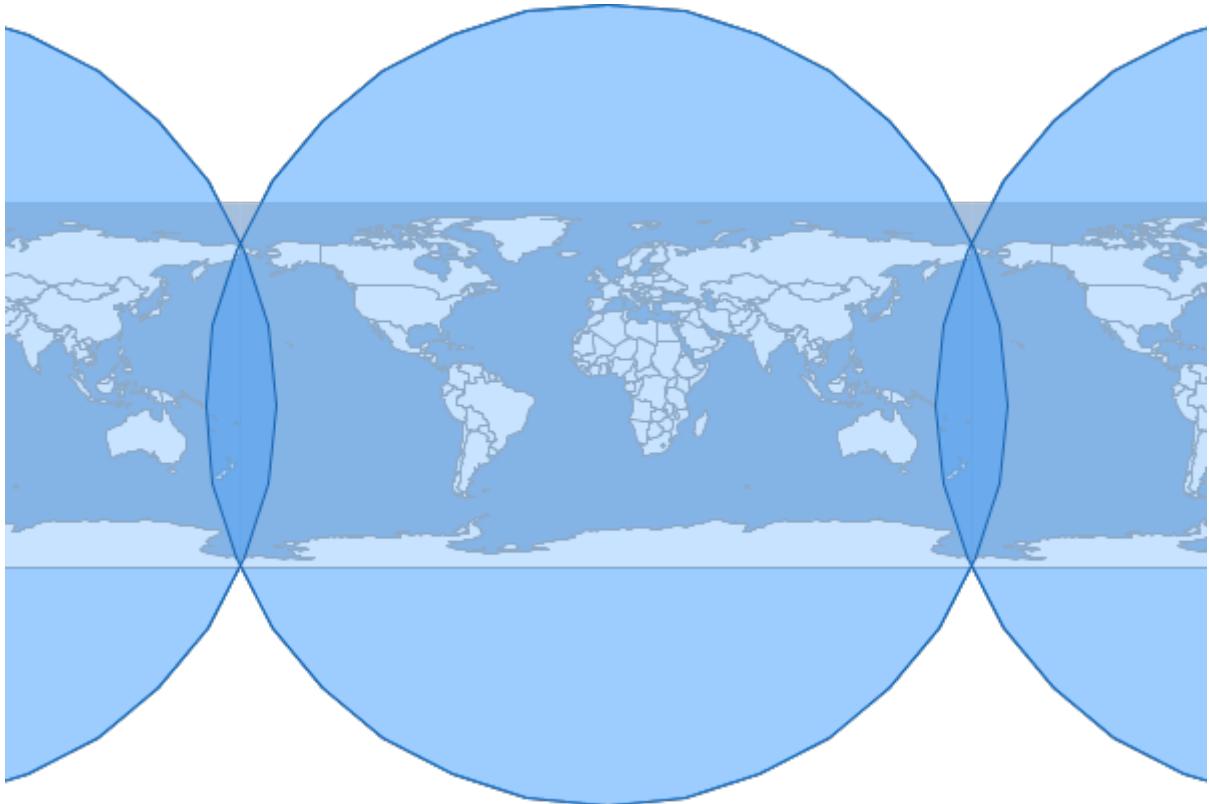
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map

geo-shell> map add layer --name map --layer countries
Added countries layer to map map

geo-shell> map add layer --name map --layer mincircle
Added mincircle layer to map map

geo-shell> map draw --name map --file examples/layer_mincircle.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_mincircle.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Minimum Circles

Calculate the minimum bounding circle of each Feature in the input Layer and save them to the output Layer.

```
geo-shell> layer mincircles --input-name countries --output-workspace layers --output-name mincircles
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
```

Opened Workspace naturalearth Layer ocean as ocean

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> layer mincircles --input-name countries --output-workspace layers --output-name
mincircles
```

Done!

```
geo-shell> style vector default --layer mincircles --color #1E90FF --opacity 0.25 --file
examples/mincircles.sld
```

```
Default Vector Style for mincircles written to /home/travis/build/jericks/geo-
shell/examples/mincircles.sld!
```

```
geo-shell> layer style set --name mincircles --style examples/mincircles.sld
Style /home/travis/build/jericks/geo-shell/examples/mincircles.sld set on mincircles
```

```
geo-shell> map open --name map
```

```
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean
```

```
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
```

```
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer mincircles
```

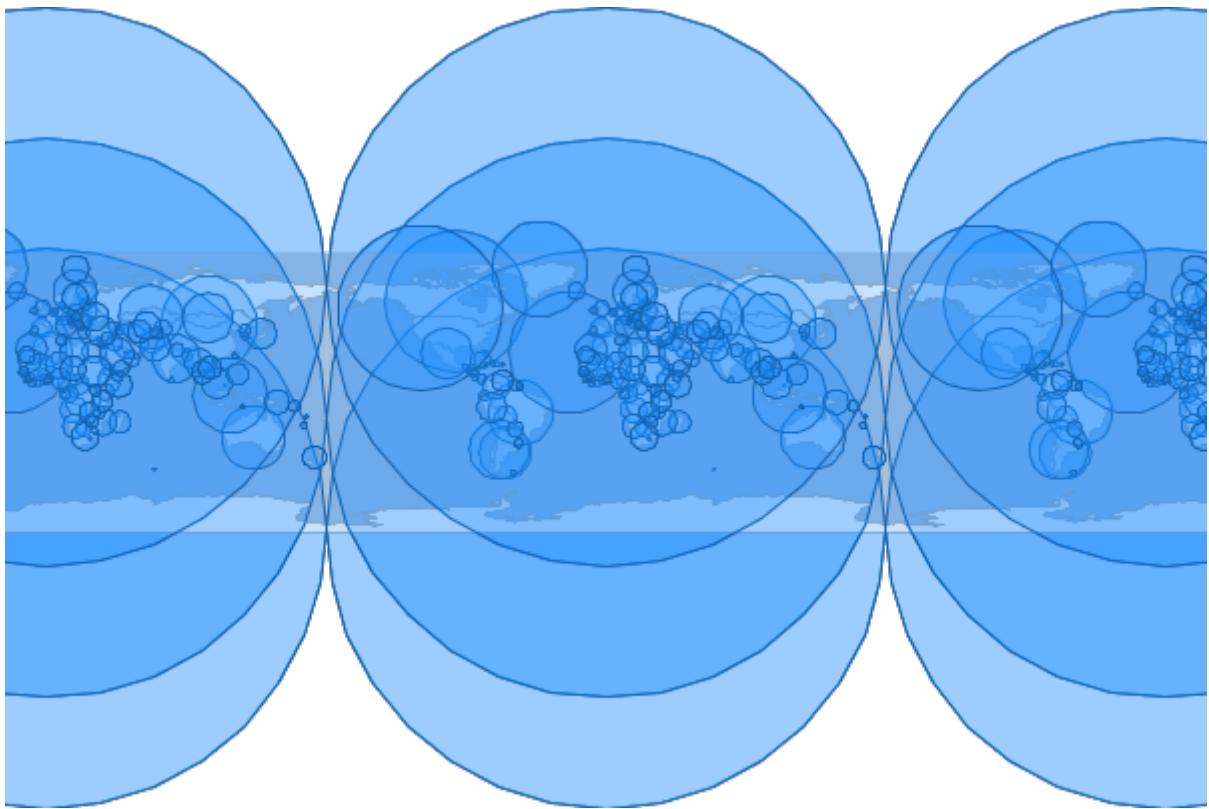
```
Added mincircles layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_mincircles.png
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_mincircles.png!
```

```
geo-shell> map close --name map
```

```
Map map closed!
```



Minimum Rectangle

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

```
geo-shell> layer minrect --input-name countries --output-workspace layers --output-name minrect
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
geometry-field	The geometry field name	false	the_geom	the_geom

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> layer minrect --input-name countries --output-workspace layers --output-name minrect
Done!
```

```
geo-shell> style vector default --layer minrect --color #1E90FF --opacity 0.25 --file
examples/minrect.sld
Default Vector Style for minrect written to /home/travis/build/jericks/geo-
shell/examples/minrect.sld!
```

```
geo-shell> layer style set --name minrect --style examples/minrect.sld
Style /home/travis/build/jericks/geo-shell/examples/minrect.sld set on minrect
```

```
geo-shell> map open --name map
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer minrect
Added minrect layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_minrect.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_minrect.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Minimum Rectangles

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

```
geo-shell> layer minrects --input-name countries --output-workspace layers --output-name minrects
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth -layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth -layer ocean --name ocean
```

Opened Workspace naturalearth Layer ocean as ocean

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> layer minrects --input-name countries --output-workspace layers --output-name
minrects
```

Done!

```
geo-shell> style vector default --layer minrects --color #1E90FF --opacity 0.25 --file
examples/minrects.sld
```

```
Default Vector Style for minrects written to /home/travis/build/jericks/geo-
shell/examples/minrects.sld!
```

```
geo-shell> layer style set --name minrects --style examples/minrects.sld
Style /home/travis/build/jericks/geo-shell/examples/minrects.sld set on minrects
```

```
geo-shell> map open --name map
```

```
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean
```

```
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
```

```
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer minrects
```

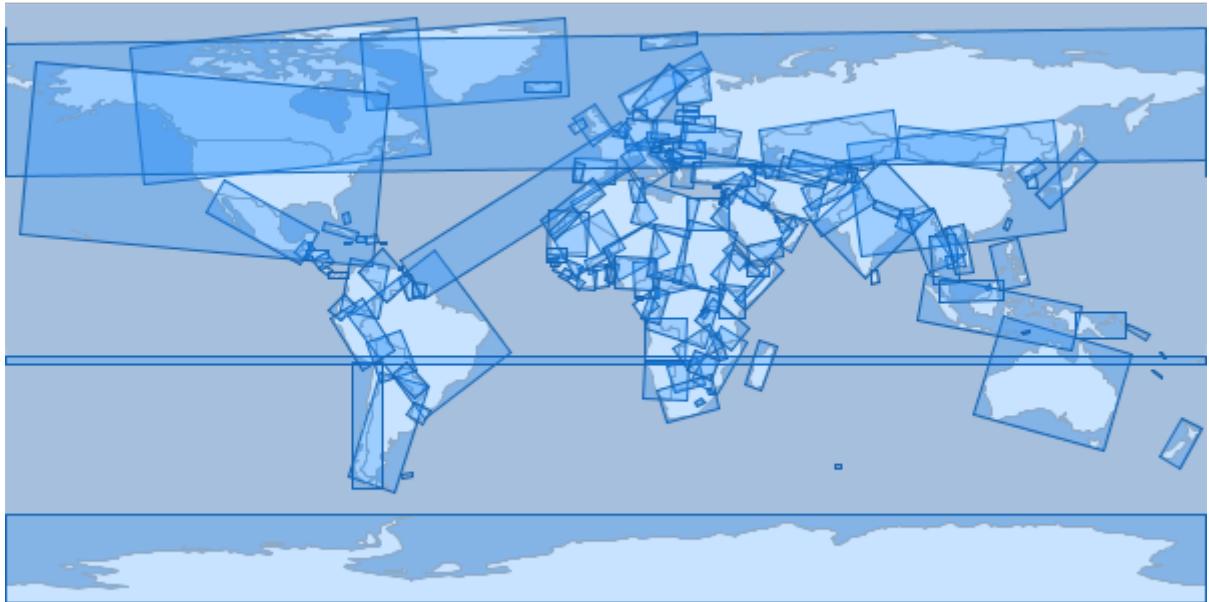
```
Added minrects layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_minrects.png
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_minrects.png!
```

```
geo-shell> map close --name map
```

```
Map map closed!
```



Octangle Envelope

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

```
geo-shell> layer octagonalenvelope --input-name countries --output-workspace layers --output-name octagonalenvelope
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
geometry-field	The geometry field name	false	the_geom	the_geom

```
geo-shell> workspace open --name layers --params memory
```

Workspace layers opened!

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

Workspace naturalearth opened!

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
```

Opened Workspace naturalearth Layer countries as countries

```
geo-shell> layer style set --name countries --style examples/countries.sld
```

Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> layer octagonalenvelope --input-name countries --output-workspace layers --output
-name octagonalenvelope
```

```
Done!
```

```
geo-shell> style vector default --layer octagonalenvelope --color #1E90FF --opacity 0.25 --file
examples/octagonalenvelope.sld
```

```
Default Vector Style for octagonalenvelope written to /home/travis/build/jericks/geo-
shell/examples/octagonalenvelope.sld!
```

```
geo-shell> layer style set --name octagonalenvelope --style examples/octagonalenvelope.sld
Style /home/travis/build/jericks/geo-shell/examples/octagonalenvelope.sld set on octagonalenvelope
```

```
geo-shell> map open --name map
```

```
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean
```

```
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
```

```
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer octagonalenvelope
```

```
Added octagonalenvelope layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_octagonalenvelope.png
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_octagonalenvelope.png!
```

```
geo-shell> map close --name map
```

```
Map map closed!
```



Octangle Envelopes

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

```
geo-shell> layer octagonalenvelopes --input-name countries --output-workspace layers --output-name octagonalenvelopes
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
```

Opened Workspace naturalearth Layer ocean as ocean

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> layer octagonalenvelopes --input-name countries --output-workspace layers --output  
-name octagonalenvelopes
```

Done!

```
geo-shell> style vector default --layer octagonalenvelopes --color #1E90FF --opacity 0.25 --file  
examples/octagonalenvelopes.sld
```

Default Vector Style for octagonalenvelopes written to /home/travis/build/jericks/geo-
shell/examples/octagonalenvelopes.sld!

```
geo-shell> layer style set --name octagonalenvelopes --style examples/octagonalenvelopes.sld  
Style      /home/travis/build/jericks/geo-shell/examples/octagonalenvelopes.sld      set      on  
octagonalenvelopes
```

```
geo-shell> map open --name map
```

Map map opened!

```
geo-shell> map add layer --name map --layer ocean
```

Added ocean layer to map map

```
geo-shell> map add layer --name map --layer countries
```

Added countries layer to map map

```
geo-shell> map add layer --name map --layer octagonalenvelopes
```

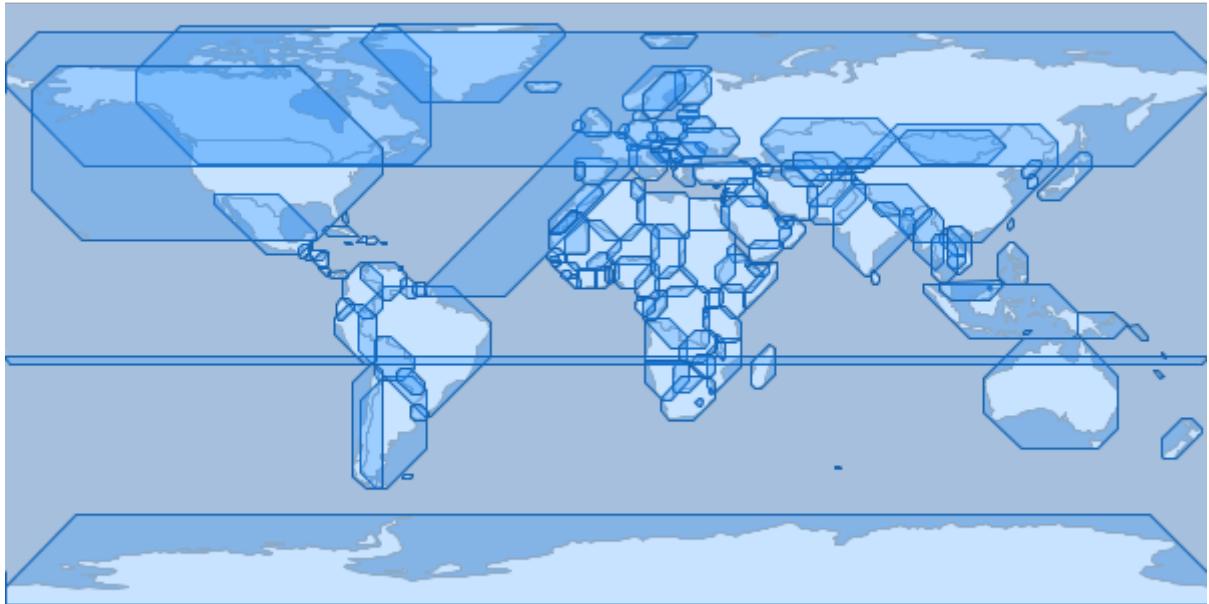
Added octagonalenvelopes layer to map map

```
geo-shell> map draw --name map --file examples/layer_octagonalenvelopes.png
```

Done drawing /home/travis/build/jericks/geo-shell/examples/layer_octagonalenvelopes.png!

```
geo-shell> map close --name map
```

Map map closed!



Points Along Lines

Create points along lines

```
geo-shell> layer points along lines --input-name mississippi --output-workspace layers --output-name points --distance 2.0
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
distance	The distance between points	true		

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> workspace open --name rivers --params
src/test/resources/rivers/ne_110m_rivers_lake_centerlines.shp
Workspace rivers opened!
```

```
geo-shell> layer open --workspace rivers --layer ne_110m_rivers_lake_centerlines --name rivers
Opened Workspace rivers Layer ne_110m_rivers_lake_centerlines as rivers
```

```
geo-shell> layer copy --input-name rivers --output-workspace layers --output-name mississippi #
```

```
[gray]--filter# "name='Mississippi'"
```

Done!

```
geo-shell> style vector default --layer mississippi --color blue --file examples/river.sld  
Default Vector Style for mississippi written to /home/travis/build/jericks/geo-shell/examples/river.sld!
```

```
geo-shell> layer style set --name mississippi --style examples/river.sld  
Style /home/travis/build/jericks/geo-shell/examples/river.sld set on mississippi
```

```
geo-shell> layer points along lines --input-name mississippi --output-workspace layers --output-name points --distance 2.0  
Done placing points along mississippi every 2.0 to create points!
```

```
geo-shell> style vector default --layer points --color green --file examples/points.sld  
Default Vector Style for points written to /home/travis/build/jericks/geo-shell/examples/points.sld!
```

```
geo-shell> layer style set --name points --style examples/points.sld  
Style /home/travis/build/jericks/geo-shell/examples/points.sld set on points
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean  
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries  
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer mississippi  
Added mississippi layer to map map
```

```
geo-shell> map add layer --name map --layer points  
Added points layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_points_along_lines.png --bounds "-180,-
```

8.233,-36.738,73.378"

Done drawing /home/travis/build/jericks/geo-shell/examples/layer_points_along_lines.png!



Simplify

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

```
geo-shell> layer simplify --input-name mississippi --output-workspace layers --output-name simplified --distance 1.0
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
algorithm	The simplify algorithm (DouglasPeucker - dp or TopologyPreserving - tp)	false	tp	tp
distance	The distance tolerance	true		

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> workspace open --name rivers --params
src/test/resources/rivers/ne_110m_rivers_lake_centerlines.shp
Workspace rivers opened!

geo-shell> layer open --workspace rivers --layer ne_110m_rivers_lake_centerlines --name rivers
Opened Workspace rivers Layer ne_110m_rivers_lake_centerlines as rivers

geo-shell> layer copy --input-name rivers --output-workspace layers --output-name mississippi #
[gray]--filter# "name='Mississippi'"
Done!

geo-shell> layer simplify --input-name mississippi --output-workspace layers --output-name
simplified --distance 1.0
Done!

geo-shell> style vector default --layer simplified --color blue --file examples/river.sld
Default Vector Style for simplified written to /home/travis/build/jericks/geo-shell/examples/river.sld

geo-shell> layer style set --name simplified --style examples/river.sld
Style /home/travis/build/jericks/geo-shell/examples/river.sld set on simplified

geo-shell> layer coordinates --input-name simplified --output-workspace layers --output-name
points
Done!

geo-shell> style vector default --layer points --color green --file examples/points.sld
Default Vector Style for points written to /home/travis/build/jericks/geo-shell/examples/points.sld

geo-shell> layer style set --name points --style examples/points.sld
Style /home/travis/build/jericks/geo-shell/examples/points.sld set on points

geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!

geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries

geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> map open --name map
Map map opened!

geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries  
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer simplified  
Added simplified layer to map map
```

```
geo-shell> map add layer --name map --layer points  
Added points layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_simplify.png --bounds "-180,-8.233,-36.738,73.378"  
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_simplify.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Symmetric Difference

Calculate the symmetric difference between a Layer and another Layer.

```
geo-shell> layer symdifference --input-name a --other-name b --output-workspace results --output-name results
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
other-name	The other Layer name	true		

output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
postfix-all	Whether to postfix all field names when combining schemas	false	false	false
include-duplicates	Whether to include duplicate field names	false	true	true

```
geo-shell> workspace open --name layers --params src/test/resources/layeralgebra.gpkg
Workspace layers opened!
```

```
geo-shell> workspace open --name results --params memory
Workspace results opened!
```

```
geo-shell> layer open --workspace layers --layer a --name a
Opened Workspace layers Layer a as a
```

```
geo-shell> layer open --workspace layers --layer b --name b
Opened Workspace layers Layer b as b
```

```
geo-shell> layer symdifference --input-name a --other-name b --output-workspace results --output-name results
```

Done calculating the symmetric difference between a and b to create results!

```
geo-shell> style vector default --layer a --color red --opacity 0.75 --file examples/red.sld
Default Vector Style for a written to /home/travis/build/jericks/geo-shell/examples/red.sld!
```

```
geo-shell> style vector default --layer b --color green --opacity 0.75 --file examples/green.sld
Default Vector Style for b written to /home/travis/build/jericks/geo-shell/examples/green.sld!
```

```
geo-shell> style vector default --layer results --color blue --opacity 0.75 --file examples/blue.sld
Default Vector Style for results written to /home/travis/build/jericks/geo-shell/examples/blue.sld!
```

```
geo-shell> layer style set --name a --style examples/red.sld
Style /home/travis/build/jericks/geo-shell/examples/red.sld set on a
```

```
geo-shell> layer style set --name b --style examples/green.sld
Style /home/travis/build/jericks/geo-shell/examples/green.sld set on b
```

```
geo-shell> layer style set --name results --style examples/blue.sld
Style /home/travis/build/jericks/geo-shell/examples/blue.sld set on results
```

```
geo-shell> map open --name map
Map map opened!
```

```
geo-shell> map add layer --name map --layer a
```

Added a layer to map map

```
geo-shell> map add layer --name map --layer b
```

Added b layer to map map

```
geo-shell> map add layer --name map --layer results
```

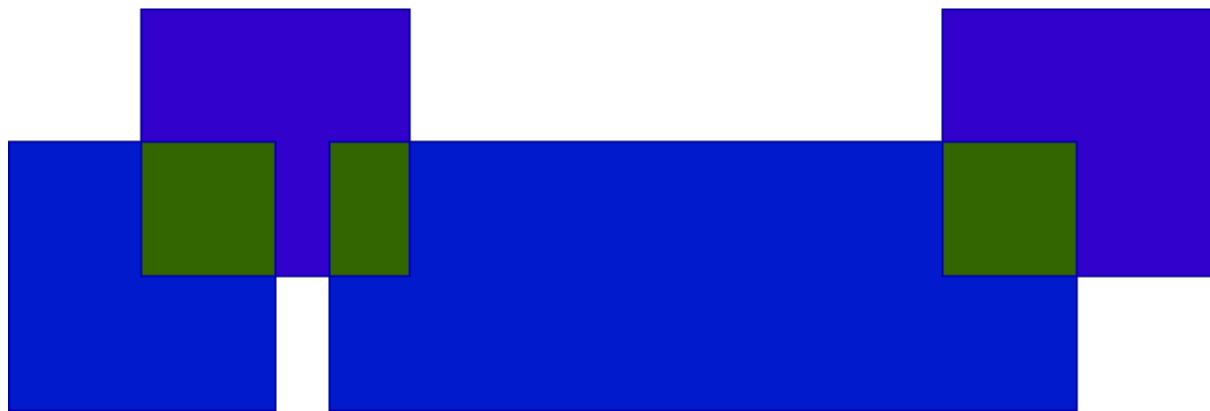
Added results layer to map map

```
geo-shell> map draw --name map --file examples/layer_symdifference.png
```

Done drawing /home/travis/build/jericks/geo-shell/examples/layer_symdifference.png!

```
geo-shell> map close --name map
```

Map map closed!



Transform

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

```
geo-shell> layer transform --input-name points --output-workspace layers --output-name polys  
--transforms "the_geom=buffer(the_geom, 5)|id=id*10"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

transforms	The pipe delimited list of transforms (field=expression or function)	true		
------------	--	------	--	--

geo-shell> **workspace open** --name layers --params memory
 Workspace layers opened!

geo-shell> **layer random** --output-workspace layers --output-name points --geometry -180,-90,180,90
 --number 100 --projection EPSG:4326
 Done!

geo-shell> **style vector default** --layer points --color #1E90FF --file examples/points.sld
 Default Vector Style for points written to /home/travis/build/jericks/geo-shell/examples/points.sld!

geo-shell> **layer style set** --name points --style examples/points.sld
 Style /home/travis/build/jericks/geo-shell/examples/points.sld set on points

geo-shell> **layer transform** --input-name points --output-workspace layers --output-name polys
 --transforms "the_geom=buffer(the_geom, 5)|id=id*10"
 Done transforming points to polys with the_geom=buffer(the_geom, 5)|id=id*10!

geo-shell> **style vector default** --layer polys --color blue --opacity 0.25 --file examples/polys.sld
 Default Vector Style for polys written to /home/travis/build/jericks/geo-shell/examples/polys.sld!

geo-shell> **layer style set** --name polys --style examples/polys.sld
 Style /home/travis/build/jericks/geo-shell/examples/polys.sld set on polys

geo-shell> **workspace open** --name naturalearth --params examples/naturalearth.gpkg
 Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer countries --name countries
 Opened Workspace naturalearth Layer countries as countries

geo-shell> **layer style set** --name countries --style examples/countries.sld
 Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> **layer open** --workspace naturalearth --layer ocean --name ocean
 Opened Workspace naturalearth Layer ocean as ocean

geo-shell> **layer style set** --name ocean --style examples/ocean.sld
 Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> **map open** --name map
 Map map opened!

geo-shell> **map add layer** --name map --layer ocean
 Added ocean layer to map map

geo-shell> **map add layer** --name map --layer countries
 Added countries layer to map map

```
geo-shell> map add layer --name map --layer polys
```

```
Added polys layer to map map
```

```
geo-shell> map add layer --name map --layer points
```

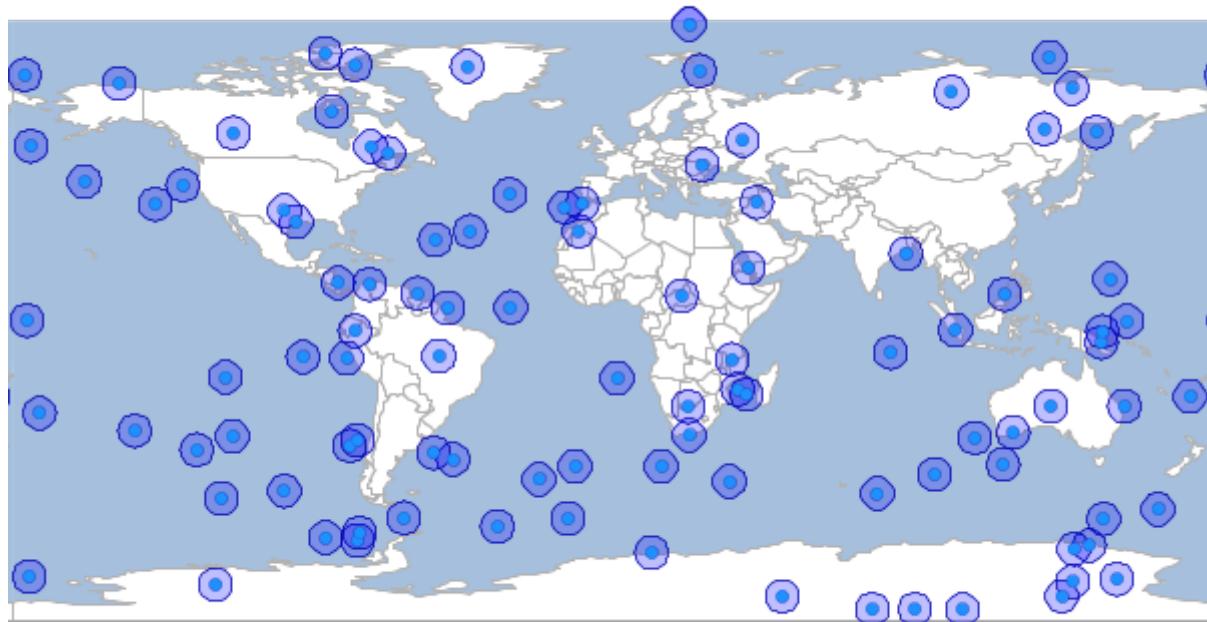
```
Added points layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_transform.png
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_transform.png!
```

```
geo-shell> map close --name map
```

```
Map map closed!
```



Union

Union a Layer with another Layer

```
geo-shell> layer union --input-name a --other-name b --output-workspace results --output-name results
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
other-name	The other Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

postfix-all	Whether to postfix all field names when combining schemas	false	false	false
include-duplicates	Whether to include duplicate field names	false	true	true

geo-shell> **workspace open** --name layers --params src/test/resources/layeralgebra.gpkg
 Workspace layers opened!

geo-shell> **workspace open** --name results --params memory
 Workspace results opened!

geo-shell> **layer open** --workspace layers --layer a --name a
 Opened Workspace layers Layer a as a

geo-shell> **layer open** --workspace layers --layer b --name b
 Opened Workspace layers Layer b as b

geo-shell> **layer union** --input-name a --other-name b --output-workspace results --output-name results
 Done unioning a and b to create results!

geo-shell> **style vector default** --layer a --color red --opacity 0.75 --file examples/red.sld
 Default Vector Style for a written to /home/travis/build/jericks/geo-shell/examples/red.sld!

geo-shell> **style vector default** --layer b --color green --opacity 0.75 --file examples/green.sld
 Default Vector Style for b written to /home/travis/build/jericks/geo-shell/examples/green.sld!

geo-shell> **style vector default** --layer results --color blue --opacity 0.75 --file examples/blue.sld
 Default Vector Style for results written to /home/travis/build/jericks/geo-shell/examples/blue.sld!

geo-shell> **layer style set** --name a --style examples/red.sld
 Style /home/travis/build/jericks/geo-shell/examples/red.sld set on a

geo-shell> **layer style set** --name b --style examples/green.sld
 Style /home/travis/build/jericks/geo-shell/examples/green.sld set on b

geo-shell> **layer style set** --name results --style examples/blue.sld
 Style /home/travis/build/jericks/geo-shell/examples/blue.sld set on results

geo-shell> **map open** --name map
 Map map opened!

geo-shell> **map add layer** --name map --layer a
 Added a layer to map map

geo-shell> **map add layer** --name map --layer b
 Added b layer to map map

```
geo-shell> map add layer --name map --layer results
```

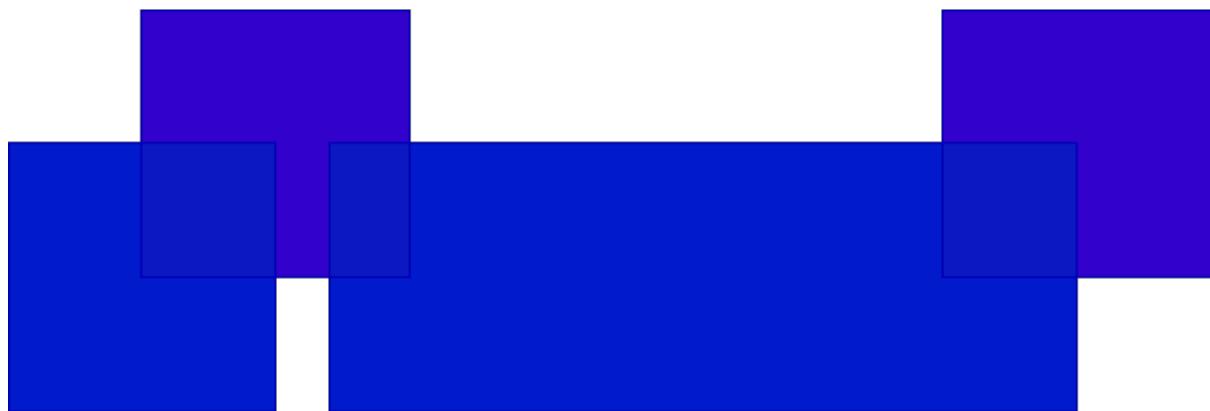
```
Added results layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_union.png
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_union.png!
```

```
geo-shell> map close --name map
```

```
Map map closed!
```



Update

Calculate the update between a Layer with another Layer

```
geo-shell> layer update --input-name a --other-name b --output-workspace results --output-name results
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
other-name	The other Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params src/test/resources/layeralgebra.gpkg
```

```
Workspace layers opened!
```

```
geo-shell> workspace open --name results --params memory
Workspace results opened!
```

```
geo-shell> layer open --workspace layers --layer a --name a
Opened Workspace layers Layer a as a
```

```
geo-shell> layer open --workspace layers --layer b --name b
Opened Workspace layers Layer b as b
```

```
geo-shell> layer update --input-name a --other-name b --output-workspace results --output-name
results
```

```
Done calculating the update between a and b to create results!
```

```
geo-shell> style vector default --layer a --color red --opacity 0.75 --file examples/red.sld
Default Vector Style for a written to /home/travis/build/jericks/geo-shell/examples/red.sld!
```

```
geo-shell> style vector default --layer b --color green --opacity 0.75 --file examples/green.sld
Default Vector Style for b written to /home/travis/build/jericks/geo-shell/examples/green.sld!
```

```
geo-shell> style vector default --layer results --color blue --opacity 0.75 --file examples/blue.sld
Default Vector Style for results written to /home/travis/build/jericks/geo-shell/examples/blue.sld!
```

```
geo-shell> layer style set --name a --style examples/red.sld
Style /home/travis/build/jericks/geo-shell/examples/red.sld set on a
```

```
geo-shell> layer style set --name b --style examples/green.sld
Style /home/travis/build/jericks/geo-shell/examples/green.sld set on b
```

```
geo-shell> layer style set --name results --style examples/blue.sld
Style /home/travis/build/jericks/geo-shell/examples/blue.sld set on results
```

```
geo-shell> map open --name map
Map map opened!
```

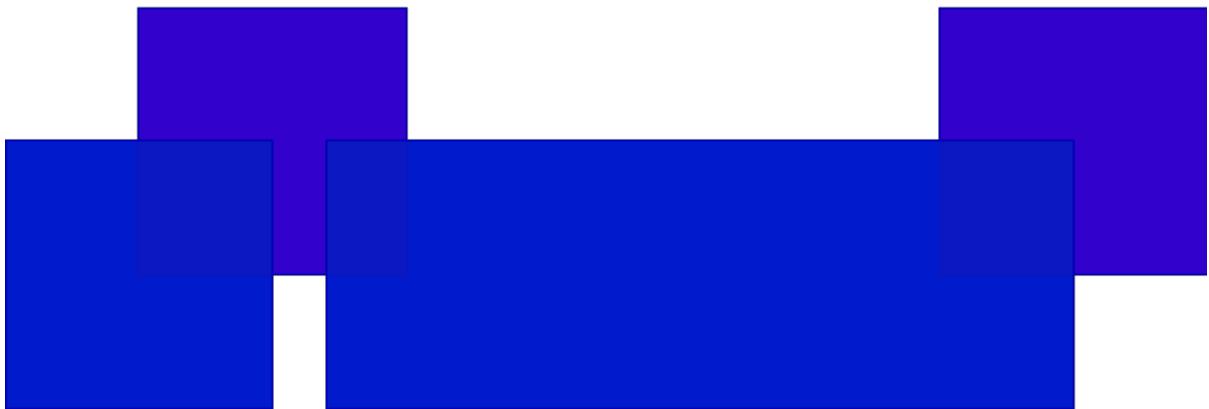
```
geo-shell> map add layer --name map --layer a
Added a layer to map map
```

```
geo-shell> map add layer --name map --layer b
Added b layer to map map
```

```
geo-shell> map add layer --name map --layer results
Added results layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_update.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_update.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Voronoi

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

```
geo-shell> layer voronoi --input-name places --output-workspace layers --output-name voronoi
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
geometry-field	The geometry field name	false	the_geom	the_geom

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer places --name places  
Opened Workspace naturalearth Layer places as places
```

```
geo-shell> layer voronoi --input-name places --output-workspace layers --output-name voronoi  
Done!
```

```
geo-shell> style vector default --layer voronoi --color #1E90FF --opacity 0.25 --file examples/voronoi.sld
```

```
Default Vector Style for voronoi written to /home/travis/build/jericks/geo-shell/examples/voronoi.sld!
```

```
geo-shell> layer style set --name voronoi --style examples/voronoi.sld  
Style /home/travis/build/jericks/geo-shell/examples/voronoi.sld set on voronoi
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean  
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries  
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer voronoi  
Added voronoi layer to map map
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_voronoi.png!  
geo-shell> map draw --name map --file examples/layer_voronoi.png --bounds -180,-90,180,90
```

```
Map map closed!
```

```
geo-shell> map close --name map
```



Random Points

Create a Layer with a number of randomly located points

```
geo-shell> layer random --output-workspace layers --output-name points --geometry -180,-90,180,90  
--number 100 --projection EPSG:4326
```

Name	Description	Mandatory	Specified Default	Unspecified Default
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
number	The number of points	true		
geometry	The geometry or bounds in which to create the points	true		
projection	The projection	true		
id-field	The id field name	false	id	id
geometry-field	The geometry field name	false	the_geom	the_geom
grid	Whether to create points in a grid	false	false	false

constrained-to-circle	Whether points should be constrained to a circle	false	false	false
gutter-fraction	The size of gutter between cells	false	0	0

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> layer random --output-workspace layers --output-name points --geometry -180,-90,180,90
--number 100 --projection EPSG:4326
Done!
```

```
geo-shell> style vector default --layer points --color #1E90FF --file examples/points.sld
Default Vector Style for points written to /home/travis/build/jericks/geo-shell/examples/points.sld!
```

```
geo-shell> layer style set --name points --style examples/points.sld
Style /home/travis/build/jericks/geo-shell/examples/points.sld set on points
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name randomMap
Map randomMap opened!
```

```
geo-shell> map add layer --name randomMap --layer ocean
Added ocean layer to map randomMap
```

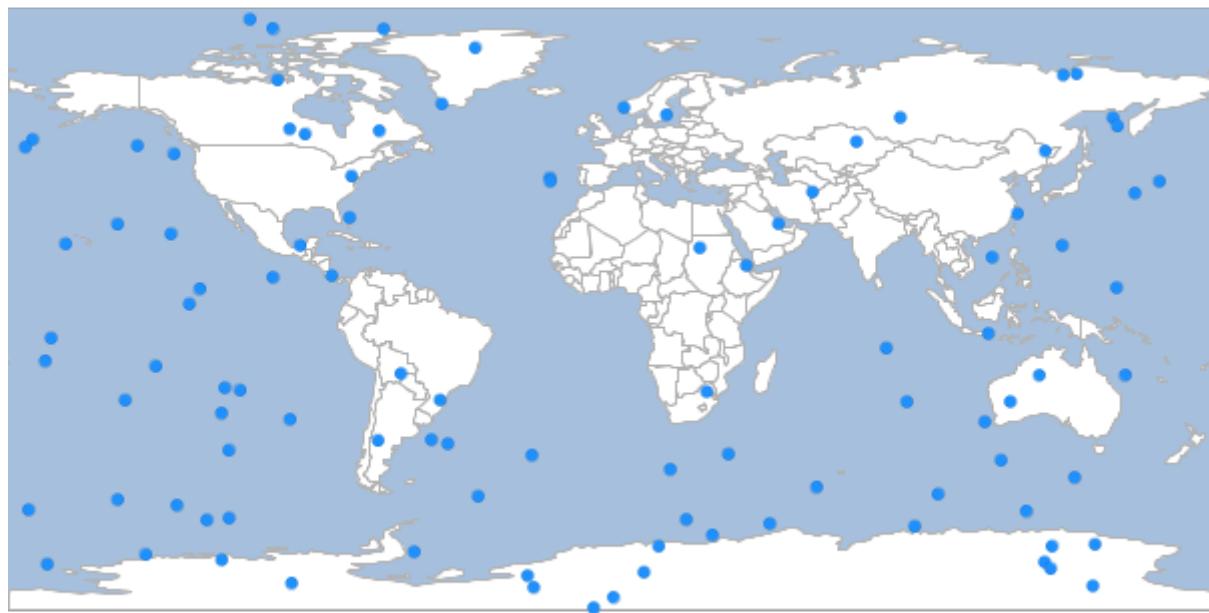
```
geo-shell> map add layer --name randomMap --layer countries
Added countries layer to map randomMap
```

```
geo-shell> map add layer --name randomMap --layer points
Added points layer to map randomMap
```

```
geo-shell> map draw --name randomMap --file examples/random_points.png
Done drawing /home/travis/build/jericks/geo-shell/examples/random_points.png!
```

```
geo-shell> map close --name randomMap
```

Map randomMap closed!



Buffer

Buffer the input Layer to the output Layer.

```
geo-shell> layer buffer --input-name points --output-workspace layers --output-name buffers  
--distance 10
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
distance	The buffer distance	true		

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> layer random --output-workspace layers --output-name points --geometry -180,-90,180,90  
--number 100 --projection EPSG:4326  
Done!
```

```
geo-shell> layer buffer --input-name points --output-workspace layers --output-name buffers  
--distance 10
```

Done!

```
geo-shell> style vector default --layer points --color #1E90FF --file examples/points.sld
Default Vector Style for points written to /home/travis/build/jericks/geo-shell/examples/points.sld!

geo-shell> style vector default --layer buffers --color #1E90FF --opacity 0.25 --file
examples/buffers.sld
Default Vector Style for buffers written to /home/travis/build/jericks/geo-shell/examples/buffers.sld!

geo-shell> layer style set --name points --style examples/points.sld
Style /home/travis/build/jericks/geo-shell/examples/points.sld set on points

geo-shell> layer style set --name buffers --style examples/buffers.sld
Style /home/travis/build/jericks/geo-shell/examples/buffers.sld set on buffers

geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!

geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries

geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> map open --name map
Map map opened!

geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map

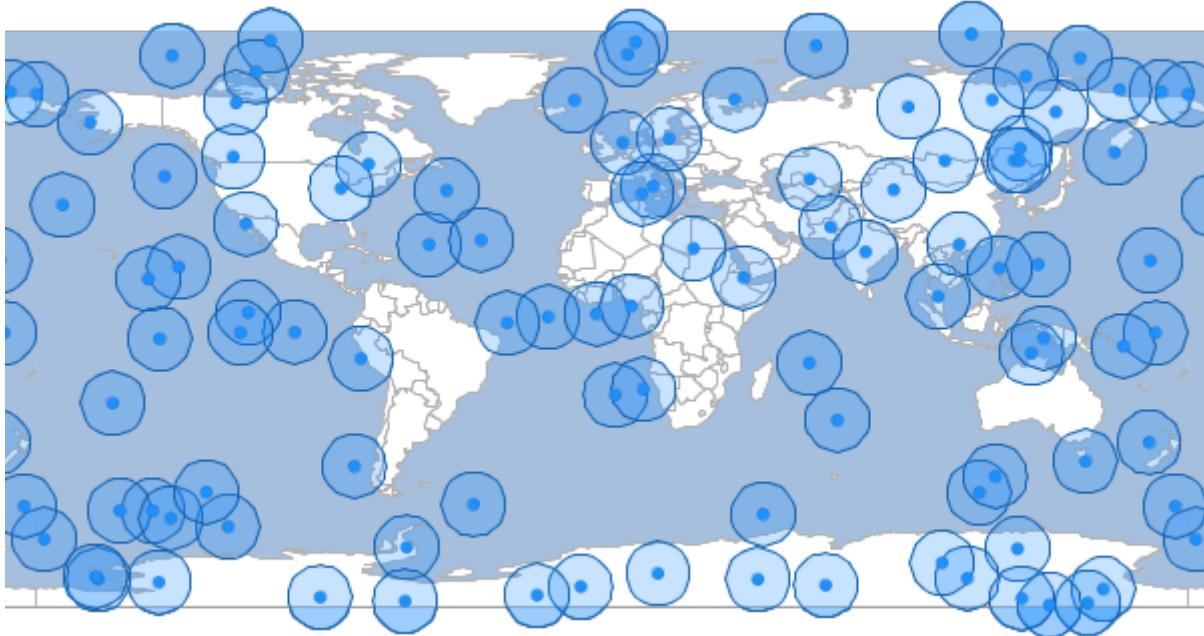
geo-shell> map add layer --name map --layer countries
Added countries layer to map map

geo-shell> map add layer --name map --layer buffers
Added buffers layer to map map

geo-shell> map add layer --name map --layer points
Added points layer to map map

geo-shell> map draw --name map --file examples/layer_buffer.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_buffer.png!

geo-shell> map close --name map
Map map closed!
```



Centroid

Calculate the centroids of the input Layer to the output Layer.

```
geo-shell> layer centroid --input-name countries --output-name centroids --output-workspace layers
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer centroid --input-name countries --output-name centroids --output-workspace layers
```

Done!

```
geo-shell> style vector default --layer centroids --color #1E90FF --file examples/centroids.sld
Default Vector Style for centroids written to /home/travis/build/jericks/geo-
shell/examples/centroids.sld!
```

```
geo-shell> layer style set --name centroids --style examples/centroids.sld
Style /home/travis/build/jericks/geo-shell/examples/centroids.sld set on centroids
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map
Map map opened!
```

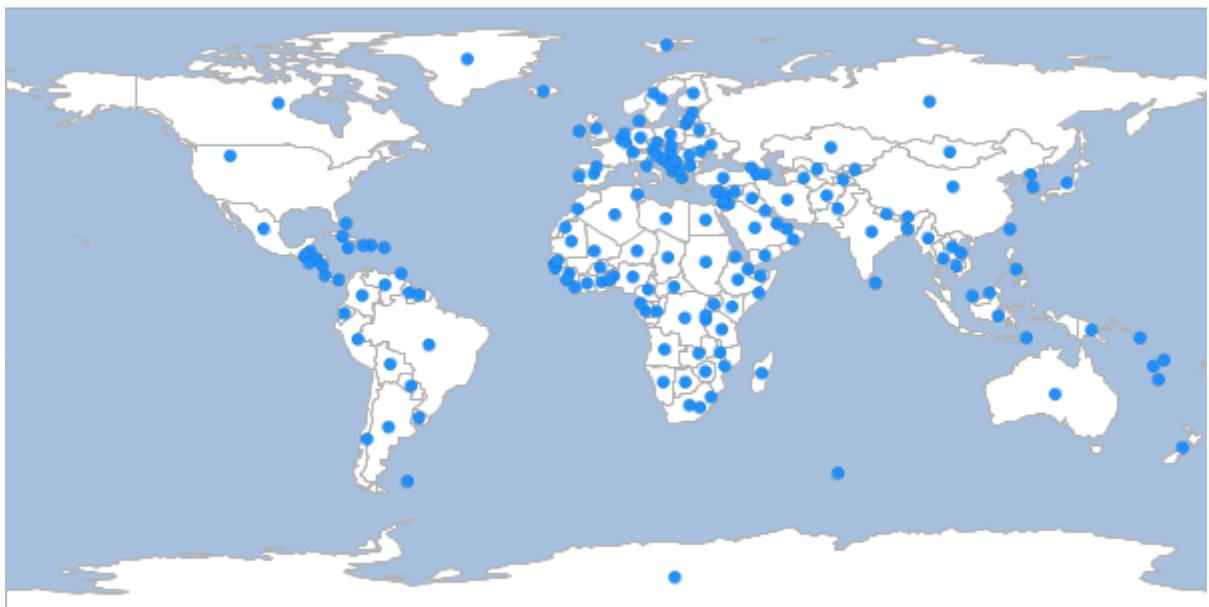
```
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer centroids
Added centroids layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_centroid.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_centroid.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Interior Point

Calculate the interior points of the input Layer to the output Layer.

```
geo-shell> layer interiorpoint --input-name countries --output-name interiorpoints --output
-workspace layers
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer interiorpoint --input-name countries --output-name interiorpoints --output
-workspace layers
```

Done!

```
geo-shell> style vector default --layer interiorpoints --color #1E90FF --file examples/interiorpoints.sld
Default Vector Style for interiorpoints written to /home/travis/build/jericks/geo-shell/examples/interiorpoints.sld!
```

```
geo-shell> layer style set --name interiorpoints --style examples/interiorpoints.sld
Style /home/travis/build/jericks/geo-shell/examples/interiorpoints.sld set on interiorpoints
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map
Map map opened!
```

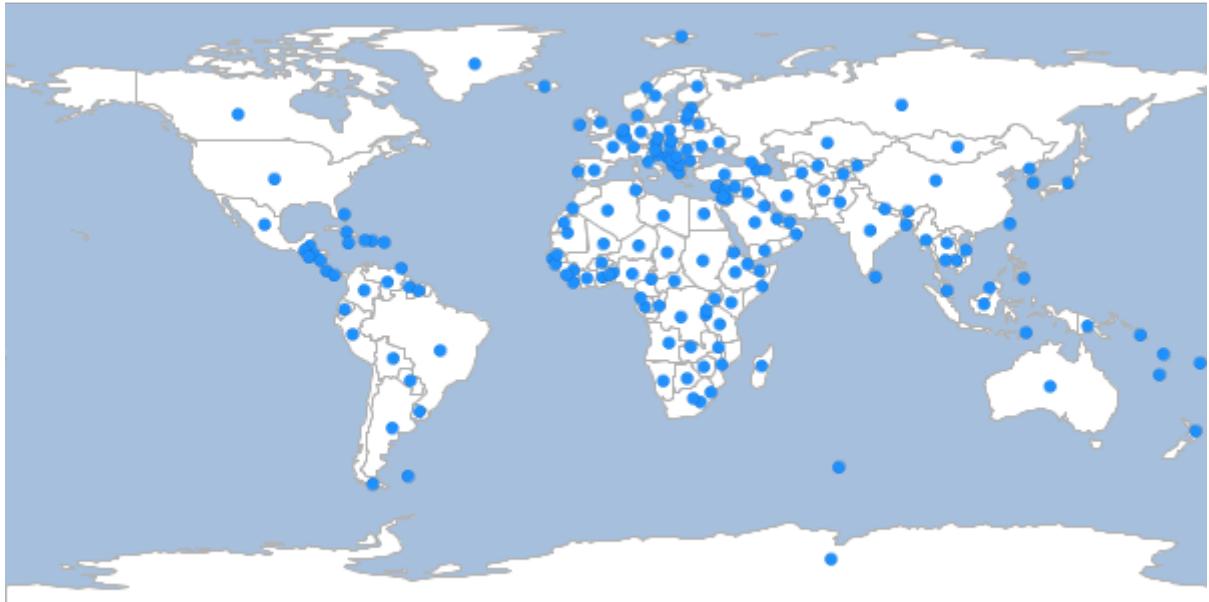
```
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer interiorpoints
Added interiorpoints layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_interiorpoint.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_interiorpoint.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Extent

Calculate the extent of the input Layer and save it to the output Layer.

```
geo-shell> layer extent --input-name states --output-workspace layers --output-name usa
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
geometry-field	The geometry field name	false	the_geom	the_geom

```
geo-shell> workspace open --name layers --params memory
```

Workspace layers opened!

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

Workspace naturalearth opened!

```
geo-shell> layer style set --name states --style examples/states.sld
```

Unable to find Layer states

```
geo-shell> layer open --workspace naturalearth --layer states --name states
```

Opened Workspace naturalearth Layer states as states

```
geo-shell> layer extent --input-name states --output-workspace layers --output-name usa  
Done!
```

```
geo-shell> style vector default --layer usa --color #1E90FF --opacity 0.25 --file examples/extent.sld  
Default Vector Style for usa written to /home/travis/build/jericks/geo-shell/examples/extent.sld!
```

```
geo-shell> layer style set --name usa --style examples/extent.sld  
Style /home/travis/build/jericks/geo-shell/examples/extent.sld set on usa
```

```
geo-shell> layer open --workspace naturalearth -layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth -layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean  
Added ocean layer to map map
```

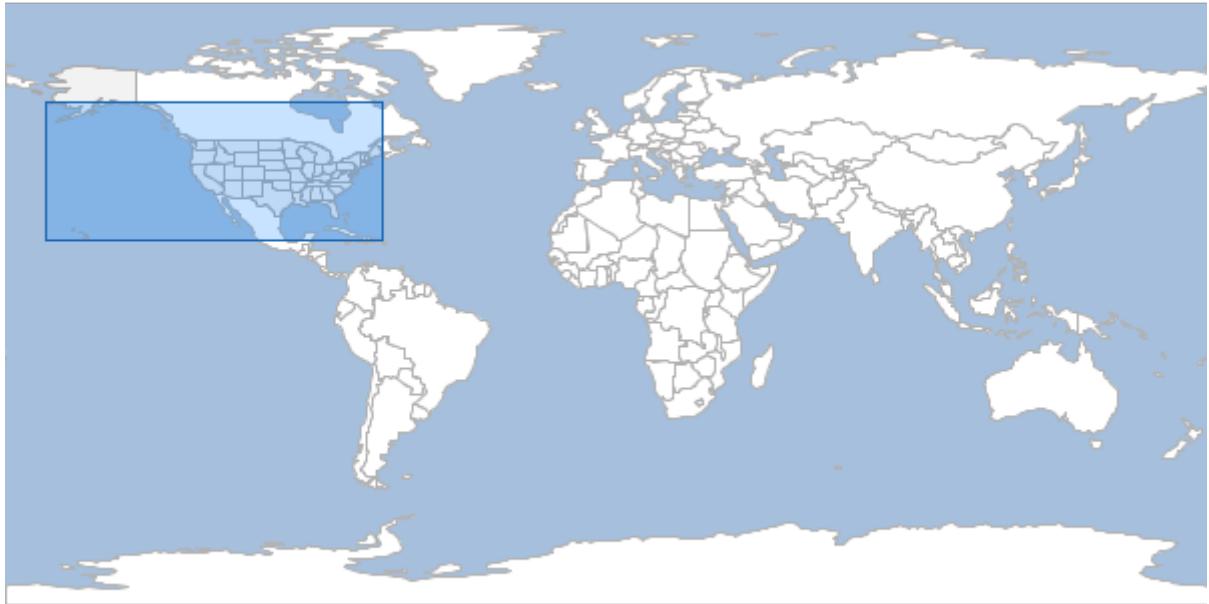
```
geo-shell> map add layer --name map --layer countries  
Added countries layer to map map
```

```
geo-shell> map add layer --name map --layer states  
Added states layer to map map
```

```
geo-shell> map add layer --name map --layer usa  
Added usa layer to map map
```

```
geo-shell> map draw --name map --file examples/layer_extent.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_extent.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Extents

Calculate the extents of each Feature in the input Layer and save them to the output Layer.

```
geo-shell> layer extents --input-name states --output-workspace layers --output-name state_extents
```

Name	Description	Mandatory	Specified Default	Unspecified Default
input-name	The Layer name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> workspace open --name layers --params memory
```

Workspace layers opened!

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

Workspace naturalearth opened!

```
geo-shell> layer style set --name states --style examples/states.sld
```

Unable to find Layer states

```
geo-shell> layer open --workspace naturalearth --layer states --name states
```

Opened Workspace naturalearth Layer states as states

```
geo-shell> layer extents --input-name states --output-workspace layers --output-name state_extents
```

Done!

```
geo-shell> style vector default --layer state_extents --color #1E90FF --opacity 0.25 --file examples/extent.sld
Default Vector Style for state_extents written to /home/travis/build/jericks/geo-shell/examples/extent.sld!

geo-shell> layer style set --name state_extents --style examples/extent.sld
Style /home/travis/build/jericks/geo-shell/examples/extent.sld set on state_extents

geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries

geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> map open --name map
Map map opened!

geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map

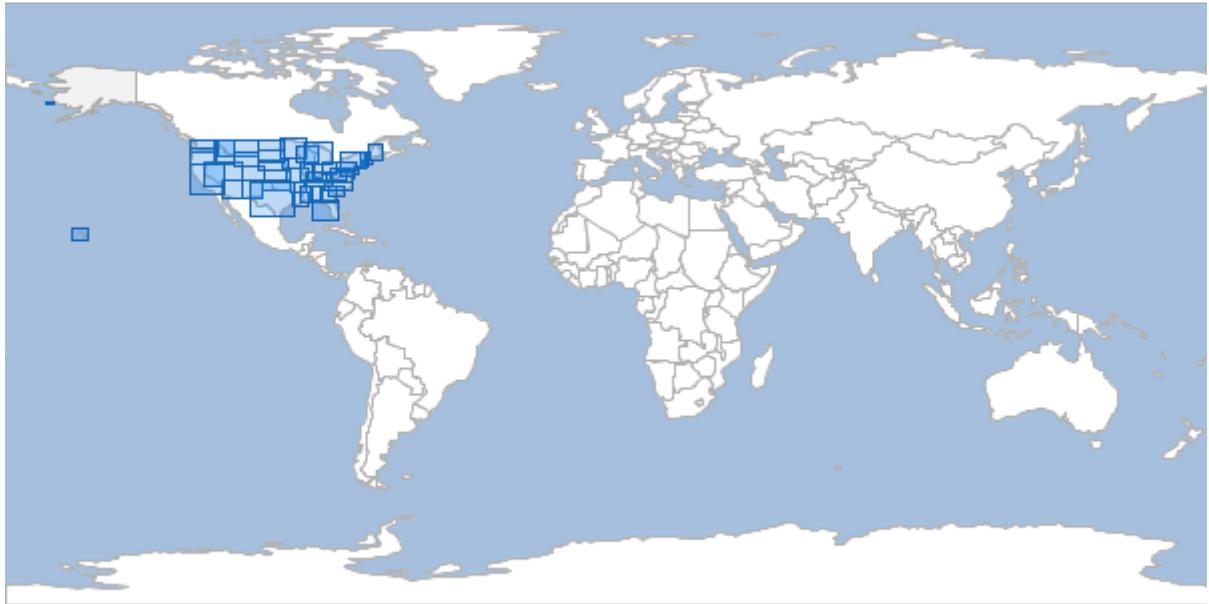
geo-shell> map add layer --name map --layer countries
Added countries layer to map map

geo-shell> map add layer --name map --layer states
Added states layer to map map

geo-shell> map add layer --name map --layer state_extents
Added state_extents layer to map map

geo-shell> map draw --name map --file examples/layer_extents.png
Done drawing /home/travis/build/jericks/geo-shell/examples/layer_extents.png!

geo-shell> map close --name map
Map map closed!
```



Graticule

Square

Create a square graticule.

```
geo-shell> layer graticule square --workspace layers --name squares --bounds -180,-90,180,90
--length 20
```

Name	Description	Mandatory	Specified Default	Unspecified Default
workspace	The Workspace name	true		
name	The new Layer name	true		
bounds	The bounds	true		
length	The length	true		
spacing	The spacing	false	-1	-1

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> layer graticule square --workspace layers --name squares --bounds -180,-90,180,90
--length 20
```

Created Square Graticule Layer squares!

```
geo-shell> style vector default --layer squares --color #1E90FF --opacity 0.30 --file
```

```
examples/squares.sld
Default Vector Style for squares written to /home/travis/build/jericks/geo-
shell/examples/squares.sld!
```

```
geo-shell> layer style set --name squares --style examples/squares.sld
Style /home/travis/build/jericks/geo-shell/examples/squares.sld set on squares
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name graticule
Map graticule opened!
```

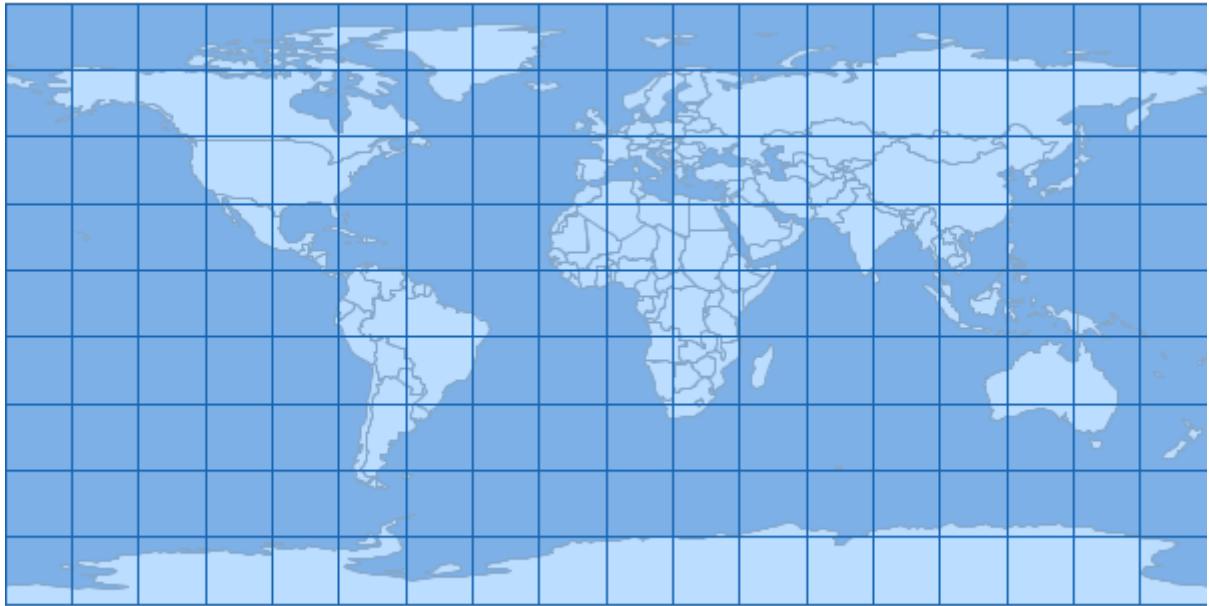
```
geo-shell> map add layer --name graticule --layer ocean
Added ocean layer to map graticule
```

```
geo-shell> map add layer --name graticule --layer countries
Added countries layer to map graticule
```

```
geo-shell> map add layer --name graticule --layer squares
Added squares layer to map graticule
```

```
geo-shell> map draw --name graticule --file examples/square_graticules.png
Done drawing /home/travis/build/jericks/geo-shell/examples/square_graticules.png!
```

```
geo-shell> map close --name graticule
Map graticule closed!
```



Rectangle

Create a rectangle graticule.

```
geo-shell> layer graticule rectangle --workspace layers --name rectangles --bounds -180,-90,180,90
--width 20 --height 10
```

Name	Description	Mandatory	Specified Default	Unspecified Default
workspace	The Workspace name	true		
name	The new Layer name	true		
bounds	The bounds	true		
width	The width	true		
height	The height	true		
spacing	The spacing	false	-1	-1

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> layer graticule rectangle --workspace layers --name rectangles --bounds -180,-90,180,90
--width 20 --height 10
```

Created Rectangle Graticule Layer rectangles!

```
geo-shell> style vector default --layer rectangles --color #1E90FF --opacity 0.30 --file
examples/rectangles.sld
```

Default Vector Style for rectangles written to /home/travis/build/jericks/geoshell/examples/rectangles.sld!

```
geo-shell> layer style set --name rectangles --style examples/rectangles.sld  
Style /home/travis/build/jericks/geoshell/examples/rectangles.sld set on rectangles
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth -layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geoshell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth -layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geoshell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name graticule  
Map graticule opened!
```

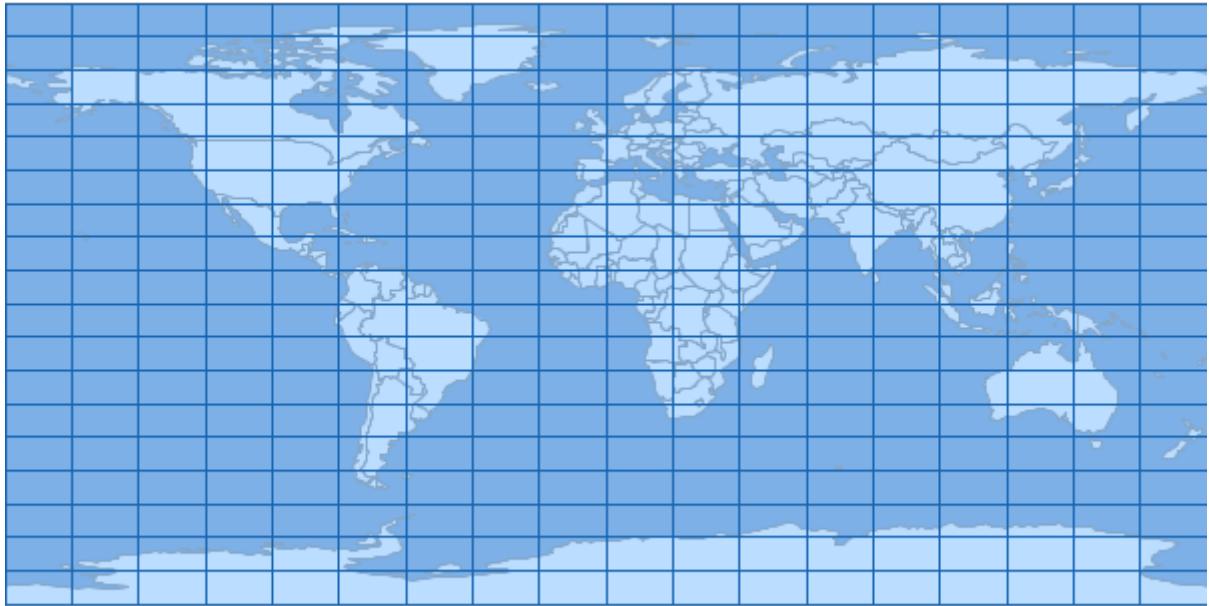
```
geo-shell> map add layer --name graticule --layer ocean  
Added ocean layer to map graticule
```

```
geo-shell> map add layer --name graticule --layer countries  
Added countries layer to map graticule
```

```
geo-shell> map add layer --name graticule --layer rectangles  
Added rectangles layer to map graticule
```

```
geo-shell> map draw --name graticule --file examples/rectangle_graticules.png  
Done drawing /home/travis/build/jericks/geoshell/examples/rectangle_graticules.png!
```

```
geo-shell> map close --name graticule  
Map graticule closed!
```



Oval

Create a oval graticule.

```
geo-shell> layer graticule oval --workspace layers --name ovals --bounds -180,-90,180,90 --size 20
```

Name	Description	Mandatory	Specified Default	Unspecified Default
workspace	The Workspace name	true		
name	The new Layer name	true		
bounds	The bounds	true		
size	The size	true		

```
geo-shell> workspace open --name layers --params memory
```

Workspace layers opened!

```
geo-shell> layer graticule oval --workspace layers --name ovals --bounds -180,-90,180,90 --size 20
```

Created Oval Graticule Layer ovals!

```
geo-shell> style vector default --layer ovals --color #1E90FF --opacity 0.30 --file examples/ovals.sld
```

Default Vector Style for ovals written to /home/travis/build/jericks/geo-shell/examples/ovals.sld!

```
geo-shell> layer style set --name ovals --style examples/ovals.sld
```

Style /home/travis/build/jericks/geo-shell/examples/ovals.sld set on ovals

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

Workspace naturalearth opened!

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name graticule  
Map graticule opened!
```

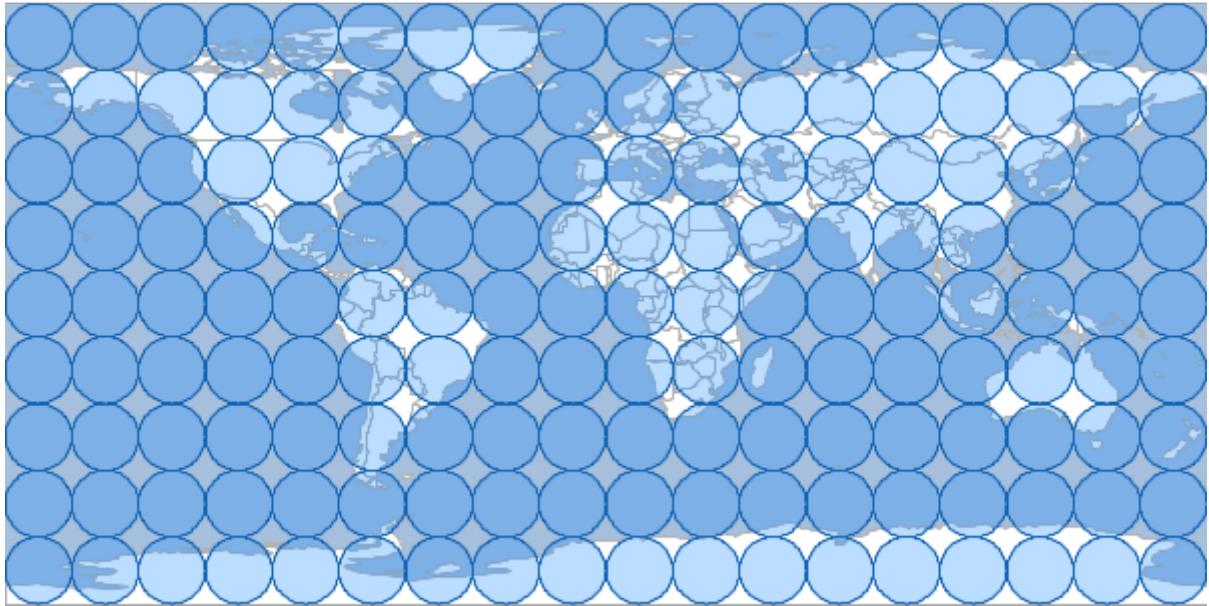
```
geo-shell> map add layer --name graticule --layer ocean  
Added ocean layer to map graticule
```

```
geo-shell> map add layer --name graticule --layer countries  
Added countries layer to map graticule
```

```
geo-shell> map add layer --name graticule --layer ovals  
Added ovals layer to map graticule
```

```
geo-shell> map draw --name graticule --file examples/oval_graticules.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/oval_graticules.png!
```

```
geo-shell> map close --name graticule  
Map graticule closed!
```



Hexagon

Create a hexagon graticule.

```
geo-shell> layer graticule hexagon --workspace layers --name hexagons --bounds -180,-90,180,90
--length 10
```

Name	Description	Mandatory	Specified Default	Unspecified Default
workspace	The Workspace name	true		
name	The new Layer name	true		
bounds	The bounds	true		
length	The length	true		
spacing	The spacing	false	5	5
orientation	The orientation (flat or angled)	false	flat	flat

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> layer graticule hexagon --workspace layers --name hexagons --bounds -180,-90,180,90
--length 10
```

Created Hexagon Graticule Layer hexagons!

```
geo-shell> style vector default --layer hexagons --color #1E90FF --opacity 0.30 --file
```

```
examples/hexagons.sld
Default Vector Style for hexagons written to /home/travis/build/jericks/geo-
shell/examples/hexagons.sld!
```

```
geo-shell> layer style set --name hexagons --style examples/hexagons.sld
Style /home/travis/build/jericks/geo-shell/examples/hexagons.sld set on hexagons
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name graticule
Map graticule opened!
```

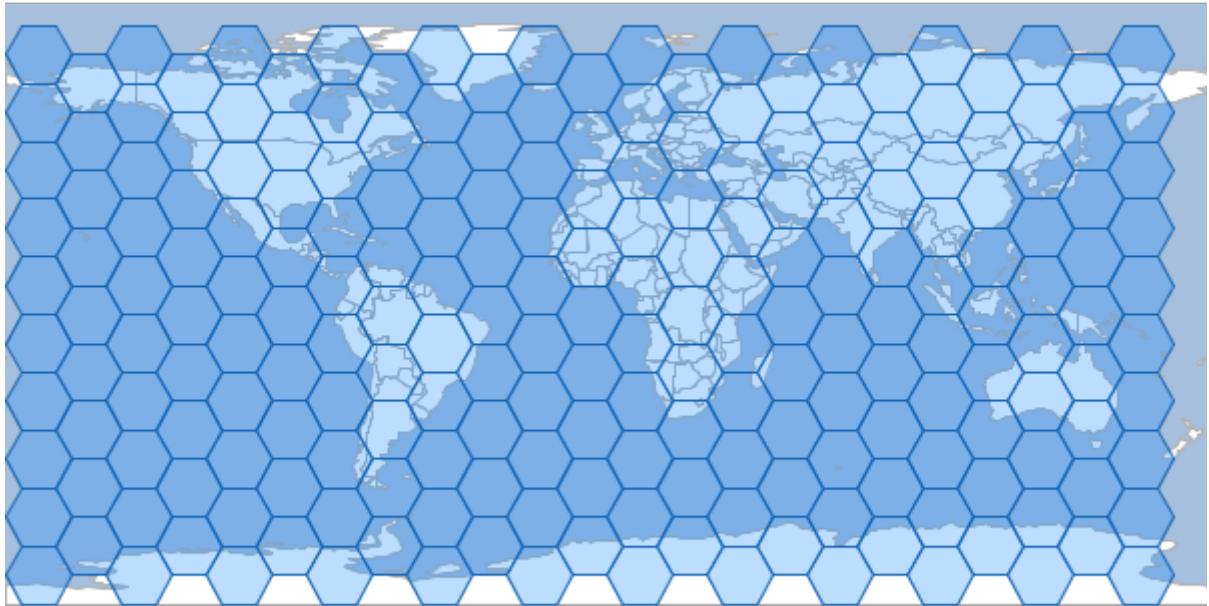
```
geo-shell> map add layer --name graticule --layer ocean
Added ocean layer to map graticule
```

```
geo-shell> map add layer --name graticule --layer countries
Added countries layer to map graticule
```

```
geo-shell> map add layer --name graticule --layer hexagons
Added hexagons layer to map graticule
```

```
geo-shell> map draw --name graticule --file examples/hexagon_graticules.png
Done drawing /home/travis/build/jericks/geo-shell/examples/hexagon_graticules.png!
```

```
geo-shell> map close --name graticule
Map graticule closed!
```



Format

Open

Open a Raster Format.

```
geo-shell> format open --name earth --input src/test/resources/earth.tif
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Format name	false		
input	The input string	true		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> format close --name earth
```

```
Format earth closed!
```

List

List open Raster Formats.

```
geo-shell> format list
```



No parameters

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> format open --name raster --input src/test/resources/raster.tif  
Format raster opened!
```

```
geo-shell> format list
```

```
earth = GeoTIFF
```

```
raster = GeoTIFF
```

```
geo-shell> format close --name earth
```

```
Format earth closed!
```

```
geo-shell> format close --name raster
```

```
Format raster closed!
```

Close

Close a Raster Format.

```
geo-shell> format close --name earth
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Format name	true		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> format close --name earth
```

```
Format earth closed!
```

Rasters

List the Rasters in a Format.

```
geo-shell> format rasters --name earth
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Format name	true		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> format rasters --name earth
```

```
earth
```

```
geo-shell> format close --name earth
```

Format earth closed!

Raster

Open

Open a Raster.

```
geo-shell> raster open --format earth --raster earth --name earth
```

Name	Description	Mandatory	Specified Default	Unspecified Default
format	The Format name	true		
raster	The Raster name	true		
name	The name	false		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif
```

Format earth opened!

```
geo-shell> raster open --format earth --raster earth --name earth
```

Opened Format earth Raster earth as earth

```
geo-shell> raster close --name earth
```

Raster earth closed!

```
geo-shell> format close --name earth
```

Format earth closed!

Close

Close a Raster.

```
geo-shell> raster close --name earth
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif
```

Format earth opened!

```
geo-shell> raster open --format earth --raster earth --name earth
```

Opened Format earth Raster earth as earth

```
geo-shell> raster close --name earth
```

Raster earth closed!

```
geo-shell> format close --name earth  
Format earth closed!
```

List

List open Rasters.

```
geo-shell> raster list
```



No parameters

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> raster open --format earth --raster earth --name earth  
Opened Format earth Raster earth as earth
```

```
geo-shell> raster list  
earth = GeoTIFF
```

```
geo-shell> raster close --name earth  
Raster earth closed!
```

```
geo-shell> format close --name earth  
Format earth closed!
```

Info

Get information about a Raster.

```
geo-shell> raster info --name earth
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> raster open --format earth --raster earth --name earth  
Opened Format earth Raster earth as earth
```

```
geo-shell> raster info --name earth  
Format: GeoTIFF  
Size: 800, 400  
Projection ID: EPSG:4326  
Projection WKT: GEOGCS["WGS 84",  
DATUM["World Geodetic System 1984",  
SPHEROID["WGS 84", 6378137.0, 298.257223563, AUTHORITY["EPSG", "7030"]],
```

```

AUTHORITY["EPSG","6326"],
PRIMEM["Greenwich", 0.0, AUTHORITY["EPSG","8901"]],
UNIT["degree", 0.017453292519943295],
AXIS["Geodetic longitude", EAST],
AXIS["Geodetic latitude", NORTH],
AUTHORITY["EPSG","4326"]

Extent: -179.9999999999997, -89.9999999998205, 179.99999999996405, 90.0
Pixel Size: 0.4499999999995505, 0.449999999999551
Block Size: 800, 8
Bands:
RED_BAND
Min Value: 56.0 Max Value: 255.0
GREEN_BAND
Min Value: 84.0 Max Value: 255.0
BLUE_BAND
Min Value: 91.0 Max Value: 255.0

```

```

geo-shell> raster close --name earth
Raster earth closed!

```

```

geo-shell> format close --name earth
Format earth closed!

```

Value

Get a value from the Raster.

```

geo-shell> raster value --name earth --x 60 --y 45

```

```

geo-shell> raster value --name earth --x 10 --y 15 --type pixel

```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
band	The x coordinate	false	0	0
x	The x coordinate	true		
y	The y coordinate	true		
type	The y coordinate	false	geometry	geometry

```

geo-shell> format open --name earth --input src/test/resources/earth.tif
Format earth opened!

```

```

geo-shell> raster open --format earth --raster earth --name earth
Opened Format earth Raster earth as earth

```

```

geo-shell> raster value --name earth --x 60 --y 45
235.0

```

```
geo-shell> raster value --name earth --x 10 --y 15 --type pixel  
109.0
```

```
geo-shell> raster close --name earth  
Raster earth closed!
```

```
geo-shell> format close --name earth  
Format earth closed!
```

Envelope

Create a Vector Layer from the envelope of a Raster.

```
geo-shell> raster envelope --name earth --output-workspace layers --output-name outline
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> raster open --format earth --raster earth --name earth  
Opened Format earth Raster earth as earth
```

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> raster envelope --name earth --output-workspace layers --output-name outline  
Done creating envelope in outline from earth!
```

```
geo-shell> style create --params "stroke=black stroke-width=3" --file examples/outline.sld  
Style      stroke=black      stroke-width=3      written      to      /home/travis/build/jericks/geo-  
shell/examples/outline.sld!
```

```
geo-shell> layer style set --name outline --style examples/outline.sld  
Style /home/travis/build/jericks/geo-shell/examples/outline.sld set on outline
```

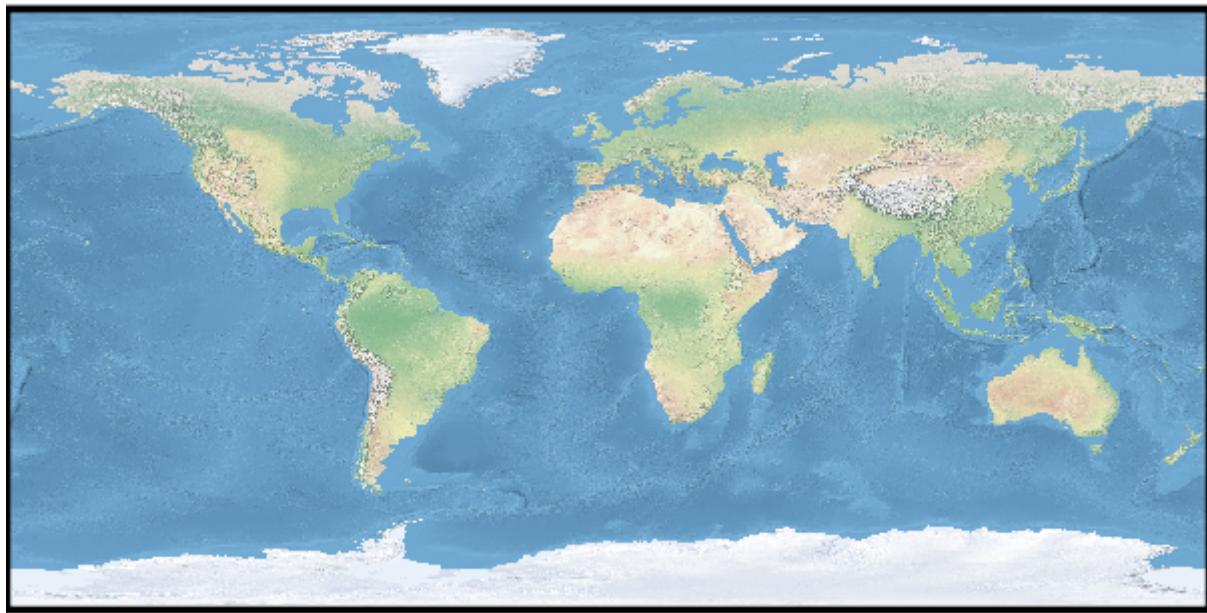
```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add raster --name map --raster earth  
Added earth layer to map map
```

```
geo-shell> map add layer --name map --layer outline  
Added outline layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_envelope.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_envelope.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Get Style

Get the Raster's style.

```
geo-shell> raster style get --name pc --style examples/pc_style.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
style	The SLD File	false		

```
geo-shell> format open --name pierce_county --input src/test/resources/pc.tif  
Format pierce_county opened!
```

```
geo-shell> raster open --format pierce_county --raster pc --name pc  
Opened Format pierce_county Raster pc as pc
```

```
geo-shell> style raster colormap --raster pc --values  
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5"  
--file examples/style_raster_colormap.sld  
Colormap Raster Style for pc written to /home/travis/build/jericks/geo-  
shell/examples/style_raster_colormap.sld!
```

```
geo-shell> raster style set --name pc --style examples/style_raster_colormap.sld  
Style /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.sld set on pc
```

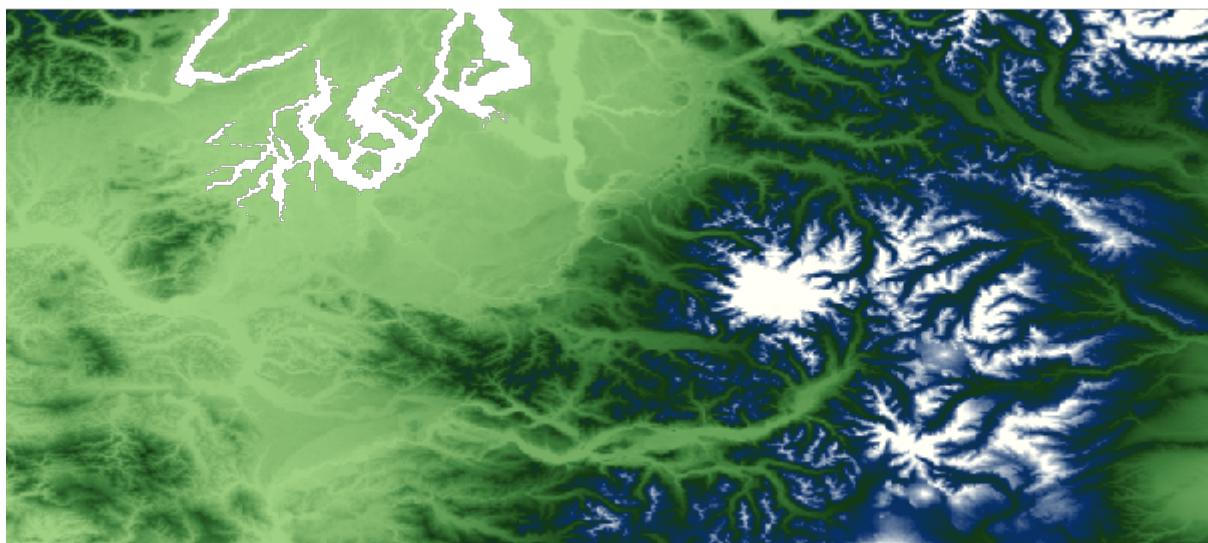
```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add raster --name map --raster pc  
Added pc layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_style_get.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_style_get.png!
```

```
geo-shell> map close --name map  
Map map closed!
```

```
geo-shell> raster style get --name pc --style examples/pc_style.sld  
pc style written to /home/travis/build/jericks/geo-shell/examples/pc_style.sld
```



Set Style

Set a Raster's style

```
geo-shell> raster style set --name pc --style examples/style_raster_colormap.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		

style	The SLD or CSS File	true		
-------	---------------------	------	--	--

geo-shell> **format open** --name pierce_county --input src/test/resources/pc.tif
Format pierce_county opened!

geo-shell> **raster open** --format pierce_county --raster pc --name pc
Opened Format pierce_county Raster pc as pc

geo-shell> **style raster colormap --raster pc --values -file**
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5"
examples/style_raster_colormap.sld
Colormap Raster Style for pc written to /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.sld!

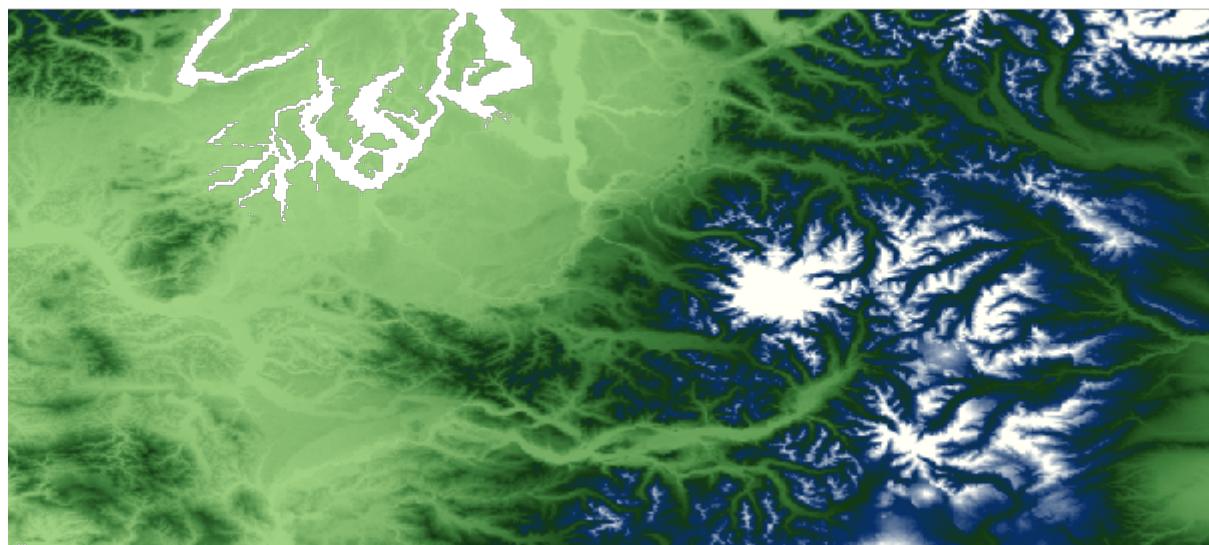
geo-shell> **raster style set** --name pc --style examples/style_raster_colormap.sld
Style /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.sld set on pc

geo-shell> **map open** --name map
Map map opened!

geo-shell> **map add raster** --name map --raster pc
Added pc layer to map map

geo-shell> **map draw** --name map --file examples/raster_style_set.png
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_style_set.png!

geo-shell> **map close** --name map
Map map closed!



Add Raster

Add two Rasters together

```
geo-shell> raster add raster --name1 high --name2 low --output-format add --output-name add
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name1	The Raster name	true		
name2	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		

```
geo-shell> format open --name high --input src/test/resources/high.tif  
Format high opened!
```

```
geo-shell> raster open --format high --raster high --name high  
Opened Format high Raster high as high
```

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> style create --params "stroke=black stroke-width=2 label=value label-size=12" --file  
examples/grid.sld  
Style stroke=black stroke-width=2 label=value label-size=12 written to  
/home/travis/build/jericks/geo-shell/examples/grid.sld!
```

```
geo-shell> raster polygon --name high --output-workspace layers --output-name high_polygons  
Done converting Raster high to a Polygon Layer high_polygons!
```

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file  
examples/high.sld  
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/high.sld!
```

```
geo-shell> raster style set --name high --style examples/high.sld  
Style /home/travis/build/jericks/geo-shell/examples/high.sld set on high
```

```
geo-shell> layer style set --name high_polygons --style examples/grid.sld  
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on high_polygons
```

```
geo-shell> map open --name mapHigh  
Map mapHigh opened!
```

```
geo-shell> map add raster --name mapHigh --raster high  
Added high layer to map mapHigh
```

```
geo-shell> map add layer --name mapHigh --layer high_polygons
```

Added high_polygons layer to map mapHigh

```
geo-shell> map draw --name mapHigh --file examples/raster_add_raster_high.png --bounds "-180,-90,180,90,EPGS:4326"
```

Done drawing /home/travis/build/jericks/geo-shell/examples/raster_add_raster_high.png!

```
geo-shell> map close --name mapHigh
```

Map mapHigh closed!

17.0	18.0	19.0	20.0
13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0

```
geo-shell> format open --name low --input src/test/resources/low.tif
```

Format low opened!

```
geo-shell> raster open --format low --raster low --name low
```

Opened Format low Raster low as low

```
geo-shell> raster polygon --name low --output-workspace layers --output-name low_polygons
```

Done converting Raster low to a Polygon Layer low_polygons!

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file examples/low.sld
```

Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/low.sld!

```
geo-shell> raster style set --name low --style examples/low.sld
```

Style /home/travis/build/jericks/geo-shell/examples/low.sld set on low

```
geo-shell> layer style set --name low_polygons --style examples/grid.sld
```

Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on low_polygons

```
geo-shell> map open --name mapLow
```

Map mapLow opened!

```
geo-shell> map add raster --name mapLow --raster low
```

```
Added low layer to map mapLow
```

```
geo-shell> map add layer --name mapLow --layer low_polygons
```

```
Added low_polygons layer to map mapLow
```

```
geo-shell> map draw --name mapLow --file examples/raster_add_raster_low.png --bounds "-180,-90,180,90,EPGS:4326"
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_add_raster_low.png!
```

```
geo-shell> map close --name mapLow
```

```
Map mapLow closed!
```

13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0
1.0	2.0	3.0	4.0

```
geo-shell> format open --name add --input examples/add.tif
```

```
Format add opened!
```

```
geo-shell> raster add raster --name1 high --name2 low --output-format add --output-name add
```

```
Added high to low to create add!
```

```
geo-shell> raster polygon --name add --output-workspace layers --output-name add_polygons
```

```
Done converting Raster add to a Polygon Layer add_polygons!
```

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file examples/add.sld
```

```
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/add.sld!
```

```
geo-shell> raster style set --name add --style examples/add.sld
```

```
Style /home/travis/build/jericks/geo-shell/examples/add.sld set on add
```

```
geo-shell> layer style set --name add_polygons --style examples/grid.sld
```

Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on add_polygons

```
geo-shell> map open --name mapAdd
```

Map mapAdd opened!

```
geo-shell> map add raster --name mapAdd --raster add
```

Added add layer to map mapAdd

```
geo-shell> map add layer --name mapAdd --layer add_polygons
```

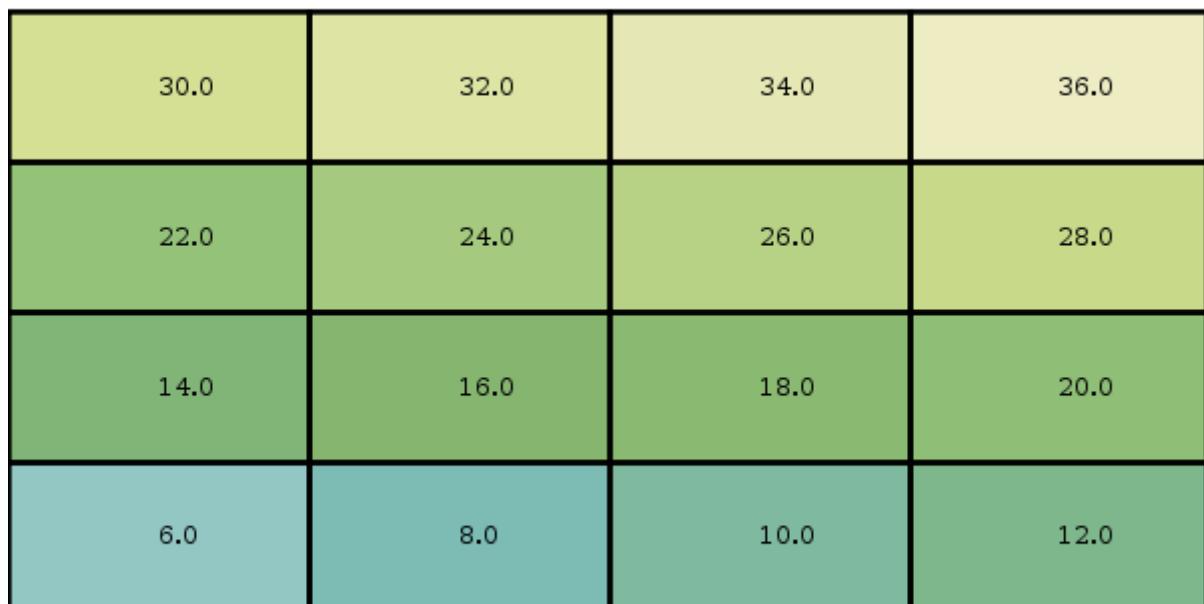
Added add_polygons layer to map mapAdd

```
geo-shell> map draw --name mapAdd --file examples/raster_add_raster_add.png --bounds "-180,-90,180,90,EPNG:4326"
```

Done drawing /home/travis/build/jericks/geo-shell/examples/raster_add_raster_add.png!

```
geo-shell> map close --name mapAdd
```

Map mapAdd closed!



Add Constant

Add constant values to a Raster

```
geo-shell> raster add constant --name pc --output-format pcAdd100 --output-name pcAdd100  
--values 100
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		

output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
values	The values	true		

```
geo-shell> format open --name pierce_county --input src/test/resources/pc.tif
Format pierce_county opened!
```

```
geo-shell> raster open --format pierce_county --raster pc --name pc
Opened Format pierce_county Raster pc as pc
```

```
geo-shell> raster value --name pc --x -121.799927 --y 46.867703
3069.0
```

```
geo-shell> format open --name pcAdd100 --input examples/pcAdd100.tif
Format pcAdd100 opened!
```

```
geo-shell> raster add constant --name pc --output-format pcAdd100 --output-name pcAdd100
--values 100
Added 100 to pc to create pcAdd100!
```

```
geo-shell> raster value --name pcAdd100 --x -121.799927 --y 46.867703
3169.0
```

```
geo-shell>      style      raster      colormap      --raster      pcAdd100      --values
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5"
--file
examples/style_raster_colormap.sld
Colormap      Raster      Style      for      pcAdd100      written      to      /home/travis/build/jericks/geo-
shell/examples/style_raster_colormap.sld!
```

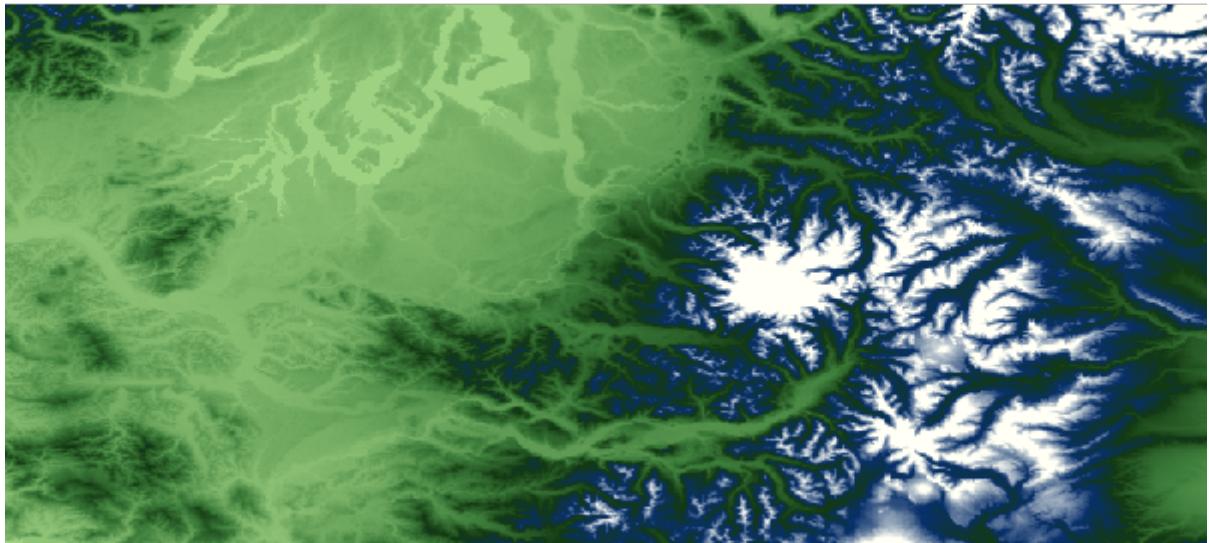
```
geo-shell> raster style set --name pcAdd100 --style examples/style_raster_colormap.sld
Style /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.sld set on pcAdd100
```

```
geo-shell> map open --name map
Map map opened!
```

```
geo-shell> map add raster --name map --raster pcAdd100
Added pcAdd100 layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_add_constant.png
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_add_constant.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Subtract Raster

Subtract one Raster from another

```
geo-shell> raster subtract raster --name1 high --name2 low --output-format subtract --output-name subtract
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name1	The Raster name	true		
name2	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		

```
geo-shell> format open --name high --input src/test/resources/high.tif  
Format high opened!
```

```
geo-shell> raster open --format high --raster high --name high  
Opened Format high Raster high as high
```

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> style create --params "stroke=black stroke-width=2 label=value label-size=12" --file examples/grid.sld  
Style      stroke=black      stroke-width=2      label=value      label-size=12      written      to
```

```
/home/travis/build/jericks/geo-shell/examples/grid.sld!
```

```
geo-shell> raster polygon --name high --output-workspace layers --output-name high_polygons  
Done converting Raster high to a Polygon Layer high_polygons!
```

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file  
examples/high.sld
```

```
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/high.sld!
```

```
geo-shell> raster style set --name high --style examples/high.sld  
Style /home/travis/build/jericks/geo-shell/examples/high.sld set on high
```

```
geo-shell> layer style set --name high_polygons --style examples/grid.sld  
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on high_polygons
```

```
geo-shell> map open --name mapHigh  
Map mapHigh opened!
```

```
geo-shell> map add raster --name mapHigh --raster high  
Added high layer to map mapHigh
```

```
geo-shell> map add layer --name mapHigh --layer high_polygons  
Added high_polygons layer to map mapHigh
```

```
geo-shell> map draw --name mapHigh --file examples/raster_subtract_raster_high.png --bounds "-  
180,-90,180,90,EPSC:4326"  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_subtract_raster_high.png!
```

```
geo-shell> map close --name mapHigh  
Map mapHigh closed!
```

17.0	18.0	19.0	20.0
13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0

```
geo-shell> format open --name low --input src/test/resources/low.tif
Format low opened!

geo-shell> raster open --format low --raster low --name low
Opened Format low Raster low as low

geo-shell> raster polygon --name low --output-workspace layers --output-name low_polygons
Done converting Raster low to a Polygon Layer low_polygons!

geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file
examples/low.sld
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/low.sld!

geo-shell> raster style set --name low --style examples/low.sld
Style /home/travis/build/jericks/geo-shell/examples/low.sld set on low

geo-shell> layer style set --name low_polygons --style examples/grid.sld
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on low_polygons

geo-shell> map open --name mapLow
Map mapLow opened!

geo-shell> map add raster --name mapLow --raster low
Added low layer to map mapLow

geo-shell> map add layer --name mapLow --layer low_polygons
Added low_polygons layer to map mapLow

geo-shell> map draw --name mapLow --file examples/raster_subtract_raster_low.png --bounds "
-180,-90,180,90,EPSC:4326"
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_subtract_raster_low.png!

geo-shell> map close --name mapLow
Map mapLow closed!
```

13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0
1.0	2.0	3.0	4.0

```
geo-shell> format open --name subtract --input examples/subtract.tif
Format subtract opened!
```

```
geo-shell> raster subtract raster --name1 high --name2 low --output-format subtract --output-name
subtract
Subtracted high from low to create subtract!
```

```
geo-shell> raster polygon --name subtract --output-workspace layers --output-name
subtract_polygons
Done converting Raster subtract to a Polygon Layer subtract_polygons!
```

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file
examples/subtract.sld
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/subtract.sld!
```

```
geo-shell> raster style set --name subtract --style examples/subtract.sld
Style /home/travis/build/jericks/geo-shell/examples/subtract.sld set on subtract
```

```
geo-shell> layer style set --name subtract_polygons --style examples/grid.sld
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on subtract_polygons
```

```
geo-shell> map open --name mapSubtract
Map mapSubtract opened!
```

```
geo-shell> map add raster --name mapSubtract --raster subtract
Added subtract layer to map mapSubtract
```

```
geo-shell> map add layer --name mapSubtract --layer subtract_polygons
Added subtract_polygons layer to map mapSubtract
```

```
geo-shell> map draw --name mapSubtract --file examples/raster_subtract_raster_subtract.png  
--bounds "-180,-90,180,90,EPSC:4326"
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_subtract_raster_subtract.png!
```

```
geo-shell> map close --name mapSubtract
```

```
Map mapSubtract closed!
```



Subtract Constant

Subtract constant values from a Raster

```
geo-shell> raster subtract constant --name pc --output-format pcMinus100 --output-name  
pcMinus100 --values 100
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
values	The values	true		
from	Whether to subtract the Raster from the constant or vice versa	false	false	false

```
geo-shell> format open --name pierce_county --input src/test/resources/pc.tif
Format pierce_county opened!
```

```
geo-shell> raster open --format pierce_county --raster pc --name pc
Opened Format pierce_county Raster pc as pc
```

```
geo-shell> raster value --name pc --x -121.799927 --y 46.867703
3069.0
```

```
geo-shell> format open --name pcMinus100 --input examples/pcMinus100.tif
Format pcMinus100 opened!
```

```
geo-shell> raster subtract constant --name pc --output-format pcMinus100 --output-name
pcMinus100 --values 100
Subtracted 100 from pc to create pcMinus100!
```

```
geo-shell> raster value --name pcMinus100 --x -121.799927 --y 46.867703
2969.0
```

```
geo-shell> style raster colormap --raster pcMinus100 --values
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5"
-examples/style_raster_colormap.sld
--file
Colormap Raster Style for pcMinus100 written to /home/travis/build/jericks/geo-
shell/examples/style_raster_colormap.sld!
```

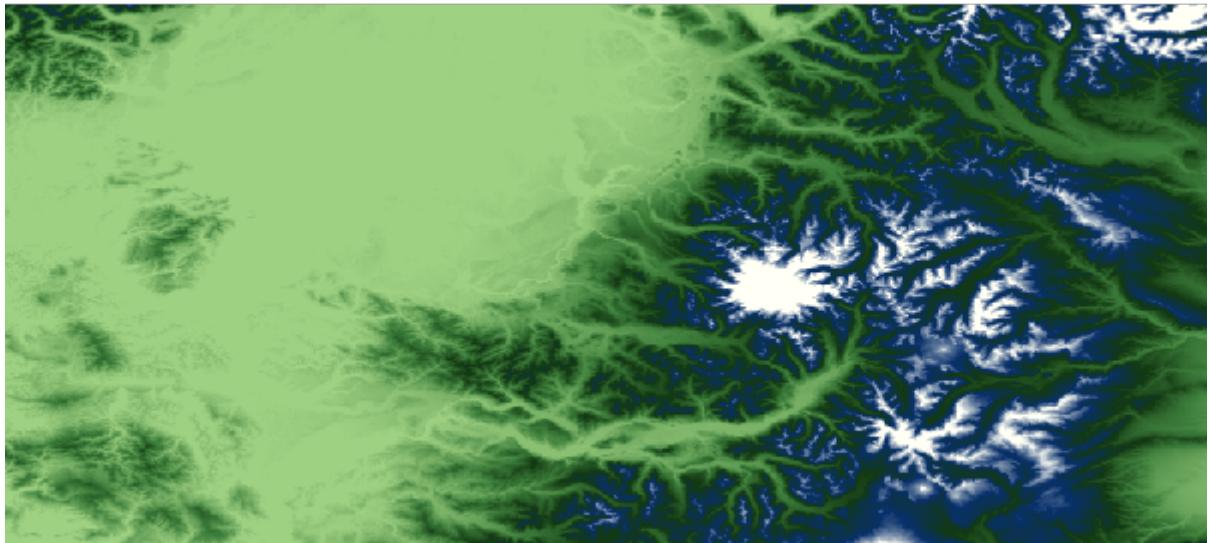
```
geo-shell> raster style set --name pcMinus100 --style examples/style_raster_colormap.sld
Style /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.sld set on pcMinus100
```

```
geo-shell> map open --name map
Map map opened!
```

```
geo-shell> map add raster --name map --raster pcMinus100
Added pcMinus100 layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_subtract_constant.png
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_subtract_constant.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Multiply Raster

Multiply two Raster together

```
geo-shell> raster multiply raster --name1 high --name2 low --output-format multiply --output-name multiply
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name1	The Raster name	true		
name2	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		

```
geo-shell> format open --name high --input src/test/resources/high.tif  
Format high opened!
```

```
geo-shell> raster open --format high --raster high --name high  
Opened Format high Raster high as high
```

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> style create --params "stroke=black stroke-width=2 label=value label-size=12" --file examples/grid.sld  
Style      stroke=black      stroke-width=2      label=value      label-size=12      written      to
```

```
/home/travis/build/jericks/geo-shell/examples/grid.sld!
```

```
geo-shell> raster polygon --name high --output-workspace layers --output-name high_polygons  
Done converting Raster high to a Polygon Layer high_polygons!
```

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file  
examples/high.sld
```

```
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/high.sld!
```

```
geo-shell> raster style set --name high --style examples/high.sld  
Style /home/travis/build/jericks/geo-shell/examples/high.sld set on high
```

```
geo-shell> layer style set --name high_polygons --style examples/grid.sld  
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on high_polygons
```

```
geo-shell> map open --name mapHigh  
Map mapHigh opened!
```

```
geo-shell> map add raster --name mapHigh --raster high  
Added high layer to map mapHigh
```

```
geo-shell> map add layer --name mapHigh --layer high_polygons  
Added high_polygons layer to map mapHigh
```

```
geo-shell> map draw --name mapHigh --file examples/raster_multiply_raster_high.png --bounds "-  
180,-90,180,90,EPSC:4326"  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_multiply_raster_high.png!
```

```
geo-shell> map close --name mapHigh  
Map mapHigh closed!
```

17.0	18.0	19.0	20.0
13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0

```
geo-shell> format open --name low --input src/test/resources/low.tif
Format low opened!

geo-shell> raster open --format low --raster low --name low
Opened Format low Raster low as low

geo-shell> raster polygon --name low --output-workspace layers --output-name low_polygons
Done converting Raster low to a Polygon Layer low_polygons!

geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file
examples/low.sld
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/low.sld!

geo-shell> raster style set --name low --style examples/low.sld
Style /home/travis/build/jericks/geo-shell/examples/low.sld set on low

geo-shell> layer style set --name low_polygons --style examples/grid.sld
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on low_polygons

geo-shell> map open --name mapLow
Map mapLow opened!

geo-shell> map add raster --name mapLow --raster low
Added low layer to map mapLow

geo-shell> map add layer --name mapLow --layer low_polygons
Added low_polygons layer to map mapLow

geo-shell> map draw --name mapLow --file examples/raster_multiply_raster_low.png --bounds "
-180,-90,180,90,EPSC:4326"
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_multiply_raster_low.png!

geo-shell> map close --name mapLow
Map mapLow closed!
```

13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0
1.0	2.0	3.0	4.0

```
geo-shell> format open --name multiply --input examples/multiply.tif
Format multiply opened!
```

```
geo-shell> raster multiply raster --name1 high --name2 low --output-format multiply --output
-name multiply
Multiplied high and low to create multiply!
```

```
geo-shell> raster polygon --name multiply --output-workspace layers --output-name
multiply_polygons
Done converting Raster multiply to a Polygon Layer multiply_polygons!
```

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file
examples/multiply.sld
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/multiply.sld!
```

```
geo-shell> raster style set --name multiply --style examples/multiply.sld
Style /home/travis/build/jericks/geo-shell/examples/multiply.sld set on multiply
```

```
geo-shell> layer style set --name multiply_polygons --style examples/grid.sld
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on multiply_polygons
```

```
geo-shell> map open --name mapSubtract
Map mapSubtract opened!
```

```
geo-shell> map add raster --name mapSubtract --raster multiply
Added multiply layer to map mapSubtract
```

```
geo-shell> map add layer --name mapSubtract --layer multiply_polygons
Added multiply_polygons layer to map mapSubtract
```

```
geo-shell> map draw --name mapSubtract --file examples/raster_multiply_raster_multiply.png  
--bounds "-180,-90,180,90,EPSC:4326"
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_multiply_raster_multiply.png!
```

```
geo-shell> map close --name mapSubtract
```

```
Map mapSubtract closed!
```

221.0	252.0	285.0	320.0
117.0	140.0	165.0	192.0
45.0	60.0	77.0	96.0
5.0	12.0	21.0	32.0

Multiply Constant

Multiply constant values to a Raster

```
geo-shell> raster multiply constant --name pc --output-format pcTimes2 --output-name pcTimes2  
--values 2
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
values	The values	true		

```
geo-shell> format open --name pierce_county --input src/test/resources/pc.tif  
Format pierce_county opened!
```

```
geo-shell> raster open --format pierce_county --raster pc --name pc  
Opened Format pierce_county Raster pc as pc
```

```
geo-shell> raster value --name pc --x -121.799927 --y 46.867703  
3069.0

geo-shell> format open --name pcTimes2 --input examples/pcTimes2.tif  
Format pcTimes2 opened!

geo-shell> raster multiply constant --name pc --output-format pcTimes2 --output-name pcTimes2  
--values 2  
Multiplied pc by 2 to create pcTimes2!

geo-shell> raster value --name pcTimes2 --x -121.799927 --y 46.867703  
6138.0

geo-shell> style style raster colormap --raster pcTimes2 --values  
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5"  
--file examples/style_raster_colormap.sld  
Colormap Raster Style for pcTimes2 written to /home/travis/build/jericks/geo-  
shell/examples/style_raster_colormap.sld!

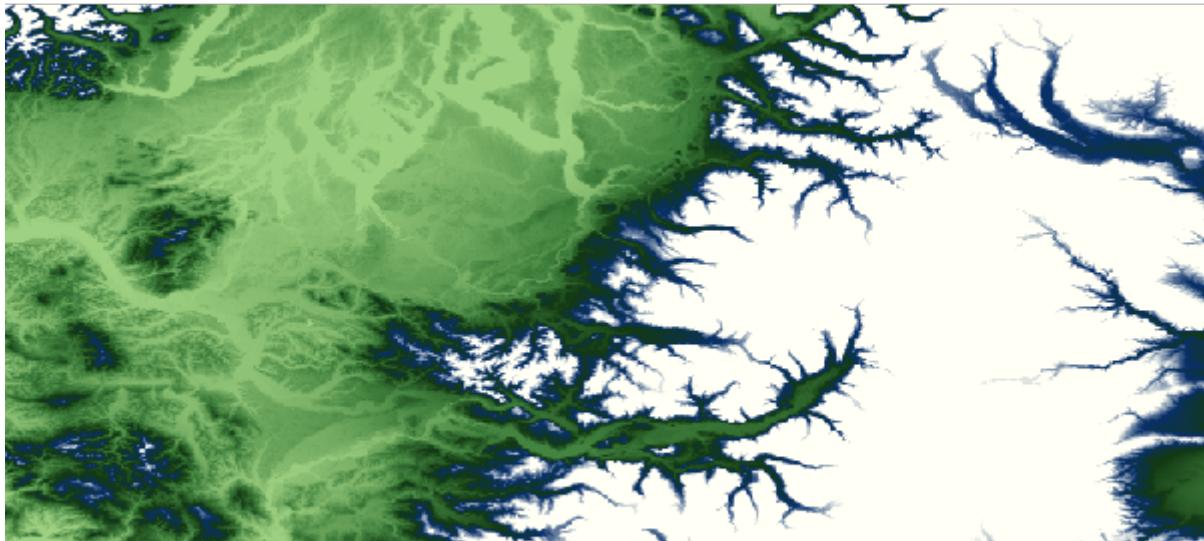
geo-shell> raster style set --name pcTimes2 --style examples/style_raster_colormap.sld  
Style /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.sld set on pcTimes2

geo-shell> map open --name map  
Map map opened!

geo-shell> map add raster --name map --raster pcTimes2  
Added pcTimes2 layer to map map

geo-shell> map draw --name map --file examples/raster_multiply_constant.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_multiply_constant.png!

geo-shell> map close --name map  
Map map closed!
```



Divide Raster

Divide one Raster by another Raster

```
geo-shell> raster divide raster --name1 high --name2 low --output-format divide --output-name divide
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name1	The Raster name	true		
name2	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		

```
geo-shell> format open --name high --input src/test/resources/high.tif  
Format high opened!
```

```
geo-shell> raster open --format high --raster high --name high  
Opened Format high Raster high as high
```

```
geo-shell> workspace open --name layers --params memory  
Workspace layers opened!
```

```
geo-shell> style create --params "stroke=black stroke-width=2 label=value label-size=12" --file examples/grid.sld  
Style      stroke=black      stroke-width=2      label=value      label-size=12      written      to
```

```
/home/travis/build/jericks/geo-shell/examples/grid.sld!
```

```
geo-shell> raster polygon --name high --output-workspace layers --output-name high_polygons  
Done converting Raster high to a Polygon Layer high_polygons!
```

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file  
examples/high.sld
```

```
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/high.sld!
```

```
geo-shell> raster style set --name high --style examples/high.sld  
Style /home/travis/build/jericks/geo-shell/examples/high.sld set on high
```

```
geo-shell> layer style set --name high_polygons --style examples/grid.sld  
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on high_polygons
```

```
geo-shell> map open --name mapHigh  
Map mapHigh opened!
```

```
geo-shell> map add raster --name mapHigh --raster high  
Added high layer to map mapHigh
```

```
geo-shell> map add layer --name mapHigh --layer high_polygons  
Added high_polygons layer to map mapHigh
```

```
geo-shell> map draw --name mapHigh --file examples/raster_divide_raster_high.png --bounds "-  
180,-90,180,90,EPSC:4326"  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_divide_raster_high.png!
```

```
geo-shell> map close --name mapHigh  
Map mapHigh closed!
```

17.0	18.0	19.0	20.0
13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0

```
geo-shell> format open --name low --input src/test/resources/low.tif
Format low opened!

geo-shell> raster open --format low --raster low --name low
Opened Format low Raster low as low

geo-shell> raster polygon --name low --output-workspace layers --output-name low_polygons
Done converting Raster low to a Polygon Layer low_polygons!

geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file
examples/low.sld
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/low.sld!

geo-shell> raster style set --name low --style examples/low.sld
Style /home/travis/build/jericks/geo-shell/examples/low.sld set on low

geo-shell> layer style set --name low_polygons --style examples/grid.sld
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on low_polygons

geo-shell> map open --name mapLow
Map mapLow opened!

geo-shell> map add raster --name mapLow --raster low
Added low layer to map mapLow

geo-shell> map add layer --name mapLow --layer low_polygons
Added low_polygons layer to map mapLow

geo-shell> map draw --name mapLow --file examples/raster_divide_raster_low.png --bounds "-180,-
90,180,90,EPSC:4326"
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_divide_raster_low.png!

geo-shell> map close --name mapLow
Map mapLow closed!
```

13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0
1.0	2.0	3.0	4.0

```
geo-shell> format open --name divide --input examples/divide.tif
Format divide opened!
```

```
geo-shell> raster divide raster --name1 high --name2 low --output-format divide --output-name divide
Divided high by low to create divide!
```

```
geo-shell> raster polygon --name divide --output-workspace layers --output-name divide_polygons
Done converting Raster divide to a Polygon Layer divide_polygons!
```

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file examples/divide.sld
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/divide.sld!
```

```
geo-shell> raster style set --name divide --style examples/divide.sld
Style /home/travis/build/jericks/geo-shell/examples/divide.sld set on divide
```

```
geo-shell> layer style set --name divide_polygons --style examples/grid.sld
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on divide_polygons
```

```
geo-shell> map open --name mapSubtract
Map mapSubtract opened!
```

```
geo-shell> map add raster --name mapSubtract --raster divide
Added divide layer to map mapSubtract
```

```
geo-shell> map add layer --name mapSubtract --layer divide_polygons
Added divide_polygons layer to map mapSubtract
```

```
geo-shell> map draw --name mapSubtract --file examples/raster_divide_raster_divide.png --bounds
```

"-180,-90,180,90,EPGS:4326"

Done drawing /home/travis/build/jericks/geo-shell/examples/raster_divide_raster_divide.png!

geo-shell> **map close** --name mapSubtract

Map mapSubtract closed!

1.307692289352417	1.2857142686843872	1.2666666507720947	1.25
1.4444444179534912	1.399999976158142	1.3636363744735718	
1.7999999523162842	1.6666666269302368	1.5714285373687744	1.5
5.0	3.0	2.3333332538604736	2.0

Divide Constant

Divide constant values against a Raster

geo-shell> **raster divide constant** --name pc --output-format pcDividedBy2 --output-name pcDividedBy2 --values 2

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
values	The values	true		

geo-shell> **format open** --name pierce_county --input src/test/resources/pc.tif

Format pierce_county opened!

geo-shell> **raster open** --format pierce_county --raster pc --name pc

Opened Format pierce_county Raster pc as pc

```
geo-shell> raster value --name pc --x -121.799927 --y 46.867703  
3069.0

geo-shell> format open --name pcDividedBy2 --input examples/pcDividedBy2.tif  
Format pcDividedBy2 opened!

geo-shell> raster divide constant --name pc --output-format pcDividedBy2 --output-name  
pcDividedBy2 --values 2  
Divided pc by 2 to create pcDividedBy2!

geo-shell> raster value --name pcDividedBy2 --x -121.799927 --y 46.867703  
1534.5

geo-shell> style raster colormap --raster pcDividedBy2 --values  
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5" --file  
examples/style_raster_colormap.sld  
Colormap Raster Style for pcDividedBy2 written to /home/travis/build/jericks/geo-  
shell/examples/style_raster_colormap.sld!

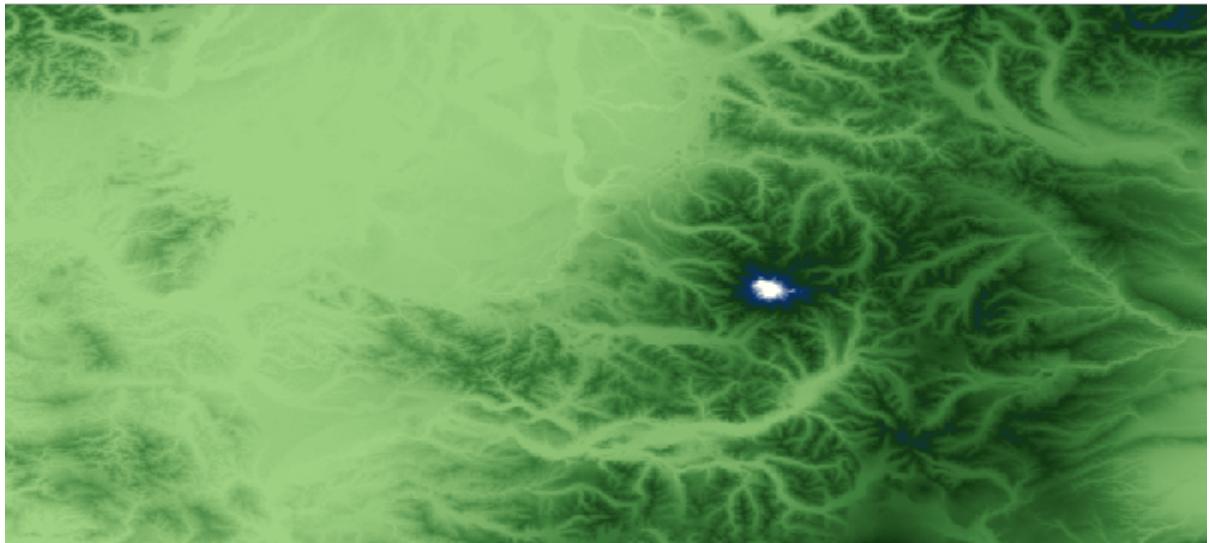
geo-shell> raster style set --name pcDividedBy2 --style examples/style_raster_colormap.sld  
Style /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.sld set on pcDividedBy2

geo-shell> map open --name map  
Map map opened!

geo-shell> map add raster --name map --raster pcDividedBy2  
Added pcDividedBy2 layer to map map

geo-shell> map draw --name map --file examples/raster_divide_constant.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_divide_constant.png!

geo-shell> map close --name map  
Map map closed!
```



Contours

Create contours.

```
geo-shell> raster contours --name pc --output-workspace contours --output-name contours --levels 0,100,200,300,600,900
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
band	The Raster band to contour	false	0	0
levels	The contour level or interval	true		
simplify	Whether to simplify	false	false	false
smooth	Whether to smooth	false	false	false
bounds	The Bounds	false		

```
geo-shell> format open --name pc --input src/test/resources/pc.tif  
Format pc opened!
```

```
geo-shell> raster open --format pc --raster pc --name pc
Opened Format pc Raster pc as pc

geo-shell>      style      raster      colormap      --raster      pc      --values
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5" --file examples/pc.sld
Colormap Raster Style for pc written to /home/travis/build/jericks/geo-shell/examples/pc.sld!

geo-shell> raster style set --name pc --style examples/pc.sld
Style /home/travis/build/jericks/geo-shell/examples/pc.sld set on pc

geo-shell> workspace open --name contours --params examples/contours.shp
Workspace contours opened!

geo-shell> raster contours --name pc --output-workspace contours --output-name contours --levels
0,100,200,300,600,900
Done creating contours!

geo-shell> style create --params "stroke=black stroke-width=0.25" --file examples/contours.sld
Style    stroke=black    stroke-width=0.25    written    to    /home/travis/build/jericks/geo-
shell/examples/contours.sld!

geo-shell> layer style set --name contours --style examples/contours.sld
Style /home/travis/build/jericks/geo-shell/examples/contours.sld set on contours

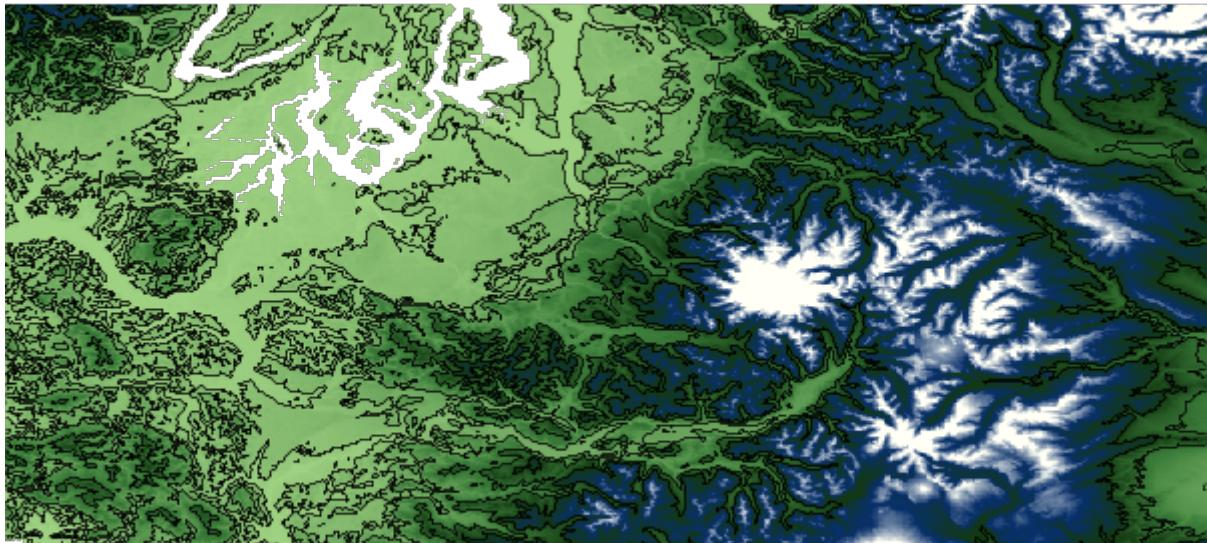
geo-shell> map open --name map
Map map opened!

geo-shell> map add raster --name map --raster pc
Added pc layer to map map

geo-shell> map add layer --name map --layer contours
Added contours layer to map map

geo-shell> map draw --name map --file examples/raster_contours.png
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_contours.png!

geo-shell> map close --name map
Map map closed!
```



Crop

Crop a Raster.

```
geo-shell> raster crop --name earth --output-format earthCropped --output-name earthCropped  
--geometry "-160.927734,6.751896,-34.716797,57.279043"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
geometry	The geometry	true		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> raster open --format earth --raster earth --name earth  
Opened Format earth Raster earth as earth
```

```
geo-shell> format open --name earthCropped --input examples/earthCropped.tif  
Format earthCropped opened!
```

```
geo-shell> raster crop --name earth --output-format earthCropped --output-name earthCropped  
--geometry "-160.927734,6.751896,-34.716797,57.279043"  
Raster earth cropped to earthCropped!
```

```
geo-shell> map open --name map
```

```
Map map opened!
```

```
geo-shell> map add raster --name map --raster earthCropped
```

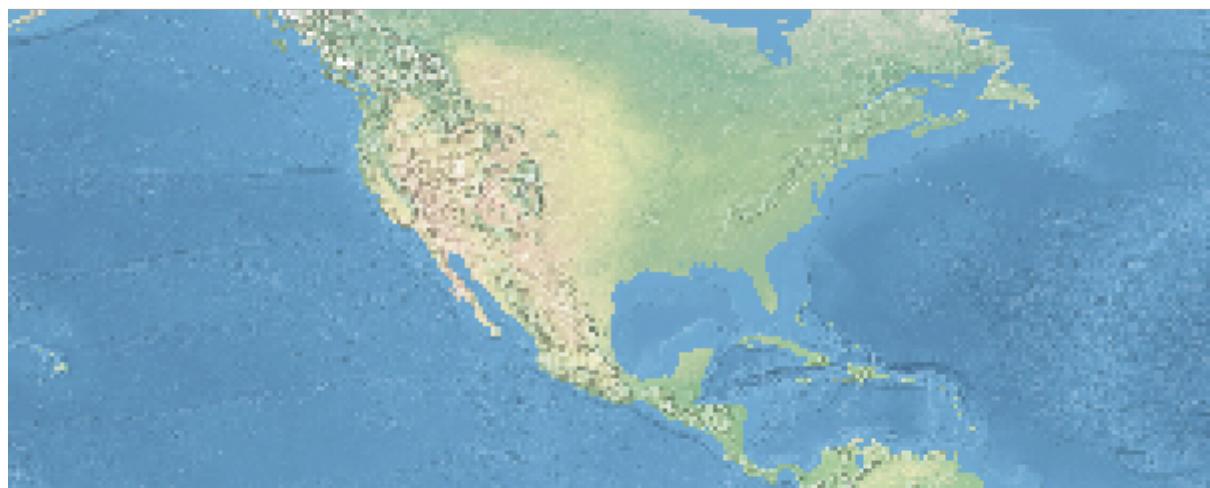
```
Added earthCropped layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_crop.png
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_crop.png!
```

```
geo-shell> map close --name map
```

```
Map map closed!
```



Mosaic

Mosaic two Rasters together

```
geo-shell> raster mosaic --name1 alki2 --name2 alki3 --output-format mosaic --output-name mosaic
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name1	The Raster name	true		
name2	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		

```
geo-shell> format open --input examples/alki2.tif --name alki2
```

Format alki2 opened!

```
geo-shell> raster open --format alki2 --raster alki2 --name alki2  
Opened Format alki2 Raster alki2 as alki2
```

```
geo-shell> format open --input examples/alki3.tif --name alki3  
Format alki3 opened!
```

```
geo-shell> raster open --format alki3 --raster alki3 --name alki3  
Opened Format alki3 Raster alki3 as alki3
```

```
geo-shell> format open --input examples/mosaic.tif --name mosaic  
Format mosaic opened!
```

```
geo-shell> raster mosaic --name1 alki2 --name2 alki3 --output-format mosaic --output-name mosaic  
Mosaic alki2 and alki3 to create mosaic!
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add raster --name map --raster mosaic  
Added mosaic layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_mosaic.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_mosaic.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Reclassify

Reclassify a Raster.

```
geo-shell> raster reclassify --name pc --output-format pcReclass --output-name pcReclass --ranges "0-0=1,0-50=2,50-200=3,200-1000=4,1000-1500=5,1500-4000=6"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
ranges	The comma delimited reclassification ranges (from-to=value)	true		
band	The Raster band to contour	false	0	0
nodata	The NODATA value	false	0	0

```
geo-shell> format open --name pc --input src/test/resources/pc.tif  
Format pc opened!
```

```
geo-shell> raster open --format pc --raster pc --name pc  
Opened Format pc Raster pc as pc
```

```
geo-shell> format open --name pcReclass --input examples/pcReclass.tif  
Format pcReclass opened!
```

```
geo-shell> raster reclassify --name pc --output-format pcReclass --output-name pcReclass --ranges "0-0=1,0-50=2,50-200=3,200-1000=4,1000-1500=5,1500-4000=6"  
Raster pc reclassified to pcReclass!
```

```
geo-shell> style raster colormap --raster pcReclass --values "1=#9fd182,2=#3e7f3c,3=#133912,4=#08306b,5=#FFF8DC,6=#ffffff" --file examples/pcReclass.sld  
Colormap Raster Style for pcReclass written to /home/travis/build/jericks/geo-shell/examples/pcReclass.sld!
```

```
geo-shell> raster style set --name pcReclass --style examples/pcReclass.sld  
Style /home/travis/build/jericks/geo-shell/examples/pcReclass.sld set on pcReclass
```

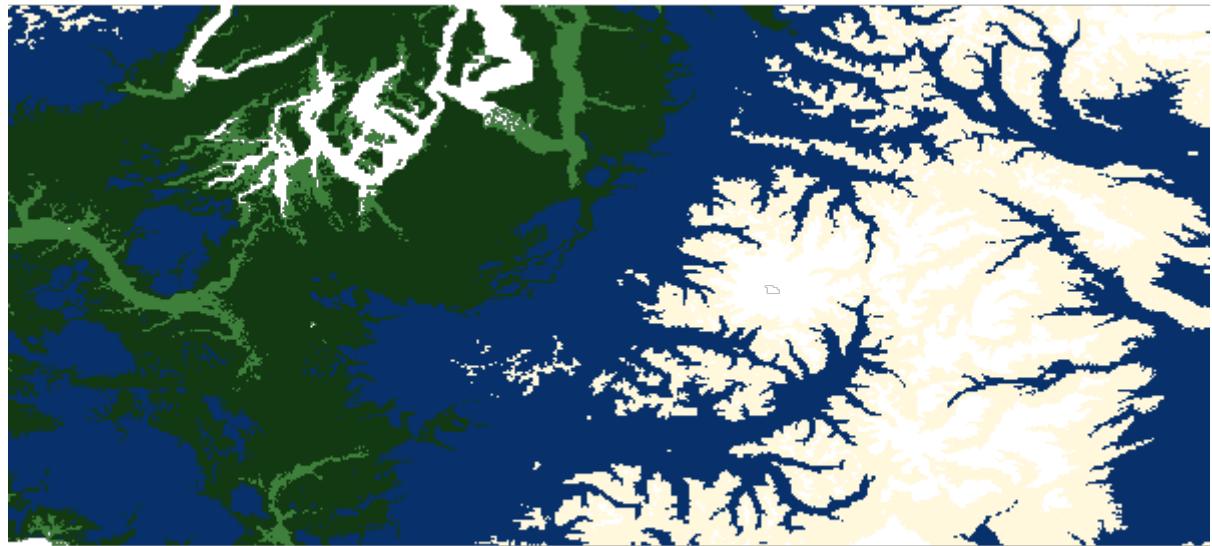
```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add raster --name map --raster pcReclass
```

Added pcReclass layer to map map

```
geo-shell> map draw --name map --file examples/raster_reclassify.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_reclassify.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Reproject

Project a Raster.

```
geo-shell> raster reproject --name earthCropped --output-format earth3857 --output-name earth3857 --projection "EPSG:3857"
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
projection	The projection	true		

```
geo-shell> format open --name earth --input src/test/resources/earth.tif  
Format earth opened!
```

```
geo-shell> raster open --format earth --raster earth --name earth
```

Opened Format earth Raster earth as earth

```
geo-shell> format open --name earthCropped --input examples/earthCropped.tif
Format earthCropped opened!
```

```
geo-shell> raster crop --name earth --output-format earthCropped --output-name earthCropped
--geometry "-180.0,-85.06,180.0,85.06"
Raster earth cropped to earthCropped!
```

```
geo-shell> format open --name earth3857 --input examples/earth3857.tif
Format earth3857 opened!
```

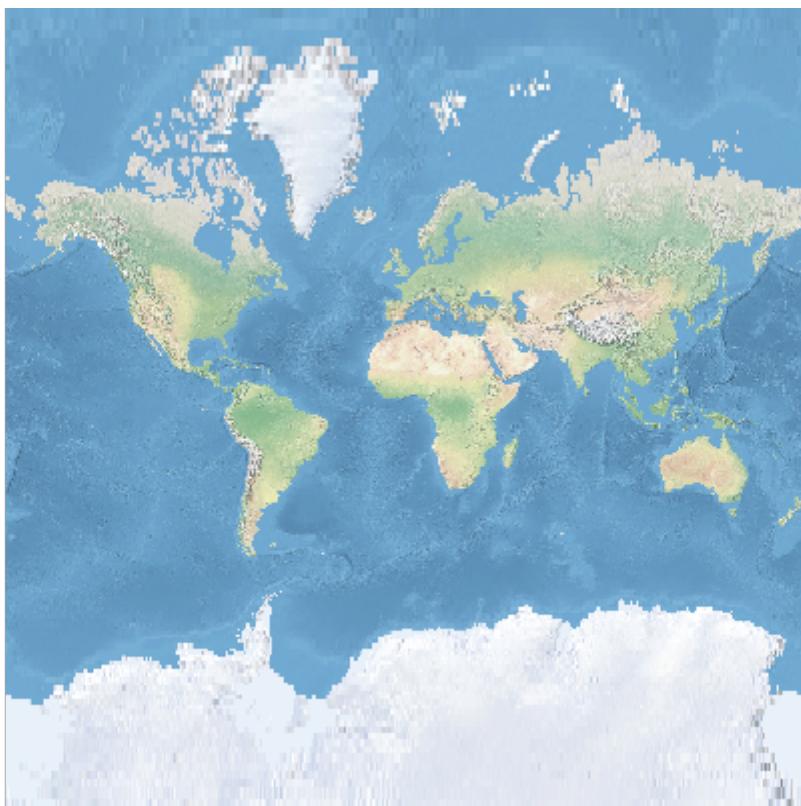
```
geo-shell> raster reproject --name earthCropped --output-format earth3857 --output-name
earth3857 --projection "EPSG:3857"
Raster earthCropped reprojected to earth3857 as EPSG:3857!
```

```
geo-shell> map open --name map
Map map opened!
```

```
geo-shell> map add raster --name map --raster earth3857
Added earth3857 layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_reproject.png
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_reproject.png!
```

```
geo-shell> map close --name map
Map map closed!
```



Scale

Scale a Raster.

```
geo-shell> raster scale --name pc --output-format pcScaled --output-name pcScaled --x 0.5 --y 0.5
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
x	The scale factor along the x axis	true		
y	The scale factor along the y axis	true		
x-trans	The x translation	false	0	0
y-trans	The y translation	false	0	0
interpolation	The interpolation method (bicubic, bicubic2, bilinear, nearest)	false	nearest	nearest

```
geo-shell> format open --name pc --input src/test/resources/pc.tif  
Format pc opened!
```

```
geo-shell> raster open --format pc --raster pc --name pc  
Opened Format pc Raster pc as pc
```

```
geo-shell> format open --name pcScaled --input examples/pcScaled.tif  
Format pcScaled opened!
```

```
geo-shell> raster scale --name pc --output-format pcScaled --output-name pcScaled --x 0.5 --y 0.5  
Raster pc scaled to pcScaled!
```

```
geo-shell> style raster colormap --raster pc --values  
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5" --file examples/pcScaled.sld  
Colormap Raster Style for pc written to /home/travis/build/jericks/geo-shell/examples/pcScaled.sld!
```

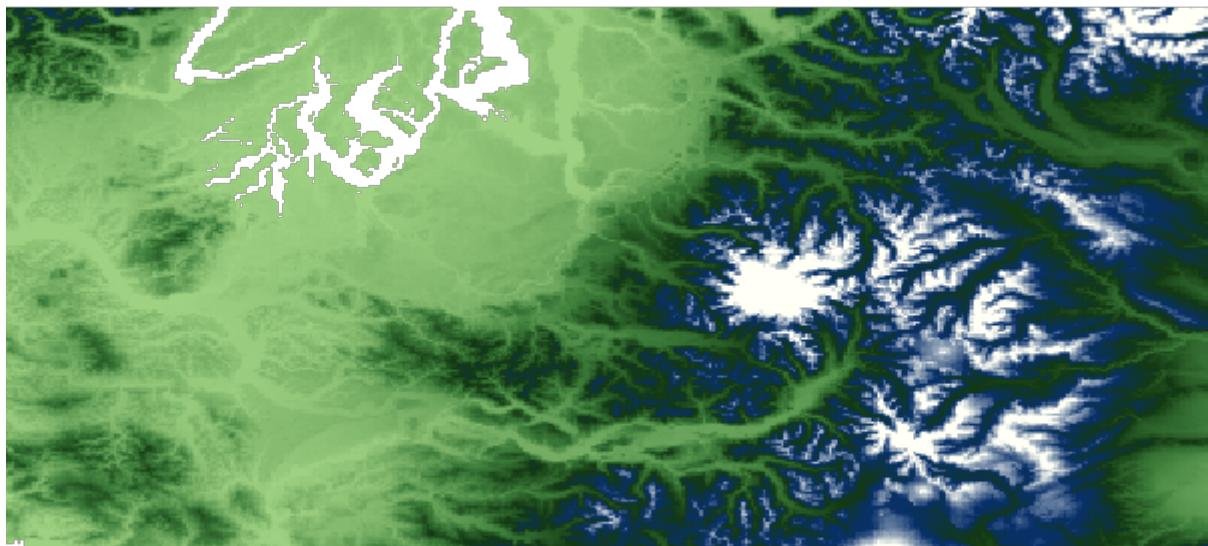
```
geo-shell> raster style set --name pcScaled --style examples/pcScaled.sld  
Style /home/travis/build/jericks/geo-shell/examples/pcScaled.sld set on pcScaled
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add raster --name map --raster pcScaled  
Added pcScaled layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_scale.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_scale.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Shaded Relief

Create a shaded relief raster

```
geo-shell> raster shadedrelief --name pc --output-format pcShaded --output-name pcShaded --scale  
1.0 --altitude 25 --azimuth 260
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		
scale	The scale	true		
altitude	The altitude	true		
azimuth	The azimuth	true		
resx	The x resolution	false	0.5	0.5
resy	The y resolution	false	0.5	0.5
zetafactory	The zeta factory	false	1.0	1.0

algorithm	The x resolution	false	DEFAULT	DEFAULT
-----------	------------------	-------	---------	---------

geo-shell> **format open** --name pc --input src/test/resources/pc.tif
Format pc opened!

geo-shell> **raster open** --format pc --raster pc --name pc
Opened Format pc Raster pc as pc

geo-shell> **format open** --name pcShaded --input examples/pcShaded.tif
Format pcShaded opened!

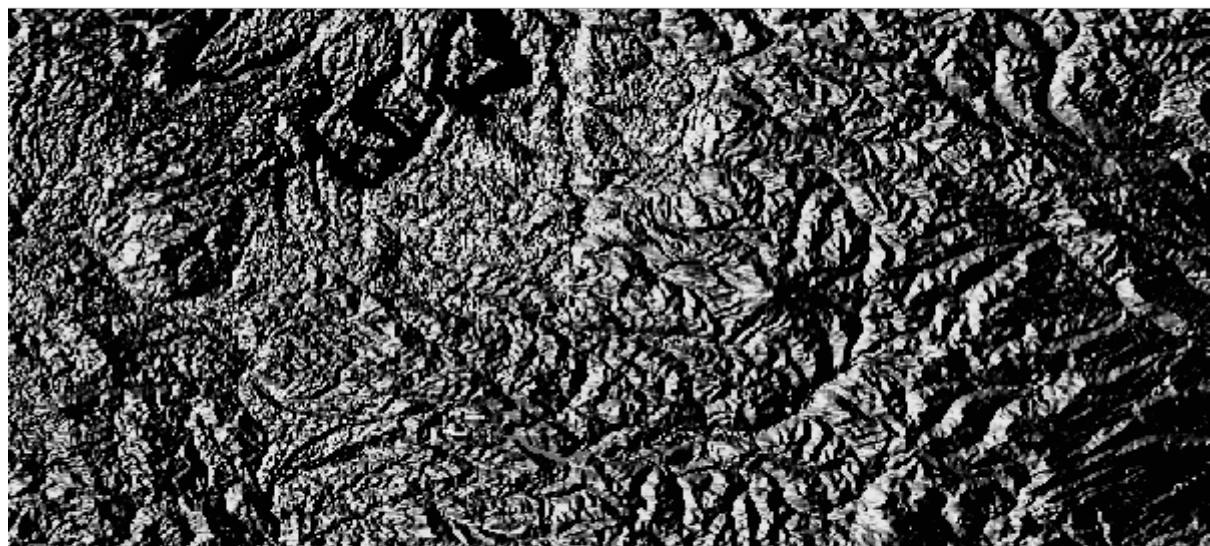
geo-shell> **raster shadedrelief** --name pc --output-format pcShaded --output-name pcShaded --scale 1.0 --altitude 25 --azimuth 260
Create shaded relief pcShaded from pc!

geo-shell> **map open** --name map
Map map opened!

geo-shell> **map add raster** --name map --raster pcShaded
Added pcShaded layer to map map

geo-shell> **map draw** --name map --file examples/raster_shadedrelief.png
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_shadedrelief.png!

geo-shell> **map close** --name map
Map map closed!



Stylize

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

```
geo-shell> raster stylize --name pc --output-format pcStyled --output-name pcStyled
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-format	The output Format Workspace	true		
output-name	The output Raster name	false		

```
geo-shell> format open --name pc --input src/test/resources/pc.tif  
Format pc opened!
```

```
geo-shell> raster open --format pc --raster pc --name pc  
Opened Format pc Raster pc as pc
```

```
geo-shell> style raster colormap --raster pc --values  
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5" --file examples/pc.sld  
Colormap Raster Style for pc written to /home/travis/build/jericks/geo-shell/examples/pc.sld!
```

```
geo-shell> raster style set --name pc --style examples/pc.sld  
Style /home/travis/build/jericks/geo-shell/examples/pc.sld set on pc
```

```
geo-shell> format open --name pcStyled --input examples/pcStyled.tif  
Format pcStyled opened!
```

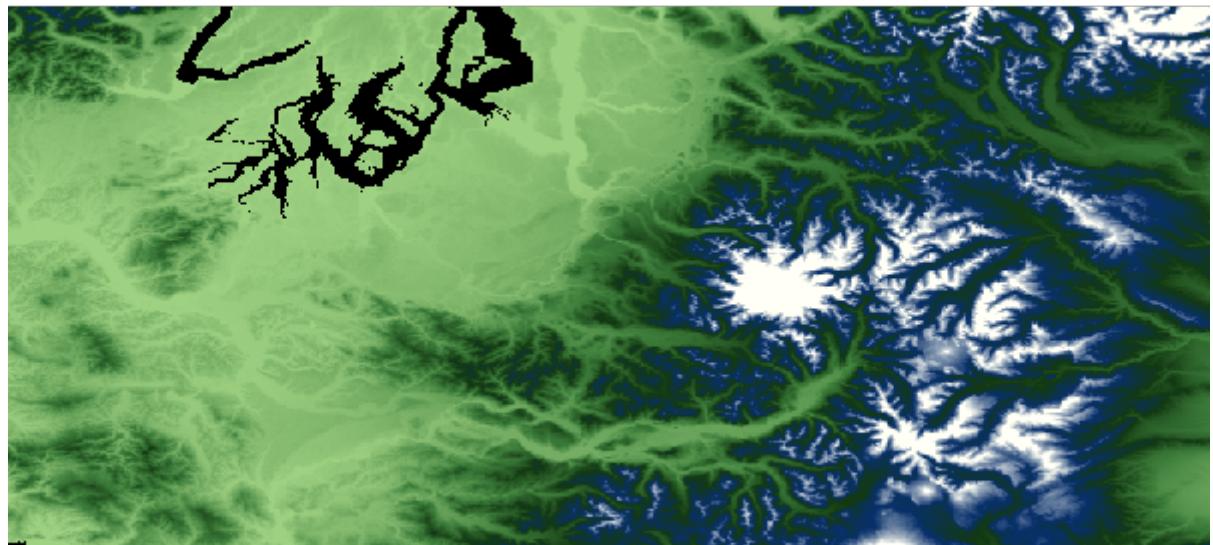
```
geo-shell> raster stylize --name pc --output-format pcStyled --output-name pcStyled  
Stylized pc to create pcStyled!
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add raster --name map --raster pcStyled  
Added pcStyled layer to map map
```

```
geo-shell> map draw --name map --file examples/raster_stylize.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_stylize.png!
```

```
geo-shell> map close --name map  
Map map closed!
```



Polygon

Convert a raster in a polygon

```
geo-shell> raster polygon --name high --output-workspace layers --output-name grid
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The Raster name	true		
output-workspace	The output Layer Workspace	true		
output-name	The output Layer name	true		
band	The band	false	0	0
inside-edges	Whether to include inside edges	false	true	true
roi	The region of interest	false		
nodata	The NODATA value	false	0	0

ranges	The comma delimited reclassification ranges (min,minIncluded, max,maxIncluded)	false		
--------	--	-------	--	--

geo-shell> **format open** --name high --input src/test/resources/high.tif
Format high opened!

geo-shell> **raster open** --format high --raster high --name high
Opened Format high Raster high as high

geo-shell> **workspace open** --name layers --params memory
Workspace layers opened!

geo-shell> **raster polygon** --name high --output-workspace layers --output-name grid
Done converting Raster high to a Polygon Layer grid!

geo-shell> **style raster palette colormap** --min 1 --max 50 --palette MutedTerrain --number 20 --file examples/high.sld
Colormap Palette Raster Style written to /home/travis/build/jericks/geo-shell/examples/high.sld!

geo-shell> **raster style set** --name high --style examples/high.sld
Style /home/travis/build/jericks/geo-shell/examples/high.sld set on high

geo-shell> **style create** --params "stroke=black stroke-width=2 label=value label-size=12" --file examples/grid.sld
Style stroke=black stroke-width=2 label=value label-size=12 written to /home/travis/build/jericks/geo-shell/examples/grid.sld!

geo-shell> **layer style set** --name grid --style examples/grid.sld
Style /home/travis/build/jericks/geo-shell/examples/grid.sld set on grid

geo-shell> **map open** --name map
Map map opened!

geo-shell> **map add raster** --name map --raster high
Added high layer to map map

geo-shell> **map add layer** --name map --layer grid
Added grid layer to map map

geo-shell> **map draw** --name map --file examples/raster_polygon.png --bounds "-180, 90,180,90,EPSC:4326"
Done drawing /home/travis/build/jericks/geo-shell/examples/raster_polygon.png!

geo-shell> **map close** --name map
Map map closed!

17.0	18.0	19.0	20.0
13.0	14.0	15.0	16.0
9.0	10.0	11.0	12.0
5.0	6.0	7.0	8.0

Tile

Open

Open a Tile Layer.

```
geo-shell> tile open --name countries --params src/test/resources/countries.mbtiles
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The tile name	true		
params	The connection parameters	true		

```
geo-shell> tile open --name countries --params src/test/resources/countries.mbtiles
Tile Layer countries opened!
```

```
geo-shell> tile close --name countries
Tile Layer countries closed!
```

Close

Close a Tile Layer.

```
geo-shell> tile close --name countries
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The tile name	true		

```
geo-shell> tile open --name countries --params src/test/resources/countries.mbtiles
Tile Layer countries opened!
```

```
geo-shell> tile close --name countries
Tile Layer countries closed!
```

List

List open Tile Layers.

```
geo-shell> tile list
```



No parameters

```
geo-shell> tile open --name countries --params src/test/resources/countries.mbtiles
Tile Layer countries opened!
```

```
geo-shell> tile list
```

countries = MBTiles

```
geo-shell> tile close --name countries
Tile Layer countries closed!
```

Info

Get information about a Tile Layer.

```
geo-shell> tile info --name countries
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The tile name	true		

```
geo-shell> tile open --name countries --params src/test/resources/countries.mbtiles
Tile Layer countries opened!
```

```
geo-shell> tile info --name countries
```

countries

EPSG:3857

-2.0036395147881314E7,

-2.0037471205137067E7,2.0036395147881314E7,2.003747120513706E7,EPSC:3857

BOTTOM_LEFT

256,256

0,1,1,156412.0,156412.0

```

1,2,2,78206.0,78206.0
2,4,4,39103.0,39103.0
3,8,8,19551.5,19551.5
4,16,16,9775.75,9775.75
5,32,32,4887.875,4887.875
6,64,64,2443.9375,2443.9375
7,128,128,1221.96875,1221.96875
8,256,256,610.984375,610.984375
9,512,512,305.4921875,305.4921875
10,1024,1024,152.74609375,152.74609375
11,2048,2048,76.373046875,76.373046875
12,4096,4096,38.1865234375,38.1865234375
13,8192,8192,19.09326171875,19.09326171875
14,16384,16384,9.546630859375,9.546630859375
15,32768,32768,4.7733154296875,4.7733154296875
16,65536,65536,2.38665771484375,2.38665771484375
17,131072,131072,1.193328857421875,1.193328857421875
18,262144,262144,0.5966644287109375,0.5966644287109375
19,524288,524288,0.29833221435546875,0.29833221435546875

```

```

geo-shell> tile close --name countries
Tile Layer countries closed!

```

Delete

Delete tiles from a Tile Layer.

```

geo-shell> tile delete --name tiles --z 3

```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The tile name	true		
tile	The tile z/x/y	false		
bounds	The bounds	false		
width	The width	false	400	400
height	The height	false	400	400
z	The zoom level	false	0	-1
minx	The min x or column	false		-1
miny	The min y or row	false		-1
maxx	The max x or column	false		-1
maxy	The max y or row	false		-1

```

geo-shell> tile open --name tiles --params target/tiles.mbtiles

```

Tile Layer tiles opened!

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name world  
Map world opened!
```

```
geo-shell> map add layer --name world --layer ocean  
Added ocean layer to map world
```

```
geo-shell> map add layer --name world --layer countries  
Added countries layer to map world
```

```
geo-shell> tile generate --name tiles --map world --start 0 --end 3  
Tiles generated!
```

```
geo-shell> tile delete --name tiles --z 3  
Deleting tiles at z level 3
```

```
geo-shell> map close --name world  
Map world closed!
```

Generate

Generate tiles for a Tile Layer.

```
geo-shell> tile generate --name tiles --map world --start 0 --end 3
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The tile name	true		
map	The map name	true		
start	The map name	true		
end	The map name	true		
bounds	The map name	false		

metatile	The metatile width,height	false		
missingOnly	The map name	false	false	false
verbose	The map name	false	false	false

geo-shell> **tile open** --name tiles --params target/tiles.mbtiles

Tile Layer tiles opened!

geo-shell> **workspace open** --name naturalearth --params examples/naturalearth.gpkg

Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer countries --name countries

Opened Workspace naturalearth Layer countries as countries

geo-shell> **layer style set** --name countries --style examples/countries.sld

Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries

geo-shell> **layer open** --workspace naturalearth --layer ocean --name ocean

Opened Workspace naturalearth Layer ocean as ocean

geo-shell> **layer style set** --name ocean --style examples/ocean.sld

Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> **map open** --name world

Map world opened!

geo-shell> **map add layer** --name world --layer ocean

Added ocean layer to map world

geo-shell> **map add layer** --name world --layer countries

Added countries layer to map world

geo-shell> **tile generate** --name tiles --map world --start 0 --end 3

Tiles generated!

geo-shell> **format open** --name world_level2 --input examples/tile_generate.png

Format world_level2 opened!

geo-shell> **tile stitch raster** --name tiles --format world_level2 --raster world_level2 --z 2

Done stitching Raster world_level2 from tiles!

geo-shell> **map close** --name world

Map world closed!



Stitch Raster

Create a Raster from a Tile Layer.

```
geo-shell> tile stitch raster --name countries --format states --raster states --bounds -18217695.5734,1222992.4526,-4207094.0368,7924991.0926
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The tile name	true		
format	The raster format name	true		
raster	The raster name	true		
bounds	The bounds	false		

width	The raster width	false	400	400
height	The raster height	false	400	400
z	The zoom level	false	0	-1
minx	The min x or column	false		-1
miny	The min y or row	false		-1
maxx	The max x or column	false		-1
maxy	The max y or row	false		-1

Create a Raster from a Tile Layer with a geographic bounds.

```
geo-shell> tile open --name countries --params src/test/resources/countries.mbtiles
Tile Layer countries opened!
```

```
geo-shell> format open --name states --input examples/tile_stitch_bounds.png
Format states opened!
```

```
geo-shell> tile stitch raster --name countries --format states --raster states --bounds
-18217695.5734,1222992.4526,-4207094.0368,7924991.0926
Done stitching Raster states from countries!
```



Tiles

List tiles within a given bounds.

```
geo-shell> tile tiles --name countries --z 8 --bounds -13787405.4140,5872198.2610,
-13349574.1159,6081635.7185
```

Name	Description	Mandatory	Specified Default	Unspecified Default

name	The tile name	true		
bounds	The bounds	true		
z	The zoom level	true		

```
geo-shell> tile open --name countries --params src/test/resources/countries.mbtiles
Tile Layer countries opened!
```

```
geo-shell> tile tiles --name countries --z 8 --bounds -13787405.4140,5872198.2610,
-13349574.1159,6081635.7185
8/39/165
8/40/165
8/41/165
8/42/165
8/39/166
8/40/166
8/41/166
8/42/166
```

```
geo-shell> tile close --name countries
Tile Layer countries closed!
```

Vector Grid

Create a Vector Grid Layer from the pyramid of a Tile Layer.

```
geo-shell> tile vector grid --name countries --workspace layers --layer level3 --z 3
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The tile name	true		
workspace	The workspace name	true		
layer	The layer name	true		
bounds	The bounds	false		
width	The raster width	false	400	400
height	The raster height	false	400	400
z	The zoom level	false	0	-1
minx	The min x or column	false		-1
miny	The min y or row	false		-1
maxx	The max x or column	false		-1
maxy	The max y or row	false		-1

```
geo-shell> tile open --name countries --params src/test/resources/countries.mbtiles
```

Tile Layer countries opened!

```
geo-shell> workspace open --name layers --params memory
Workspace layers opened!
```

```
geo-shell> tile vector grid --name countries --workspace layers --layer level3 --z 3
Done generating the vector grid level3 from countries!
```

```
geo-shell> style vector default --layer level3 --color #ffffff --opacity 0.25 --file examples/level3.sld
Default Vector Style for level3 written to /home/travis/build/jericks/geo-shell/examples/level3.sld!
```

```
geo-shell> layer style set --name level3 --style examples/level3.sld
Style /home/travis/build/jericks/geo-shell/examples/level3.sld set on level3
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name vectorGridMap
Map vectorGridMap opened!
```

```
geo-shell> map add layer --name vectorGridMap --layer ocean
Added ocean layer to map vectorGridMap
```

```
geo-shell> map add layer --name vectorGridMap --layer countries
Added countries layer to map vectorGridMap
```

```
geo-shell> map add layer --name vectorGridMap --layer level3
Added level3 layer to map vectorGridMap
```

```
geo-shell> map draw --name vectorGridMap --file examples/tile_vector_grid.png --projection
EPSG:3857 --width 400 --height 400 --bounds -20026376.39,-20048966.10,20026376.39,20048966.10
Done drawing /home/travis/build/jericks/geo-shell/examples/tile_vector_grid.png!
```

```
geo-shell> map close --name vectorGridMap
Map vectorGridMap closed!
```



Style

Create

Create a simple style.

```
geo-shell> style create --params "stroke=black stroke-width=0.25 fill=wheat" --file examples/style_create.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
params	The style parameters	true		
file	The output file	true		

```
geo-shell> style create --params "stroke=black stroke-width=0.25 fill=wheat" --file examples/style_create.sld
Style stroke=black stroke-width=0.25 fill=wheat written to /home/travis/build/jericks/geo-shell/examples/style_create.sld!
```

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/style_create.sld
```

Style /home/travis/build/jericks/geo-shell/examples/style_create.sld set on countries

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map open --name map
Map map opened!
```

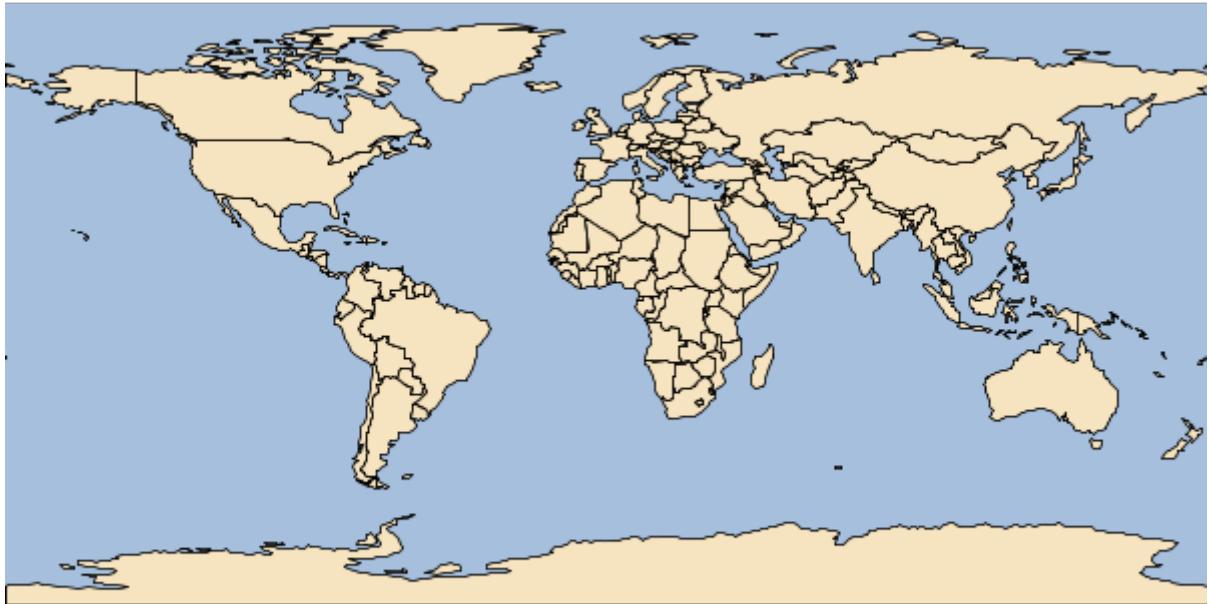
```
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
Added countries layer to map map
```

```
geo-shell> map draw --name map --file examples/style_create.png
Done drawing /home/travis/build/jericks/geo-shell/examples/style_create.png!
```

```
geo-shell> map close --name map
Map map closed!
```

```
<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:PolygonSymbolizer>
            <sld:Fill>
              <sld:CssParameter name="fill">#f5deb3</sld:CssParameter>
              <sld:CssParameter name="fill-opacity">0.6</sld:CssParameter>
            </sld:Fill>
          </sld:PolygonSymbolizer>
          <sld:LineSymbolizer>
            <sld:Stroke>
              <sld:CssParameter name="stroke-width">0.25</sld:CssParameter>
            </sld:Stroke>
          </sld:LineSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>
```



Vector Default

Create a default vector style.

```
geo-shell> style vector default --layer countries --color #F5F5DC --file examples/countries_default.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
layer	The Layer	true		
color	The color	false	#f2f2f2	#f2f2f2
opacity	The opacity	false	1.0	1.0
file	The output file	true		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> style vector default --layer countries --color #F5F5DC --file examples/countries_default.sld
Default Vector Style for countries written to /home/travis/build/jericks/geo-shell/examples/countries_default.sld!
```

```
geo-shell> layer style set --name countries --style examples/countries_default.sld
Style /home/travis/build/jericks/geo-shell/examples/countries_default.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> style vector default --layer ocean --color DeepSkyBlue --file examples/ocean_default.sld
Default Vector Style for ocean written to /home/travis/build/jericks/geo-
shell/examples/ocean_default.sld!
```

```
geo-shell> layer style set --name ocean --style examples/ocean_default.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean_default.sld set on ocean
```

```
geo-shell> map open --name map
Map map opened!
```

```
geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map
```

```
geo-shell> map add layer --name map --layer countries
Added countries layer to map map
```

```
geo-shell> map draw --name map --file examples/style_vector_default.png
Done drawing /home/travis/build/jericks/geo-shell/examples/style_vector_default.png!
```

```
geo-shell> map close --name map
Map map closed!
```

Country Style

```

<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:PolygonSymbolizer>
            <sld:Fill>
              <sld:CssParameter name="fill">#f5f5dc</sld:CssParameter>
            </sld:Fill>
          </sld:PolygonSymbolizer>
          <sld:LineSymbolizer>
            <sld:Stroke>
              <sld:CssParameter name="stroke">#abab9a</sld:CssParameter>
              <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
            </sld:Stroke>
          </sld:LineSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```

Ocean Style

```
<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:PolygonSymbolizer>
            <sld:Fill>
              <sld:CssParameter name="fill">#00bfff</sld:CssParameter>
            </sld:Fill>
          </sld:PolygonSymbolizer>
          <sld:LineSymbolizer>
            <sld:Stroke>
              <sld:CssParameter name="stroke">#0085b2</sld:CssParameter>
              <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
            </sld:Stroke>
          </sld:LineSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>
```



Vector Gradient

Create a gradient vector style.

```
geo-shell> style vector gradient --layer countries --field PEOPLE --colors greens --number 8  
--method quantile --file examples/style_vector_gradient.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
layer	The Layer	true		
field	The field	true		
number	The number of categories	true		
colors	The colors	true		
method	The classification method (Quantile or EqualInterval)	false	Quantile	Quantile
elsemode	The else mode (ignore, min, max)	false	ignore	ignore
file	The output file	true		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```

geo-shell> style vector gradient --layer countries --field PEOPLE --colors greens --number 8
--method quantile --file examples/style_vector_gradient.sld
Gradient Vector Style for countries's PEOPLE Field written to /home/travis/build/jericks/geo-
shell/examples/style_vector_gradient.sld!

geo-shell> layer style set --name countries --style examples/style_vector_gradient.sld
Style /home/travis/build/jericks/geo-shell/examples/style_vector_gradient.sld set on countries

geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> map open --name map
Map map opened!

geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map

geo-shell> map add layer --name map --layer countries
Added countries layer to map map

geo-shell> map draw --name map --file examples/style_vector_gradient.png
Done drawing /home/travis/build/jericks/geo-shell/examples/style_vector_gradient.png!

geo-shell> map close --name map
Map map closed!

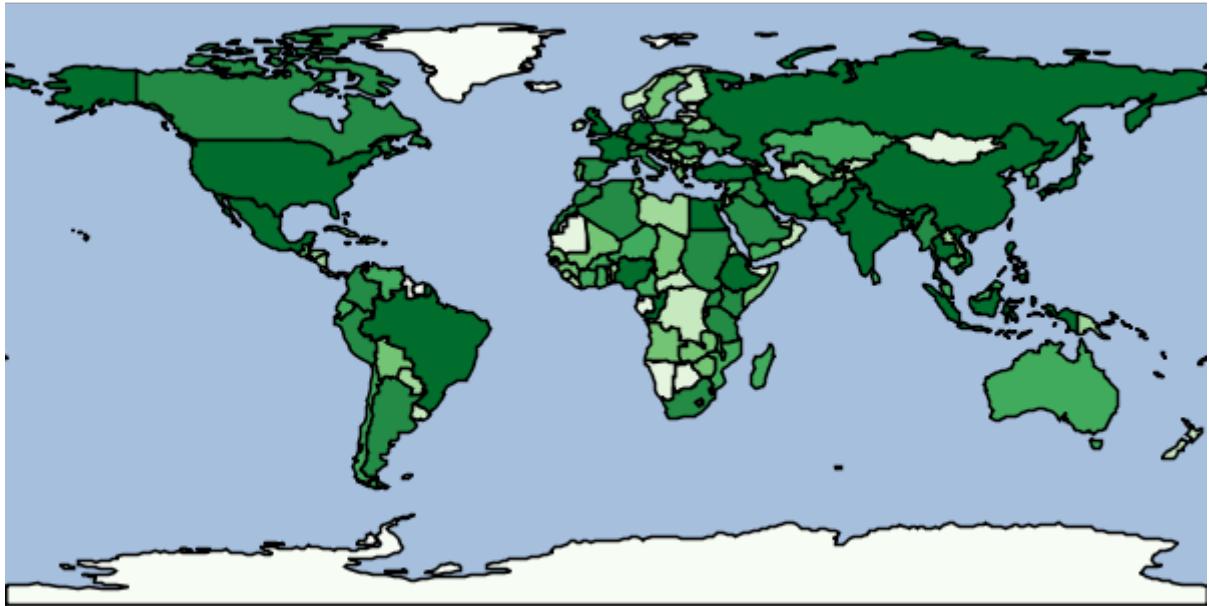
```

```

<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
 xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
 xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
 version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <ogc:Filter>
            <ogc:And>
              <ogc:PropertyIsGreaterThanOrEqualTo>
                <ogc:PropertyName>PEOPLE</ogc:PropertyName>
                <ogc:Literal>0</ogc:Literal>
              </ogc:PropertyIsGreaterThanOrEqualTo>
              <ogc:PropertyIsLessThan>
                <ogc:PropertyName>PEOPLE</ogc:PropertyName>
                <ogc:Literal>833285</ogc:Literal>
              </ogc:PropertyIsLessThan>
            </ogc:And>
          </ogc:Filter>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```

```
</ogc:PropertyIsLessThan>
</ogc:And>
</ogc:Filter>
<sld:PolygonSymbolizer>
  <sld:Fill>
    <sld:CssParameter name="fill">#F7FCF5</sld:CssParameter>
  </sld:Fill>
</sld:PolygonSymbolizer>
<sld:LineSymbolizer>
  <sld:Stroke/>
</sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
  <ogc:Filter>
    <ogc:And>
      <ogc:PropertyIsGreaterThanOrEqualTo>
        <ogc:PropertyName>PEOPLE</ogc:PropertyName>
        <ogc:Literal>833285</ogc:Literal>
      </ogc:PropertyIsGreaterThanOrEqualTo>
      <ogc:PropertyIsLessThan>
        <ogc:PropertyName>PEOPLE</ogc:PropertyName>
        <ogc:Literal>3360474</ogc:Literal>
      </ogc:PropertyIsLessThan>
    </ogc:And>
  </ogc:Filter>
  <sld:PolygonSymbolizer>
    <sld:Fill>
      <sld:CssParameter name="fill">#E5F5E0</sld:CssParameter>
    </sld:Fill>
  </sld:PolygonSymbolizer>
  <sld:LineSymbolizer>
```



Vector Unique Values

Create a unique values vector style.

```
geo-shell> style vector uniquevalues --layer countries --field NAME --colors random --file examples/style_vector_uniquevalues.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
layer	The Layer	true		
field	The field	true		
colors	The colors	true		
file	The output file	true		

```
geo-shell> workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> style vector uniquevalues --layer countries --field NAME --colors random --file examples/style_vector_uniquevalues.sld
Unique Values Vector Style for countries's NAME Field written to /home/travis/build/jericks/geo-shell/examples/style_vector_uniquevalues.sld!
```

```
geo-shell> layer style set --name countries --style examples/style_vector_uniquevalues.sld
Style /home/travis/build/jericks/geo-shell/examples/style_vector_uniquevalues.sld set on countries
```

```

geo-shell> layer open --workspace naturalearth --layer ocean --name ocean
Opened Workspace naturalearth Layer ocean as ocean

geo-shell> layer style set --name ocean --style examples/ocean.sld
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean

geo-shell> map open --name map
Map map opened!

geo-shell> map add layer --name map --layer ocean
Added ocean layer to map map

geo-shell> map add layer --name map --layer countries
Added countries layer to map map

geo-shell> map draw --name map --file examples/style_vector_uniquevalues.png
Done drawing /home/travis/build/jericks/geo-shell/examples/style_vector_uniquevalues.png!

geo-shell> map close --name map
Map map closed!

```

```

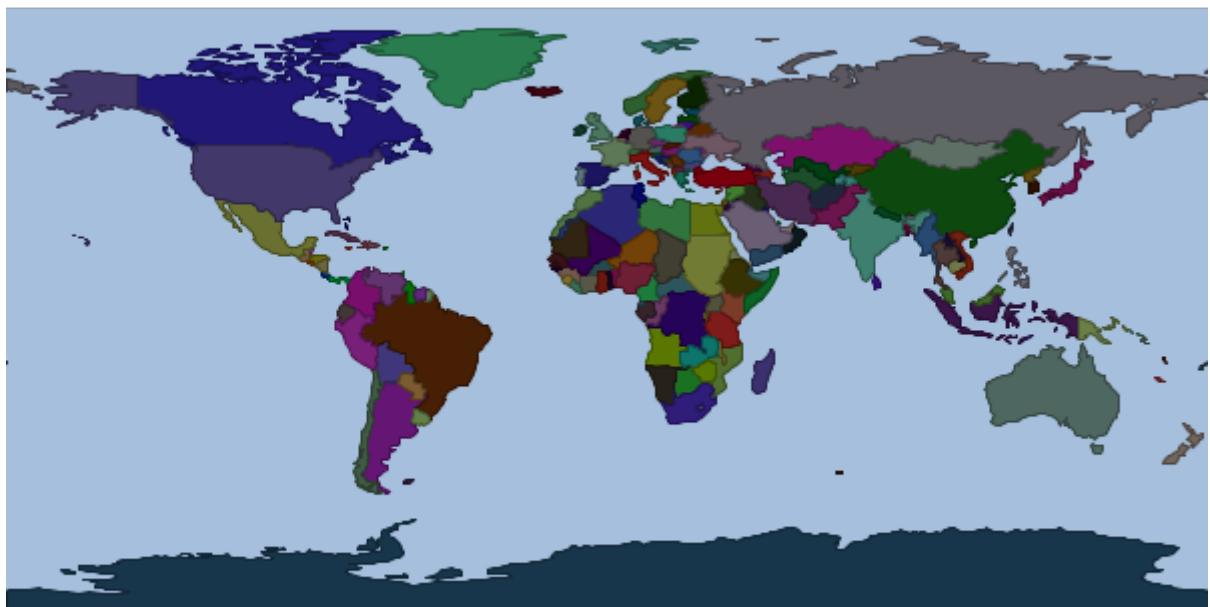
<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
  xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
  xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
  version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <ogc:Filter>
            <ogc:PropertyIsEqualTo>
              <ogc:PropertyName>NAME</ogc:PropertyName>
              <ogc:Literal>Afghanistan</ogc:Literal>
            </ogc:PropertyIsEqualTo>
          </ogc:Filter>
          <sld:PolygonSymbolizer>
            <sld:Fill>
              <sld:CssParameter name="fill">#1e263b</sld:CssParameter>
            </sld:Fill>
          </sld:PolygonSymbolizer>
          <sld:LineSymbolizer>
            <sld:Stroke>
              <sld:CssParameter name="stroke">#151a29</sld:CssParameter>
              <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
            </sld:Stroke>
          </sld:LineSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```

```

</sld:Rule>
<sld:Rule>
  <ogc:Filter>
    <ogc:PropertyIsEqualTo>
      <ogc:PropertyName>NAME</ogc:PropertyName>
      <ogc:Literal>Albania</ogc:Literal>
    </ogc:PropertyIsEqualTo>
  </ogc:Filter>
  <sld:PolygonSymbolizer>
    <sld:Fill>
      <sld:CssParameter name="fill">#721f03</sld:CssParameter>
    </sld:Fill>
  </sld:PolygonSymbolizer>
  <sld:LineSymbolizer>
    <sld:Stroke>
      <sld:CssParameter name="stroke">#4f1502</sld:CssParameter>
      <sld:CssParameter name="stroke-width">0.5</sld:CssParameter>
    </sld:Stroke>
  </sld:LineSymbolizer>
</sld:Rule>
<sld:Rule>
  <ogc:Filter>
    <ogc:PropertyIsEqualTo>

```



Vector Unique Values From Text File

Create a unique values vector style from a text file

```
geo-shell> style vector uniquevaluesfromtext --field UnitSymbol --textFile
```

```
src/test/resources/mars/I1802ABC_geo_units_RGBlut.txt --geometryType polygon --styleFile examples/style_vector_uniquevaluesfromtext.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
field	The field name	true		
geometryType	The geometry type	true		
textFile	The input text file	true		
styleFile	The output sld or ysld file	true		

```
geo-shell> workspace open --name mars --params src/test/resources/mars  
Workspace mars opened!
```

```
geo-shell> layer open --workspace mars --layer geo_units_oc_dd --name mars  
Opened Workspace mars Layer geo_units_oc_dd as mars
```

```
geo-shell> style vector uniquevaluesfromtext --field UnitSymbol --textFile  
src/test/resources/mars/I1802ABC_geo_units_RGBlut.txt --geometryType polygon --styleFile  
examples/style_vector_uniquevaluesfromtext.sld  
Create a unique values style from /home/travis/build/jericks/geo-  
shell/src/test/resources/mars/I1802ABC_geo_units_RGBlut.txt for UnitSymbol and polygon to  
/home/travis/build/jericks/geo-shell/examples/style_vector_uniquevaluesfromtext.sld
```

```
geo-shell> layer style set --name mars --style examples/style_vector_uniquevaluesfromtext.sld  
Style /home/travis/build/jericks/geo-shell/examples/style_vector_uniquevaluesfromtext.sld set on  
mars
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add layer --name map --layer mars  
Added mars layer to map map
```

```
geo-shell> map draw --name map --file examples/style_vector_uniquevaluesfromtext.png  
Done drawing /home/travis/build/jericks/geo-  
shell/examples/style_vector_uniquevaluesfromtext.png!
```

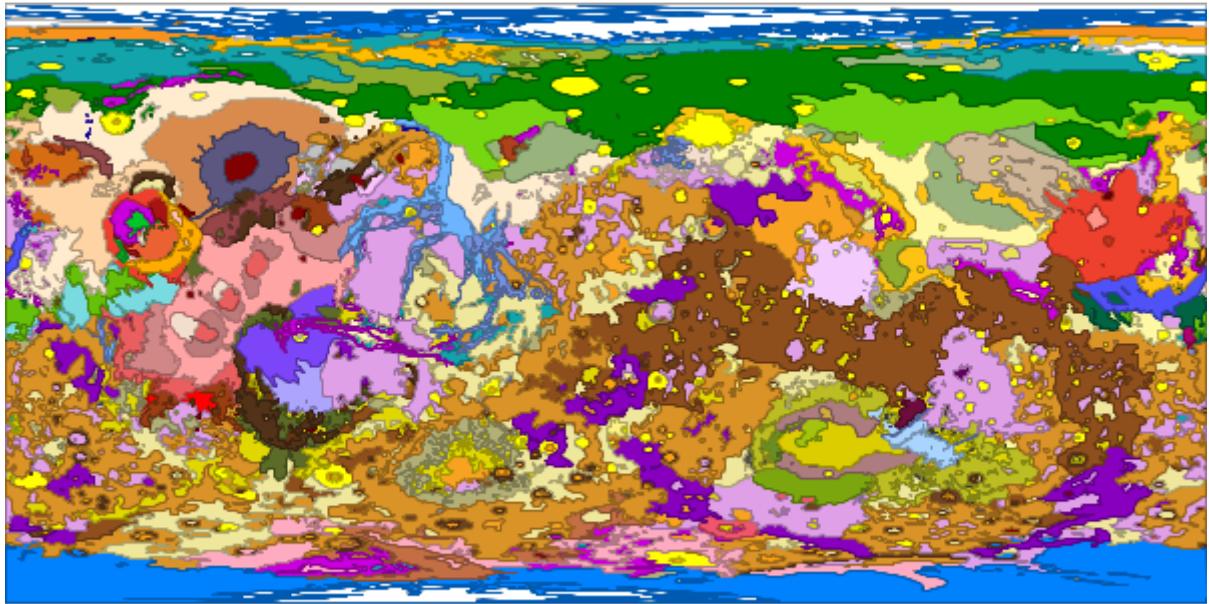
```
geo-shell> map close --name map  
Map map closed!
```

```
<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor  
xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"  
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"  
version="1.0.0">  
  <sld:UserLayer>  
    <sld:LayerFeatureConstraints>  
      <sld:FeatureTypeConstraint/>  
    </sld:LayerFeatureConstraints>
```

```

<sld:UserStyle>
  <sld:Name>Default Styler</sld:Name>
  <sld:FeatureTypeStyle>
    <sld:Name>name</sld:Name>
    <sld:Rule>
      <ogc:Filter>
        <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>UnitSymbol</ogc:PropertyName>
          <ogc:Literal>AHa</ogc:Literal>
        </ogc:PropertyIsEqualTo>
      </ogc:Filter>
      <sld:PolygonSymbolizer>
        <sld:Fill>
          <sld.CssParameter name="fill">#af006f</sld.CssParameter>
        </sld:Fill>
      </sld:PolygonSymbolizer>
      <sld:LineSymbolizer>
        <sld:Stroke>
          <sld.CssParameter name="stroke">#7a004d</sld.CssParameter>
          <sld.CssParameter name="stroke-width">0.5</sld.CssParameter>
        </sld:Stroke>
      </sld:LineSymbolizer>
    </sld:Rule>
    <sld:Rule>
      <ogc:Filter>
        <ogc:PropertyIsEqualTo>
          <ogc:PropertyName>UnitSymbol</ogc:PropertyName>
          <ogc:Literal>AHat</ogc:Literal>
        </ogc:PropertyIsEqualTo>
      </ogc:Filter>
      <sld:PolygonSymbolizer>
        <sld:Fill>
          <sld.CssParameter name="fill">#c03616</sld.CssParameter>
        </sld:Fill>
      </sld:PolygonSymbolizer>
      <sld:LineSymbolizer>
        <sld:Stroke>
          <sld.CssParameter name="stroke">#86250f</sld.CssParameter>
          <sld.CssParameter name="stroke-width">0.5</sld.CssParameter>
        </sld:Stroke>
      </sld:LineSymbolizer>
    </sld:Rule>
    <sld:Rule>
      <ogc:Filter>
        <ogc:PropertyIsEqualTo>

```



Raster Default

Create a default raster style.

```
geo-shell> style raster default --raster pc --opacity 0.75 --file examples/style_raster_default.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
raster	The Raster	true		
opacity	The opacity	false	1.0	1.0
file	The output file	true		

```
geo-shell> format open --name pierce_county --input src/test/resources/pc.tif  
Format pierce_county opened!
```

```
geo-shell> raster open --format pierce_county --raster pc --name pc  
Opened Format pierce_county Raster pc as pc
```

```
geo-shell> style raster default --raster pc --opacity 0.75 --file examples/style_raster_default.sld  
Default Raster Style for pc written to /home/travis/build/jericks/geo-  
shell/examples/style_raster_default.sld!
```

```
geo-shell> raster style set --name pc --style examples/style_raster_default.sld  
Style /home/travis/build/jericks/geo-shell/examples/style_raster_default.sld set on pc
```

```
geo-shell> map open --name map  
Map map opened!
```

```
geo-shell> map add raster --name map --raster pc
```

```
Added pc layer to map map
```

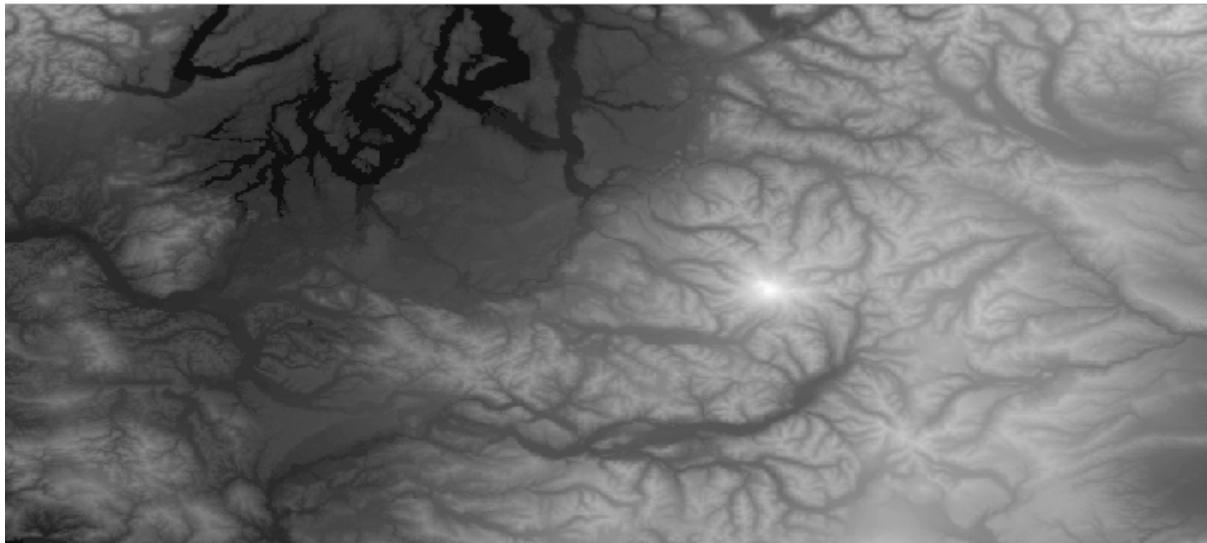
```
geo-shell> map draw --name map --file examples/style_raster_default.png
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/style_raster_default.png!
```

```
geo-shell> map close --name map
```

```
Map map closed!
```

```
<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
 xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
 xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
 version="1.0.0">
  <sld:UserLayer>
   <sld:LayerFeatureConstraints>
    <sld:FeatureTypeConstraint/>
   </sld:LayerFeatureConstraints>
   <sld:UserStyle>
    <sld:Name>Default Styler</sld:Name>
    <sld:FeatureTypeStyle>
     <sld:Name>name</sld:Name>
     <sld:Rule>
      <sld:RasterSymbolizer>
       <sld:Geometry>
        <ogc:Literal>grid</ogc:Literal>
       </sld:Geometry>
       <sld:Opacity>0.75</sld:Opacity>
       <sld:ContrastEnhancement/>
      </sld:RasterSymbolizer>
     </sld:Rule>
    </sld:FeatureTypeStyle>
   </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>
```



Raster Color Map

Create a color map raster style.

```
geo-shell> style raster colormap --raster pc --values  
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5"  
examples/style_raster_colormap.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
raster	The Raster	true		
opacity	The opacity	false	1.0	1.0
values	The comma delimited list of values (key=value)	true		
type	The type (intervals, values, ramp)	false	ramp	ramp
extended	Whether to use extended colors or not	false	false	false
file	The output file	true		

```
geo-shell> format open --name pierce_county --input src/test/resources/pc.tif  
Format pierce_county opened!
```

```
geo-shell> raster open --format pierce_county --raster pc --name pc
```

Opened Format pierce_county Raster pc as pc

```
geo-shell> style raster colormap --raster pc --values
"25=#9fd182,470=#3e7f3c,920=#133912,1370=#08306b,1820=#fffff5"
--file examples/style_raster_colormap.sld
Colormap Raster Style for pc written to /home/travis/build/jericks/geo-
shell/examples/style_raster_colormap.sld!
```

```
geo-shell> raster style set --name pc --style examples/style_raster_colormap.sld
Style /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.sld set on pc
```

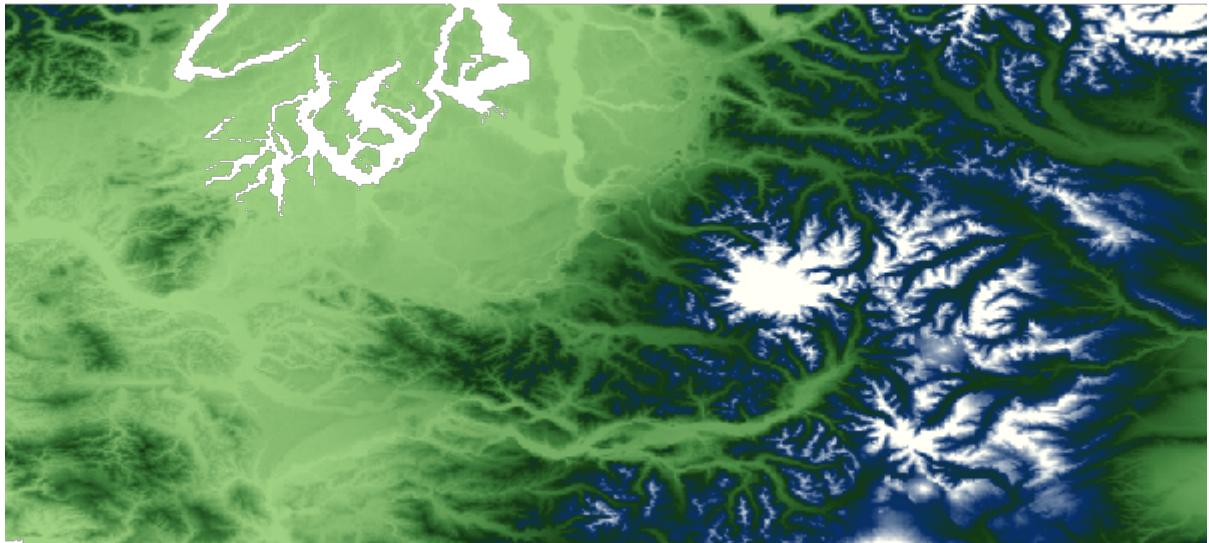
```
geo-shell> map open --name map
Map map opened!
```

```
geo-shell> map add raster --name map --raster pc
Added pc layer to map map
```

```
geo-shell> map draw --name map --file examples/style_raster_colormap.png
Done drawing /home/travis/build/jericks/geo-shell/examples/style_raster_colormap.png!
```

```
geo-shell> map close --name map
Map map closed!
```

```
<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:RasterSymbolizer>
            <sld:Geometry>
              <ogc:Literal>grid</ogc:Literal>
            </sld:Geometry>
            <sld:ColorMap>
              <sld:ColorMapEntry color="#9fd182" opacity="1.0" quantity="25"/>
              <sld:ColorMapEntry color="#3e7f3c" opacity="1.0" quantity="470"/>
              <sld:ColorMapEntry color="#133912" opacity="1.0" quantity="920"/>
              <sld:ColorMapEntry color="#08306b" opacity="1.0" quantity="1370"/>
              <sld:ColorMapEntry color="#fffff5" opacity="1.0" quantity="1820"/>
            </sld:ColorMap>
            <sld:ContrastEnhancement/>
          </sld:RasterSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>
```



Raster Palette Color Map

Create a color map raster style from a color palette.

```
geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file examples/style_raster_palette_colormap.sld
```

Name	Description	Mandatory	Specified Default	Unspecified Default
min	The min value	true		
max	The max value	true		
palette	The color palette name (from Color Brewer)	true		
number	The number of categories	true		
type	The type of interpolation	false	ramp	ramp
extended	Whether to use extended colors	false	false	false
opacity	The opacity	false	1.0	1.0
file	The output file	true		

```
geo-shell> format open --name high --input src/test/resources/high.tif  
Format high opened!
```

```
geo-shell> raster open --format high --raster high --name high
Opened Format high Raster high as high

geo-shell> style raster palette colormap --min 1 --max 50 --palette MutedTerrain --number 20 --file
examples/style_raster_palette_colormap.sld
Colormap      Palette      Raster      Style      written      to      /home/travis/build/jericks/geo-
shell/examples/style_raster_palette_colormap.sld!

geo-shell> raster style set --name high --style examples/style_raster_palette_colormap.sld
Style /home/travis/build/jericks/geo-shell/examples/style_raster_palette_colormap.sld set on high

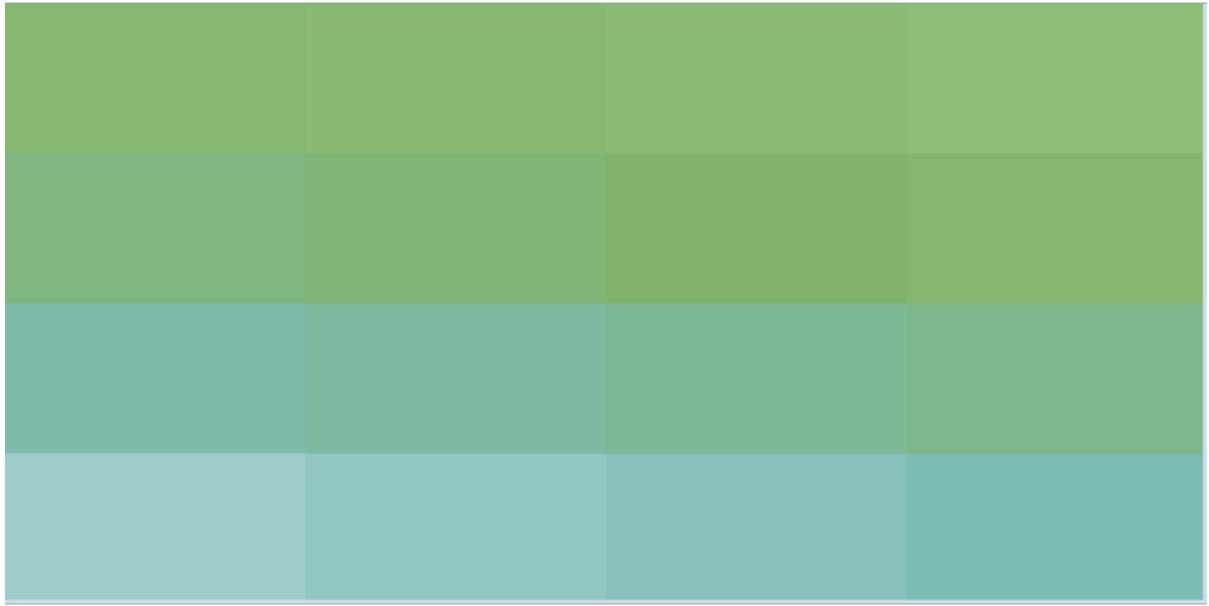
geo-shell> map open --name map
Map map opened!

geo-shell> map add raster --name map --raster high
Added high layer to map map
```

```

<?xml version="1.0" encoding="UTF-8"?><sld:StyledLayerDescriptor
xmlns="http://www.opengis.net/sld" xmlns:sld="http://www.opengis.net/sld"
xmlns:gml="http://www.opengis.net/gml" xmlns:ogc="http://www.opengis.net/ogc"
version="1.0.0">
  <sld:UserLayer>
    <sld:LayerFeatureConstraints>
      <sld:FeatureTypeConstraint/>
    </sld:LayerFeatureConstraints>
    <sld:UserStyle>
      <sld:Name>Default Styler</sld:Name>
      <sld:FeatureTypeStyle>
        <sld:Name>name</sld:Name>
        <sld:Rule>
          <sld:RasterSymbolizer>
            <sld:Geometry>
              <ogc:Literal>grid</ogc:Literal>
            </sld:Geometry>
            <sld:ColorMap>
              <sld:ColorMapEntry color="#CEE1E8" opacity="1.0" quantity="1.0"/>
              <sld:ColorMapEntry color="#7CFCB5" opacity="1.0" quantity="8.0"/>
              <sld:ColorMapEntry color="#82B36D" opacity="1.0" quantity="15.0"/>
              <sld:ColorMapEntry color="#94C279" opacity="1.0" quantity="22.0"/>
              <sld:ColorMapEntry color="#D1DE8D" opacity="1.0" quantity="29.0"/>
              <sld:ColorMapEntry color="#EDECC3" opacity="1.0" quantity="36.0"/>
              <sld:ColorMapEntry color="#CCAFB4" opacity="1.0" quantity="43.0"/>
              <sld:ColorMapEntry color="#C99884" opacity="1.0" quantity="50.0"/>
            </sld:ColorMap>
            <sld:ContrastEnhancement/>
          </sld:RasterSymbolizer>
        </sld:Rule>
      </sld:FeatureTypeStyle>
    </sld:UserStyle>
  </sld:UserLayer>
</sld:StyledLayerDescriptor>

```



Map

Open

Open a new Map.

```
geo-shell> map open --name earth
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		

```
geo-shell> map open --name earth
```

Map earth opened!

```
geo-shell> map close --name earth
```

Map earth closed!

Close

Close a Map.

```
geo-shell> map close --name earth
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		

```
geo-shell> map open --name earth  
Map earth opened!
```

```
geo-shell> map close --name earth  
Map earth closed!
```

List

List open Maps.

```
geo-shell> map list
```



No parameters

```
geo-shell> map open --name earth  
Map earth opened!
```

```
geo-shell> map open --name us  
Map us opened!
```

```
geo-shell> map list
```

earth

us

```
geo-shell> map close --name earth  
Map earth closed!
```

```
geo-shell> map close --name us  
Map us closed!
```

Add Layer

Add a Vector Layer.

```
geo-shell> map add layer --name world --layer countries
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		
layer	The layer	true		
mapLayerName	The map layer name	false		

```
geo-shell> map open --name world  
Map world opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

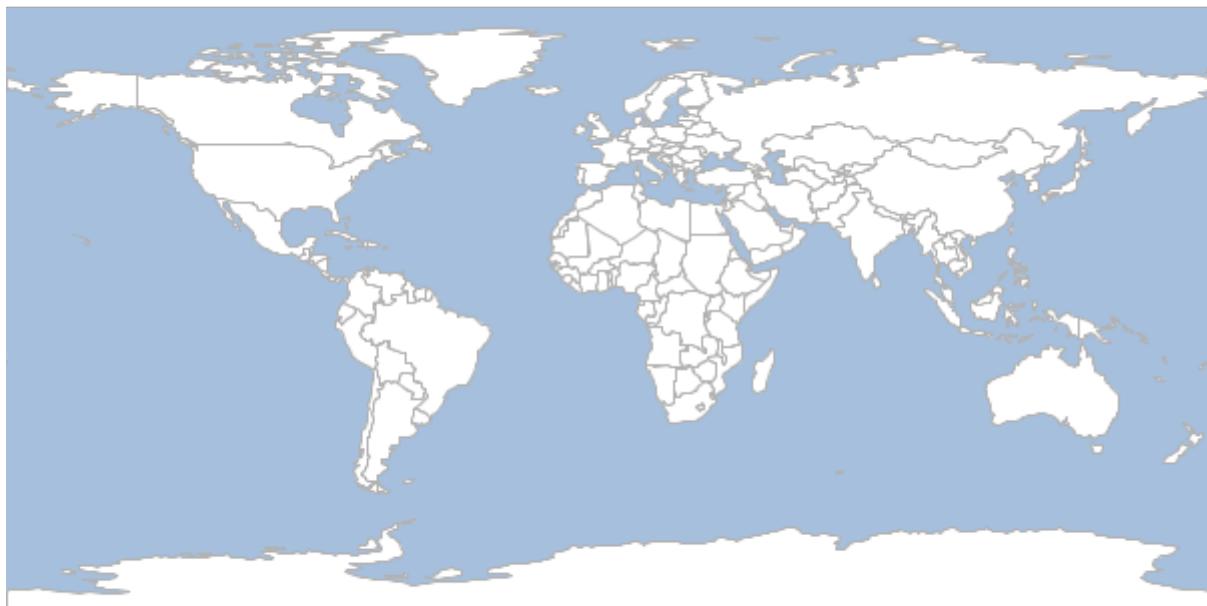
```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map add layer --name world --layer ocean  
Added ocean layer to map world
```

```
geo-shell> map add layer --name world --layer countries  
Added countries layer to map world
```

```
geo-shell> map draw --name world --file examples/map_add_layer.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/map_add_layer.png!
```

```
geo-shell> map close --name world  
Map world closed!
```



Add Raster

Add a Raster Layer.

```
geo-shell> map add raster --name world --raster earth
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		
raster	The raster	true		
mapLayerName	The map layer name	false		

geo-shell> **map open** --name world
Map world opened!

geo-shell> **format open** --name earth --input src/test/resources/earth.tif
Format earth opened!

geo-shell> **raster open** --format earth --raster earth --name earth
Opened Format earth Raster earth as earth

geo-shell> **map add raster** --name world --raster earth
Added earth layer to map world

geo-shell> **workspace open** --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries

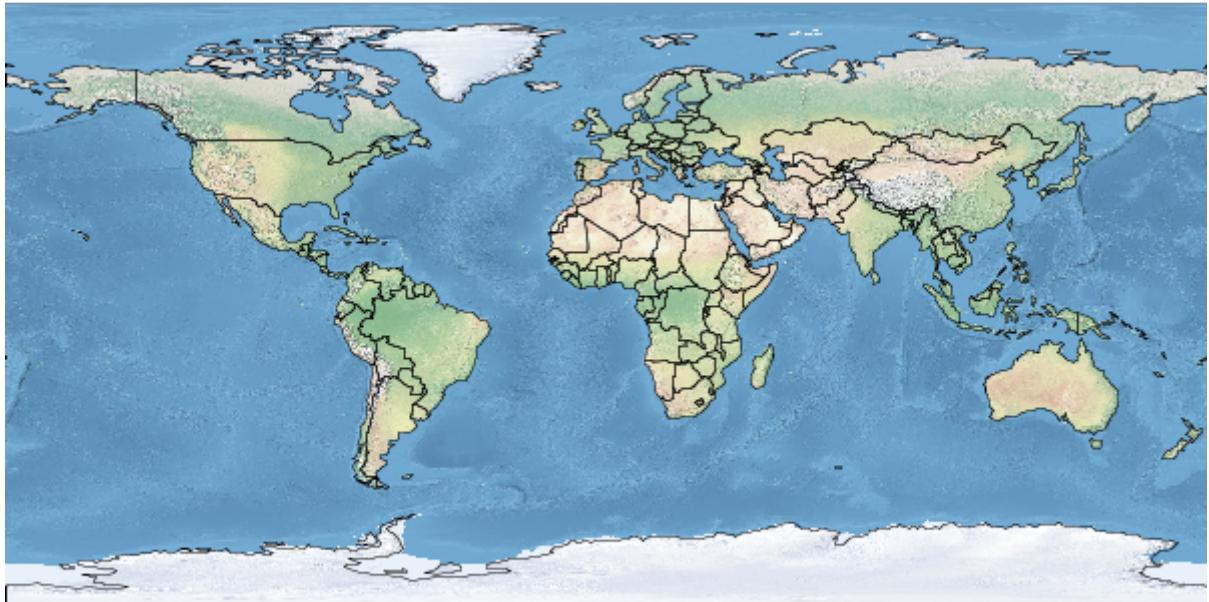
geo-shell> **style create** --params "stroke=black stroke-width=0.1" --file examples/outline.sld
Style stroke=black stroke-width=0.1 written to /home/travis/build/jericks/geo-shell/examples/outline.sld!

geo-shell> **layer style set** --name countries --style examples/outline.sld
Style /home/travis/build/jericks/geo-shell/examples/outline.sld set on countries

geo-shell> **map add layer** --name world --layer countries
Added countries layer to map world

geo-shell> **map draw** --name world --file examples/map_add_raster.png
Done drawing /home/travis/build/jericks/geo-shell/examples/map_add_raster.png!

geo-shell> **map close** --name world
Map world closed!



Add Tile Layer

Add a Tile Layer.

```
geo-shell> map add tile --name world --tile tiles
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		
tile	The tile	true		
mapLayerName	The map layer name	false		

```
geo-shell> map open --name world
```

Map world opened!

```
geo-shell> tile open --name tiles --params src/test/resources/countries.mbtiles
```

Tile Layer tiles opened!

```
geo-shell> map add tile --name world --tile tiles
```

Added tiles layer to map world

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

Workspace naturalearth opened!

```
geo-shell> layer open --workspace naturalearth --layer places --name places
```

Opened Workspace naturalearth Layer places as places

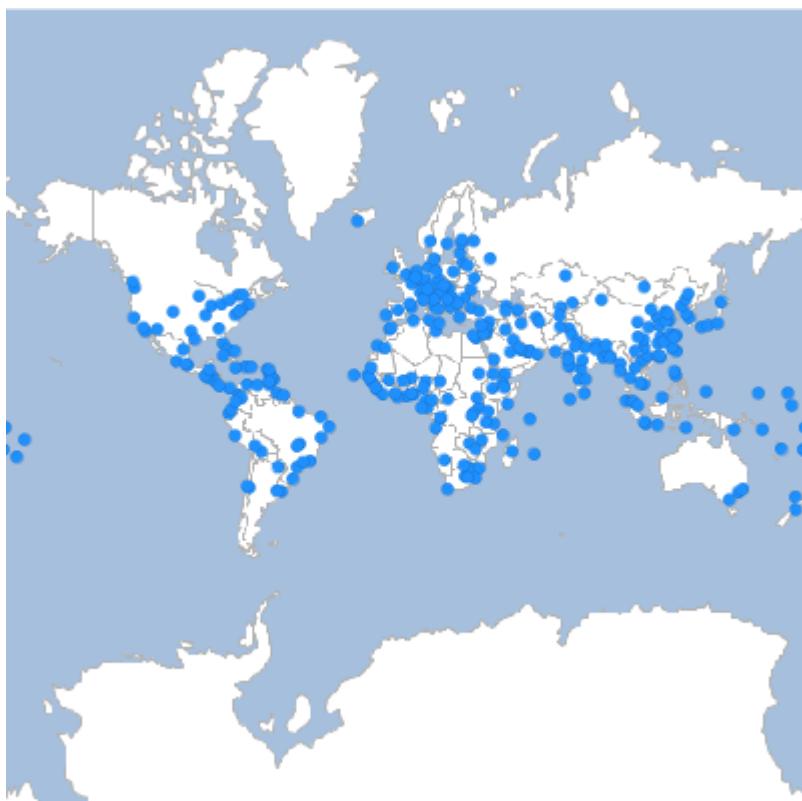
```
geo-shell> style vector default --layer places --color #1E90FF --file examples/places.sld
Default Vector Style for places written to /home/travis/build/jericks/geo-shell/examples/places.sld!
```

```
geo-shell> layer style set --name places --style examples/places.sld
Style /home/travis/build/jericks/geo-shell/examples/places.sld set on places
```

```
geo-shell> map add layer --name world --layer places
Added places layer to map world
```

```
geo-shell> map draw --name world --width 400 --height 400 --file examples/map_add_tile.png
Done drawing /home/travis/build/jericks/geo-shell/examples/map_add_tile.png!
```

```
geo-shell> map close --name world
Map world closed!
```



Remove Layer

Remove a Layer.

```
geo-shell> map remove layer --name world --layer countries
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		
layer	The layer name	true		

```
geo-shell> map open --name world
Map world opened!
```

```
geo-shell> format open --name earth --input src/test/resources/earth.tif
Format earth opened!
```

```
geo-shell> raster open --format earth --raster earth --name earth
Opened Format earth Raster earth as earth
```

```
geo-shell> map add raster --name world --raster earth
Added earth layer to map world
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> style create --params "stroke=black stroke-width=0.1" --file examples/outline.sld
Style      stroke=black      stroke-width=0.1      written      to      /home/travis/build/jericks/geo-
shell/examples/outline.sld!
```

```
geo-shell> layer style set --name countries --style examples/outline.sld
Style /home/travis/build/jericks/geo-shell/examples/outline.sld set on countries
```

```
geo-shell> map add layer --name world --layer countries
Added countries layer to map world
```

```
geo-shell> map layers --name world
earth
countries
```

```
geo-shell> map remove layer --name world --layer countries
Removed countries layer from map world
```

```
geo-shell> map layers --name world
earth
```

```
geo-shell> map close --name world
Map world closed!
```

Reorder

Reorder a Layer in the Map.

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		
layer	The layer name	true		
order	The order parameters	true		

```
geo-shell> map open --name world
```

```
Map world opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

```
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
```

```
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> style create --params "stroke=black stroke-width=0.1" --file examples/outline.sld
```

```
Style      stroke=black      stroke-width=0.1      written      to      /home/travis/build/jericks/geo-  
shell/examples/outline.sld!
```

```
geo-shell> layer style set --name countries --style examples/outline.sld
```

```
Style /home/travis/build/jericks/geo-shell/examples/outline.sld set on countries
```

```
geo-shell> map add layer --name world --layer countries
```

```
Added countries layer to map world
```

```
geo-shell> format open --name earth --input src/test/resources/earth.tif
```

```
Format earth opened!
```

```
geo-shell> raster open --format earth --raster earth --name earth
```

```
Opened Format earth Raster earth as earth
```

```
geo-shell> map add raster --name world --raster earth
```

```
Added earth layer to map world
```

```
geo-shell> map layers --name world
```

```
countries
```

```
earth
```

```
geo-shell> map reorder --name world --layer countries --order 1
```

```
Moved countries from 0 to 1
```

```
geo-shell> map layers --name world
```

```
earth
```

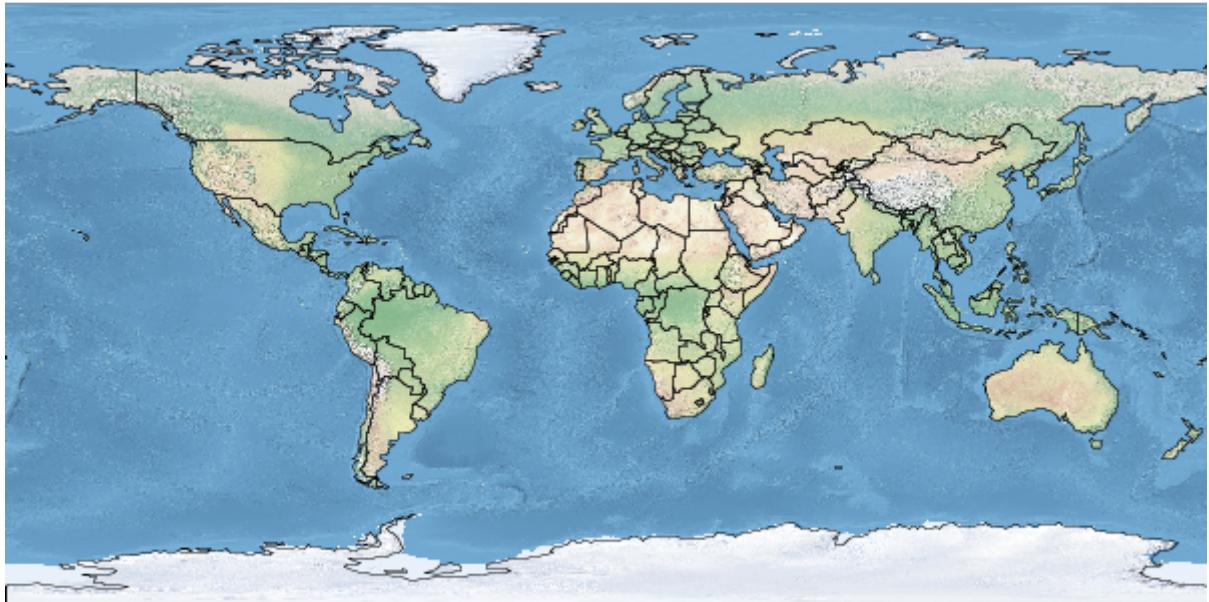
```
countries
```

```
geo-shell> map draw --name world --file examples/map_reordered.png
```

```
Done drawing /home/travis/build/jericks/geo-shell/examples/map_reordered.png!
```

```
geo-shell> map close --name world
```

```
Map world closed!
```



Layers

List the Map's Layers.

```
geo-shell> map layers --name world
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		

```
geo-shell> map open --name world
```

Map world opened!

```
geo-shell> format open --name earth --input src/test/resources/earth.tif
```

Format earth opened!

```
geo-shell> raster open --format earth --raster earth --name earth
```

Opened Format earth Raster earth as earth

```
geo-shell> map add raster --name world --raster earth
```

Added earth layer to map world

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg
```

Workspace naturalearth opened!

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries
```

Opened Workspace naturalearth Layer countries as countries

```
geo-shell> style create --params "stroke=black stroke-width=0.1" --file examples/outline.sld
```

```
Style stroke=black stroke-width=0.1 written to /home/travis/build/jericks/geo-shell/examples/outline.sld!
```

```
geo-shell> layer style set --name countries --style examples/outline.sld  
Style /home/travis/build/jericks/geo-shell/examples/outline.sld set on countries
```

```
geo-shell> map add layer --name world --layer countries  
Added countries layer to map world
```

```
geo-shell> map layers --name world  
earth  
countries
```

```
geo-shell> map close --name world  
Map world closed!
```

Draw

Draw a map.

```
geo-shell> map draw --name world --file examples/map_draw.png
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		
bounds	The Bounds	false		
projection	The Projection	false		
width	The width	false	600	600
height	The height	false	400	400
type	The type	false	png	png
file	The file	false		
background-color	The background color	false		

```
geo-shell> map open --name world  
Map world opened!
```

```
geo-shell> workspace open --name naturalearth --params examples/naturalearth.gpkg  
Workspace naturalearth opened!
```

```
geo-shell> layer open --workspace naturalearth --layer countries --name countries  
Opened Workspace naturalearth Layer countries as countries
```

```
geo-shell> layer style set --name countries --style examples/countries.sld  
Style /home/travis/build/jericks/geo-shell/examples/countries.sld set on countries
```

```
geo-shell> layer open --workspace naturalearth --layer ocean --name ocean  
Opened Workspace naturalearth Layer ocean as ocean
```

```
geo-shell> layer style set --name ocean --style examples/ocean.sld  
Style /home/travis/build/jericks/geo-shell/examples/ocean.sld set on ocean
```

```
geo-shell> map add layer --name world --layer ocean  
Added ocean layer to map world
```

```
geo-shell> map add layer --name world --layer countries  
Added countries layer to map world
```

```
geo-shell> map draw --name world --file examples/map_draw.png  
Done drawing /home/travis/build/jericks/geo-shell/examples/map_draw.png!
```

```
geo-shell> map close --name world  
Map world closed!
```



Display

Display a map in a GUI.

```
geo-shell> map display --name world
```

Name	Description	Mandatory	Specified Default	Unspecified Default
name	The map name	true		
bounds	The Bounds	false		
projection	The Projection	false		
width	The width	false	600	600

height	The height	false	400	400
background-color	The background color	false		

geo-shell> **map open** --name world

Map world opened!

geo-shell> **workspace open** --name naturalearth --params examples/naturalearth.gpkg

Workspace naturalearth opened!

geo-shell> **layer open** --workspace naturalearth --layer countries --name countries

Opened Workspace naturalearth Layer countries as countries

geo-shell> **layer style set** --name countries --style examples/countries.sld

Style /Users/jericks/Projects/geo-shell/examples/countries.sld set on countries

geo-shell> **layer open** --workspace naturalearth --layer ocean --name ocean

Opened Workspace naturalearth Layer ocean as ocean

geo-shell> **layer style set** --name ocean --style examples/ocean.sld

Style /Users/jericks/Projects/geo-shell/examples/ocean.sld set on ocean

geo-shell> **map add layer** --name world --layer ocean

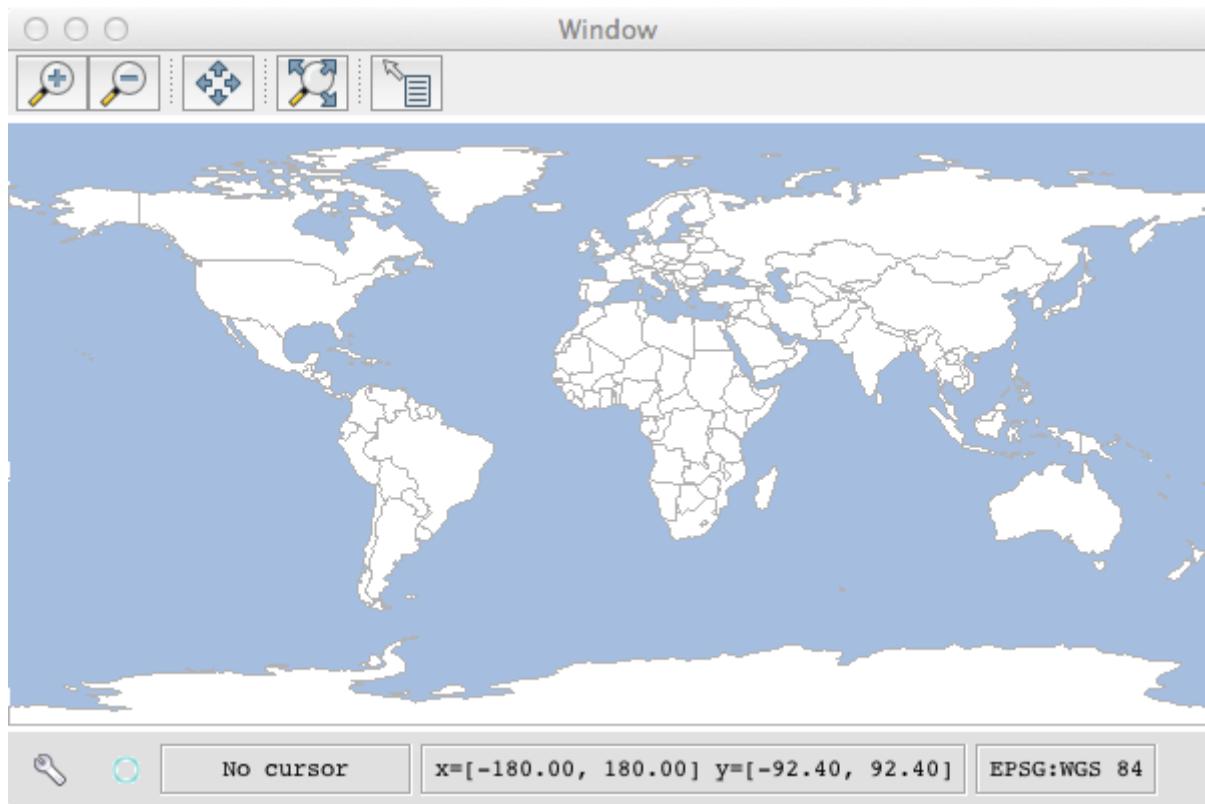
Added ocean layer to map world

geo-shell> **map add layer** --name world --layer countries

Added countries layer to map world

geo-shell> **map display** --name world

Displaying...



Built in

Exit / Quit

Exits the shell

geo-shell> **exit**



No parameters

Help

List all commands usage

Name	Description	Mandatory	Specified Default	Unspecified Default
	Command name to provide help for	false		

View all commands

geo-shell> **help**

- * ! - Allows execution of operating system (OS) commands
- * // - Inline comment markers (start of line only)
- * ; - Inline comment markers (start of line only)
- * clear - Clears the console

- * cls - Clears the console
- * date - Displays the local date and time
- * download - Download a URL to a file.
- * exit - Exits the shell
- * format close - Close a Raster Format.
- * format list - List open Raster Formats.

Get help for a command

```
geo-shell> help layer open
```

Keyword: layer open

Description: Open a Layer.

Keyword: workspace

Help: The Workspace name

Mandatory: true

Default if specified: 'NULL'

Default if unspecified: 'NULL'

Keyword: layer

Help: The Layer name

Mandatory: true

Default if specified: 'NULL'

Default if unspecified: 'NULL'

Keyword: name

Help: The name

Mandatory: false

Default if specified: 'NULL'

Default if unspecified: 'NULL'

* layer open - Open a Layer.

Run OS Command

Allows execution of operating system (OS) commands

```
geo-shell> ! ls src/test/resources/mars
```

Name	Description	Mandatory	Specified Default	Unspecified Default
	The command to execute	false		

```
geo-shell> ! ls src/test/resources/mars
```

```
geo_units_oc_dd.dbf  
geo_units_oc_dd.prj  
geo_units_oc_dd.qix
```

```
geo_units_oc_dd.sbn  
geo_units_oc_dd.sbx  
geo_units_oc_dd.shp  
geo_units_oc_dd.shp.xml  
geo_units_oc_dd.shx  
I1802ABC_geo_units_RGBlut.txt
```

Date

Displays the local date and time

```
geo-shell> date
```



No parameters

```
geo-shell> date
```

Friday, October 9, 2020 11:31:34 PM UTC

Script

Parses the specified resource file and executes its commands

```
geo-shell> script src/test/resources/layer_count.txt
```

Name	Description	Mandatory	Specified Default	Unspecified Default
	The file to locate and execute	true		
lineNumbers	Display line numbers when executing the script	false	true	false

```
workspace open --name naturalearth --params src/test/resources/naturalearth.gpkg  
layer open --workspace naturalearth --layer countries --name countries  
layer count --name countries  
workspace close --name naturalearth
```

```
geo-shell> script src/test/resources/layer_count.txt
```

```
Oct 09, 2020 11:31:31 PM org.geotools.data.ogr.OGRDataStoreFactory isAvailable  
WARNING: Error initializing GDAL/OGR library  
java.lang.UnsatisfiedLinkError: no gdaljni in java.library.path
```

```
at java.lang.ClassLoader.loadLibrary(ClassLoader.java:1867)
at java.lang.Runtime.loadLibrary0(Runtime.java:870)
at java.lang.System.loadLibrary(System.java:1122)
at org.geotools.data.ogr.jni.JniOGRDataStoreFactory.doIsAvailable(JniOGRDataStoreFactory.java:46)
at org.geotools.data.ogr.OGRDataStoreFactory.isAvailable(OGRDataStoreFactory.java:208)
at org.geotools.data.ogr.OGRDataStoreFactory.isAvailable(OGRDataStoreFactory.java:196)
at java.util.stream.ReferencePipeline$2$1.accept(ReferencePipeline.java:174)
at java.util.stream.ReferencePipeline$2$1.accept(ReferencePipeline.java:175)
at java.util.Iterator.forEachRemaining(Iterator.java:116)
at java.util.Spliterators$IteratorSpliterator.forEachRemaining(Spliterators.java:1801)
at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:481)
at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:471)
at java.util.stream.ReduceOps$ReduceOp.evaluateSequential(ReduceOps.java:708)
at java.util.stream.AbstractPipeline.evaluate(AbstractPipeline.java:234)
at java.util.stream.ReferencePipeline.collect(ReferencePipeline.java:499)
at org.geotools.data.DataAccessFinder.getAvailableDataStores(DataAccessFinder.java:199)
at org.geotools.data.DataStoreFinder.getAvailableDataStores(DataStoreFinder.java:94)
at org.geotools.data.DataStoreFinder.getDataStore(DataStoreFinder.java:67)
at org.geotools.data.DataStoreFinder$getDataStore.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at geoscript.workspace.WorkspaceFactory.create(WorkspaceFactory.groovy:32)
at geoscript.workspace.WorkspaceFactory$create$0.callCurrent(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:185)
at geoscript.workspace.WorkspaceFactory.create(WorkspaceFactory.groovy:22)
at geoscript.workspace.WorkspaceFactory$create.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at geoscript.workspace.Workspace.getWorkspace(Workspace.groovy:283)
at geoscript.workspace.Workspace$getWorkspace.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.vector.WorkspaceCommands.open(WorkspaceCommands.groovy:24)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:216)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:216)
```

```
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.docs.AbstractDocTest.run(AbstractDocTest.groovy:52)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:107)
at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:323)
at
org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:35
1)
at org.codehaus.groovy.runtime.callsite.PogoMetaClassSite.callCurrent(PogoMetaClassSite.java:61)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:203)
at org.geoshell.docs.AbstractDocTest$_run_closure1.doCall(AbstractDocTest.groovy:32)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:107)
at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:323)
at
org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:26
3)
at groovy.lang.MetaClassImpl.invokeMethod(MetaClassImpl.java:1026)
at groovy.lang.Closure.call(Closure.java:412)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
234)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
214)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
264)
at org.codehaus.groovy.runtime.dgm$213.invoke(Unknown Source)
at
org.codehaus.groovy.runtime.callsite.PojoMetaMethodSite$PojoMetaMethodSiteNoUnwrapNoCoerc
e.invoke(PojoMetaMethodSite.java:247)
at org.codehaus.groovy.runtime.callsite.PojoMetaMethodSite.call(PojoMetaMethodSite.java:56)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.docs.AbstractDocTest.run(AbstractDocTest.groovy:31)
at org.geoshell.docs.AbstractDocTest$run.callCurrent(Unknown Source)
```

```
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:203)
at org.geoshell.docs.BuiltinDocTest.script(BuiltinDocTest.groovy:66)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.junit.runners.model.FrameworkMethod$1.runReflectiveCall(FrameworkMethod.java:50)
at org.junit.internal.runners.model.ReflectiveCallable.run(ReflectiveCallable.java:12)
at org.junit.runners.model.FrameworkMethod.invokeExplosively(FrameworkMethod.java:47)
at org.junit.internal.runners.statements.InvokeMethod.evaluate(InvokeMethod.java:17)
at org.junit.internal.runners.statements.RunBefores.evaluate(RunBefores.java:26)
at org.junit.internal.runners.statements.RunAfters.evaluate(RunAfters.java:27)
at org.junit.runners.ParentRunner.runLeaf(ParentRunner.java:325)
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:78)
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:57)
at org.junit.runners.ParentRunner$3.run(ParentRunner.java:290)
at org.junit.runners.ParentRunner$1.schedule(ParentRunner.java:71)
at org.junit.runners.ParentRunner.runChildren(ParentRunner.java:288)
at org.junit.runners.ParentRunner.access$000(ParentRunner.java:58)
at org.junit.runners.ParentRunner$2.evaluate(ParentRunner.java:268)
at org.junit.runners.ParentRunner.run(ParentRunner.java:363)
at org.apache.maven.surefire.junit4.JUnit4Provider.execute(JUnit4Provider.java:365)
at org.apache.maven.surefire.junit4.JUnit4Provider.executeWithRerun(JUnit4Provider.java:273)
at org.apache.maven.surefire.junit4.JUnit4Provider.executeTestSet(JUnit4Provider.java:238)
at org.apache.maven.surefire.junit4.JUnit4Provider.invoke(JUnit4Provider.java:159)
at
org.apache.maven.surefire.booter.ForkedBooter.invokeProviderInSameClassLoader(ForkedBooter.java:383)
at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:344)
at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:125)
at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:417)
```

Oct 09, 2020 11:31:31 PM org.geotools.data.ogr.OGRDataStoreFactory isAvailable
WARNING: Error initializing GDAL/OGR library
java.lang.UnsatisfiedLinkError: no gdaljni in java.library.path
at java.lang.ClassLoader.loadLibrary(ClassLoader.java:1867)
at java.lang.Runtime.loadLibrary0(Runtime.java:870)
at java.lang.System.loadLibrary(System.java:1122)
at org.geotools.data.ogr.jni.JniOGRDataStoreFactory.doIsAvailable(JniOGRDataStoreFactory.java:46)
at org.geotools.data.ogr.OGRDataStoreFactory.isAvailable(OGRDataStoreFactory.java:208)
at org.geotools.data.ogr.OGRDataStoreFactory.isAvailable(OGRDataStoreFactory.java:196)
at java.util.stream.ReferencePipeline\$2\$1.accept(ReferencePipeline.java:174)
at java.util.stream.ReferencePipeline\$2\$1.accept(ReferencePipeline.java:175)
at java.util.Iterator.forEachRemaining(Iterator.java:116)
at java.util.Spliterators\$IteratorSpliterator.forEachRemaining(Spliterators.java:1801)
at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:481)

```
at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:471)
at java.util.stream.ReduceOps$ReduceOp.evaluateSequential(ReduceOps.java:708)
at java.util.stream.AbstractPipeline.evaluate(AbstractPipeline.java:234)
at java.util.stream.ReferencePipeline.collect(ReferencePipeline.java:499)
at org.geotools.data.DataAccessFinder.getAvailableDataStores(DataAccessFinder.java:199)
at org.geotools.data.DataStoreFinder.getAvailableDataStores(DataStoreFinder.java:94)
at org.geotools.data.DataStoreFinder.getDataStore(DataStoreFinder.java:67)
at org.geotools.data.DataStoreFinder$getDataStore.call(Unknown Source)
at geoscript.workspace.WorkspaceFactory.create(WorkspaceFactory.groovy:32)
at geoscript.workspace.WorkspaceFactory$create$0.callCurrent(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at geoscript.workspace.WorkspaceFactory$create$0.callCurrent(Unknown Source)
at geoscript.workspace.WorkspaceFactory.create(WorkspaceFactory.groovy:22)
at geoscript.workspace.WorkspaceFactory$create.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at geoscript.workspace.WorkspaceFactory$create.call(Unknown Source)
at geoscript.workspace.Workspace.getWorkspace(Workspace.groovy:283)
at geoscript.workspace.Workspace$getWorkspace.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.vector.WorkspaceCommands.open(WorkspaceCommands.groovy:24)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:216)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:216)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.docs.AbstractDocTest.run(AbstractDocTest.groovy:52)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:107)
at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:323)
at
org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:35
1)
at org.codehaus.groovy.runtime.callsite.PogoMetaClassSite.callCurrent(PogoMetaClassSite.java:61)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)
```

```
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:203)
at org.geoshell.docs.AbstractDocTest$_run_closure1.doCall(AbstractDocTest.groovy:32)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:107)
at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:323)
at
org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:26
3)
at groovy.lang.MetaClassImpl.invokeMethod(MetaClassImpl.java:1026)
at groovy.lang.Closure.call(Closure.java:412)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
234)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
214)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
264)
at org.codehaus.groovy.runtime.dgm$213.invoke(Unknown Source)
at
org.codehaus.groovy.runtime.callsite.PojoMetaMethodSite$PojoMetaMethodSiteNoUnwrapNoCoerc
e.invoke(PojoMetaMethodSite.java:247)
at org.codehaus.groovy.runtime.callsite.PojoMetaMethodSite.call(PojoMetaMethodSite.java:56)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.docs.AbstractDocTest.run(AbstractDocTest.groovy:31)
at org.geoshell.docs.AbstractDocTest$run.callCurrent(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:203)
at org.geoshell.docs.BuiltinDocTest.script(BuiltinDocTest.groovy:66)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.junit.runners.model.FrameworkMethod$1.runReflectiveCall(FrameworkMethod.java:50)
at org.junit.internal.runners.model.ReflectiveCallable.run(ReflectiveCallable.java:12)
at org.junit.runners.model.FrameworkMethod.invokeExplosively(FrameworkMethod.java:47)
at org.junit.internal.runners.statements.InvokeMethod.evaluate(InvokeMethod.java:17)
at org.junit.internal.runners.statements.RunBefores.evaluate(RunBefores.java:26)
at org.junit.internal.runners.statements.RunAfters.evaluate(RunAfters.java:27)
at org.junit.runners.ParentRunner.runLeaf(ParentRunner.java:325)
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:78)
```

```
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:57)
at org.junit.runners.ParentRunner$3.run(ParentRunner.java:290)
at org.junit.runners.ParentRunner$1.schedule(ParentRunner.java:71)
at org.junit.runners.ParentRunner.runChildren(ParentRunner.java:288)
at org.junit.runners.ParentRunner.access$000(ParentRunner.java:58)
at org.junit.runners.ParentRunner$2.evaluate(ParentRunner.java:268)
at org.junit.runners.ParentRunner.run(ParentRunner.java:363)
at org.apache.maven.surefire.junit4.JUnit4Provider.execute(JUnit4Provider.java:365)
at org.apache.maven.surefire.junit4.JUnit4Provider.executeWithRerun(JUnit4Provider.java:273)
at org.apache.maven.surefire.junit4.JUnit4Provider.executeTestSet(JUnit4Provider.java:238)
at org.apache.maven.surefire.junit4.JUnit4Provider.invoke(JUnit4Provider.java:159)
at
org.apache.maven.surefire.booter.ForkedBooter.invokeProviderInSameClassLoader(ForkedBooter.j
ava:383)
at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:344)
at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:125)
at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:417)
```

Oct 09, 2020 11:31:31 PM org.geotools.data.ogr.OGRDataStoreFactory isAvailable
WARNING: Error initializing GDAL/OGR library
java.lang.UnsatisfiedLinkError: no gdaljni in java.library.path
at java.lang.ClassLoader.loadLibrary(ClassLoader.java:1867)
at java.lang.Runtime.loadLibrary0(Runtime.java:870)
at java.lang.System.loadLibrary(System.java:1122)
at org.geotools.data.ogr.jni.JniOGRDataStoreFactory.doIsAvailable(JniOGRDataStoreFactory.java:46)
at org.geotools.data.ogr.OGRDataStoreFactory.isAvailable(OGRDataStoreFactory.java:208)
at org.geotools.data.ogr.OGRDataStoreFactory.isAvailable(OGRDataStoreFactory.java:196)
at java.util.stream.ReferencePipeline\$2\$1.accept(ReferencePipeline.java:174)
at java.util.stream.ReferencePipeline\$2\$1.accept(ReferencePipeline.java:175)
at java.util.Iterator.forEachRemaining(Iterator.java:116)
at java.util.Spliterators\$IteratorSpliterator.forEachRemaining(Spliterators.java:1801)
at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:481)
at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:471)
at java.util.stream.ReduceOps\$ReduceOp.evaluateSequential(ReduceOps.java:708)
at java.util.stream.AbstractPipeline.evaluate(AbstractPipeline.java:234)
at java.util.stream.ReferencePipeline.collect(ReferencePipeline.java:499)
at org.geotools.data.DataAccessFinder.getAvailableDataStores(DataAccessFinder.java:199)
at org.geotools.data.DataStoreFinder.getAvailableDataStores(DataStoreFinder.java:94)
at org.geotools.data.DataStoreFinder.getDataStore(DataStoreFinder.java:67)
at org.geotools.data.DataStoreFinder\$getDataStore.call(Unknown Source)
at geoscript.workspace.WorkspaceFactory.create(WorkspaceFactory.groovy:32)
at geoscript.workspace.WorkspaceFactory\$create\$0.callCurrent(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at geoscript.workspace.WorkspaceFactory\$create\$0.callCurrent(Unknown Source)
at geoscript.workspace.WorkspaceFactory.create(WorkspaceFactory.groovy:22)
at geoscript.workspace.WorkspaceFactory\$create.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at geoscript.workspace.WorkspaceFactory\$create.call(Unknown Source)

```
at geoscript.workspace.Workspace.getWorkspace(Workspace.groovy:283)
at geoscript.workspace.Workspace$getWorkspace.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.vector.WorkspaceCommands.open(WorkspaceCommands.groovy:24)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:216)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:216)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.docs.AbstractDocTest.run(AbstractDocTest.groovy:52)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:107)
at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:323)
at
org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:35
1)
at org.codehaus.groovy.runtime.callsite.PogoMetaClassSite.callCurrent(PogoMetaClassSite.java:61)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:203)
at org.geoshell.docs.AbstractDocTest$_run_closure1.doCall(AbstractDocTest.groovy:32)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:107)
at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:323)
at
org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:26
3)
at groovy.lang.MetaClassImpl.invokeMethod(MetaClassImpl.java:1026)
at groovy.lang.Closure.call(Closure.java:412)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
234)
```

```
at  
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2  
214)  
at  
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2  
264)  
at org.codehaus.groovy.runtime.dgm$213.invoke(Unknown Source)  
at  
org.codehaus.groovy.runtime.callsite.PojoMetaMethodSite$PojoMetaMethodSiteNoUnwrapNoCoerc  
e.invoke(PojoMetaMethodSite.java:247)  
at org.codehaus.groovy.runtime.callsite.PojoMetaMethodSite.call(PojoMetaMethodSite.java:56)  
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)  
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)  
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)  
at org.geoshell.docs.AbstractDocTest.run(AbstractDocTest.groovy:31)  
at org.geoshell.docs.AbstractDocTest$run.callCurrent(Unknown Source)  
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)  
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)  
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:203)  
at org.geoshell.docs.BuiltinDocTest.script(BuiltinDocTest.groovy:66)  
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)  
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)  
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)  
at java.lang.reflect.Method.invoke(Method.java:498)  
at org.junit.runners.model.FrameworkMethod$1.runReflectiveCall(FrameworkMethod.java:50)  
at org.junit.internal.runners.model.ReflectiveCallable.run(ReflectiveCallable.java:12)  
at org.junit.runners.model.FrameworkMethod.invokeExplosively(FrameworkMethod.java:47)  
at org.junit.internal.runners.statements.InvokeMethod.evaluate(InvokeMethod.java:17)  
at org.junit.internal.runners.statements.RunBefores.evaluate(RunBefores.java:26)  
at org.junit.internal.runners.statements.RunAfters.evaluate(RunAfters.java:27)  
at org.junit.runners.ParentRunner.runLeaf(ParentRunner.java:325)  
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:78)  
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:57)  
at org.junit.runners.ParentRunner$3.run(ParentRunner.java:290)  
at org.junit.runners.ParentRunner$1.schedule(ParentRunner.java:71)  
at org.junit.runners.ParentRunner.runChildren(ParentRunner.java:288)  
at org.junit.runners.ParentRunner.access$000(ParentRunner.java:58)  
at org.junit.runners.ParentRunner$2.evaluate(ParentRunner.java:268)  
at org.junit.runners.ParentRunner.run(ParentRunner.java:363)  
at org.apache.maven.surefire.junit4.JUnit4Provider.execute(JUnit4Provider.java:365)  
at org.apache.maven.surefire.junit4.JUnit4Provider.executeWithRerun(JUnit4Provider.java:273)  
at org.apache.maven.surefire.junit4.JUnit4Provider.executeTestSet(JUnit4Provider.java:238)  
at org.apache.maven.surefire.junit4.JUnit4Provider.invoke(JUnit4Provider.java:159)  
at  
org.apache.maven.surefire.booter.ForkedBooter.invokeProviderInSameClassLoader(ForkedBooter.j  
ava:383)  
at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:344)  
at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:125)
```

at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:417)

Oct 09, 2020 11:31:31 PM org.geotools.data.ogr.OGRDataStoreFactory isAvailable
WARNING: Error initializing GDAL/OGR library
java.lang.UnsatisfiedLinkError: no gdaljni in java.library.path
at java.lang.ClassLoader.loadLibrary(ClassLoader.java:1867)
at java.lang.Runtime.loadLibrary0(Runtime.java:870)
at java.lang.System.loadLibrary(System.java:1122)
at org.geotools.data.ogr.jni.JniOGRDataStoreFactory.doIsAvailable(JniOGRDataStoreFactory.java:46)
at org.geotools.data.ogr.OGRDataStoreFactory.isAvailable(OGRDataStoreFactory.java:208)
at org.geotools.data.ogr.OGRDataStoreFactory.isAvailable(OGRDataStoreFactory.java:196)
at java.util.stream.ReferencePipeline\$2\$1.accept(ReferencePipeline.java:174)
at java.util.stream.ReferencePipeline\$2\$1.accept(ReferencePipeline.java:175)
at java.util.Iterator.forEachRemaining(Iterator.java:116)
at java.util.Spliterators\$IteratorSpliterator.forEachRemaining(Spliterators.java:1801)
at java.util.stream.AbstractPipeline.copyInto(AbstractPipeline.java:481)
at java.util.stream.AbstractPipeline.wrapAndCopyInto(AbstractPipeline.java:471)
at java.util.stream.ReduceOps\$ReduceOp.evaluateSequential(ReduceOps.java:708)
at java.util.stream.AbstractPipeline.evaluate(AbstractPipeline.java:234)
at java.util.stream.ReferencePipeline.collect(ReferencePipeline.java:499)
at org.geotools.data.DataAccessFinder.getAvailableDataStores(DataAccessFinder.java:199)
at org.geotools.data.DataStoreFinder.getAvailableDataStores(DataStoreFinder.java:94)
at org.geotools.data.DataStoreFinder.getDataStore(DataStoreFinder.java:67)
at org.geotools.data.DataStoreFinder\$getDataStore.call(Unknown Source)
at geoscript.workspace.WorkspaceFactory.create(WorkspaceFactory.groovy:32)
at geoscript.workspace.WorkspaceFactory\$create\$0.callCurrent(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at geoscript.workspace.WorkspaceFactory\$create\$0.callCurrent(Unknown Source)
at geoscript.workspace.WorkspaceFactory.create(WorkspaceFactory.groovy:22)
at geoscript.workspace.WorkspaceFactory\$create.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at geoscript.workspace.WorkspaceFactory\$create.call(Unknown Source)
at geoscript.workspace.Workspace.getWorkspace(Workspace.groovy:283)
at geoscript.workspace.Workspace\$getWorkspace.call(Unknown Source)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.vector.WorkspaceCommands.open(WorkspaceCommands.groovy:24)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:216)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.springframework.util.ReflectionUtils.invokeMethod(ReflectionUtils.java:216)

```
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.docs.AbstractDocTest.run(AbstractDocTest.groovy:52)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:107)
at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:323)
at
org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:35
1)
at org.codehaus.groovy.runtime.callsite.PogoMetaClassSite.callCurrent(PogoMetaClassSite.java:61)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:203)
at org.geoshell.docs.AbstractDocTest$_run_closure1.doCall(AbstractDocTest.groovy:32)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.codehaus.groovy.reflection.CachedMethod.invoke(CachedMethod.java:107)
at groovy.lang.MetaMethod.doMethodInvoke(MetaMethod.java:323)
at
org.codehaus.groovy.runtime.metaclass.ClosureMetaClass.invokeMethod(ClosureMetaClass.java:26
3)
at groovy.lang.MetaClassImpl.invokeMethod(MetaClassImpl.java:1026)
at groovy.lang.Closure.call(Closure.java:412)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
234)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
214)
at
org.codehaus.groovy.runtime.DefaultGroovyMethods.eachWithIndex(DefaultGroovyMethods.java:2
264)
at org.codehaus.groovy.runtime.dgm$213.invoke(Unknown Source)
at
org.codehaus.groovy.runtime.callsite.PojoMetaMethodSite$PojoMetaMethodSiteNoUnwrapNoCoerc
e.invoke(PojoMetaMethodSite.java:247)
at org.codehaus.groovy.runtime.callsite.PojoMetaMethodSite.call(PojoMetaMethodSite.java:56)
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCall(CallSiteArray.java:47)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:125)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.call(AbstractCallSite.java:139)
at org.geoshell.docs.AbstractDocTest.run(AbstractDocTest.groovy:31)
at org.geoshell.docs.AbstractDocTest$run.callCurrent(Unknown Source)
```

```
at org.codehaus.groovy.runtime.callsite.CallSiteArray.defaultCallCurrent(CallSiteArray.java:51)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:171)
at org.codehaus.groovy.runtime.callsite.AbstractCallSite.callCurrent(AbstractCallSite.java:203)
at org.geoshell.docs.BuiltinDocTest.script(BuiltinDocTest.groovy:66)
at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
at java.lang.reflect.Method.invoke(Method.java:498)
at org.junit.runners.model.FrameworkMethod$1.runReflectiveCall(FrameworkMethod.java:50)
at org.junit.internal.runners.model.ReflectiveCallable.run(ReflectiveCallable.java:12)
at org.junit.runners.model.FrameworkMethod.invokeExplosively(FrameworkMethod.java:47)
at org.junit.internal.runners.statements.InvokeMethod.evaluate(InvokeMethod.java:17)
at org.junit.internal.runners.statements.RunBefores.evaluate(RunBefores.java:26)
at org.junit.internal.runners.statements.RunAfters.evaluate(RunAfters.java:27)
at org.junit.runners.ParentRunner.runLeaf(ParentRunner.java:325)
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:78)
at org.junit.runners.BlockJUnit4ClassRunner.runChild(BlockJUnit4ClassRunner.java:57)
at org.junit.runners.ParentRunner$3.run(ParentRunner.java:290)
at org.junit.runners.ParentRunner$1.schedule(ParentRunner.java:71)
at org.junit.runners.ParentRunner.runChildren(ParentRunner.java:288)
at org.junit.runners.ParentRunner.access$000(ParentRunner.java:58)
at org.junit.runners.ParentRunner$2.evaluate(ParentRunner.java:268)
at org.junit.runners.ParentRunner.run(ParentRunner.java:363)
at org.apache.maven.surefire.junit4.JUnit4Provider.execute(JUnit4Provider.java:365)
at org.apache.maven.surefire.junit4.JUnit4Provider.executeWithRerun(JUnit4Provider.java:273)
at org.apache.maven.surefire.junit4.JUnit4Provider.executeTestSet(JUnit4Provider.java:238)
at org.apache.maven.surefire.junit4.JUnit4Provider.invoke(JUnit4Provider.java:159)
at
org.apache.maven.surefire.booter.ForkedBooter.invokeProviderInSameClassLoader(ForkedBooter.java:383)
at org.apache.maven.surefire.booter.ForkedBooter.runSuitesInProcess(ForkedBooter.java:344)
at org.apache.maven.surefire.booter.ForkedBooter.execute(ForkedBooter.java:125)
at org.apache.maven.surefire.booter.ForkedBooter.main(ForkedBooter.java:417)
```

Workspace naturalearth opened!

Opened Workspace naturalearth Layer countries as countries

177

Workspace naturalearth closed!

System Properties

Shows the shell's properties

geo-shell> **system properties**



No parameters

geo-shell> **system properties**

```
awt.toolkit = sun.awt.X11.XToolkit  
basedir = /home/travis/build/jericks/geo-shell  
file.encoding = UTF-8  
file.encoding.pkg = sun.io  
file.separator = /
```

Version

Displays shell version



No parameters

```
geo-shell> version
```

0.8.0-SNAPSHOT

Download

Download a URL to a file.

```
geo-shell> download --url https://astropedia.astrogeology.usgs.gov/download/Mars/Geology/Mars15MGeologicGISRenovation.zip --file mars.zip --overwrite false
```

Name	Description	Mandatory	Specified Default	Unspecified Default
url	The url	true		
file	The file	true		
overwrite	Whether to overwrite the file or not	false	true	true

```
geo-shell> download --url https://astropedia.astrogeology.usgs.gov/download/Mars/Geology/Mars15MGeologicGISRenovation.zip --file mars.zip --overwrite false
```

```
Downloading https://astropedia.astrogeology.usgs.gov/download/Mars/Geology/Mars15MGeologicGISRenovation.zip to /Users/jericks/Projects/geo-shell/mars.zip...
```

```
geo-shell> unzip --file mars.zip --directory mars
```

```
Unzipping /Users/jericks/Projects/geo-shell/mars.zip to /Users/jericks/Projects/geo-shell/mars
```

```
geo-shell> style vector uniquevaluesfromtext --field UnitSymbol --geometryType Polygon  
--styleFile mars/units.sld --textFile mars/I1802ABC_Mars_global_geology/I1802ABC_geo_units_RGBlut.txt  
Create a unique values style from /Users/jericks/Projects/geo-shell/mars/I1802ABC_Mars_global_geology/I1802ABC_geo_units_RGBlut.txt for UnitSymbol and Polygon to /Users/jericks/Projects/geo-shell/mars/units.sld
```

```
geo-shell> workspace open --name mars --params  
mars/I1802ABC_Mars_global_geology/Shapefiles/I1802ABC_Mars2000_Sphere/geo_units_oc_dd.shp
```

Workspace mars opened!

```
geo-shell> layer open --workspace mars --layer geo_units_oc_dd
```

```
Opened Workspace mars Layer geo_units_oc_dd as mars:geo_units_oc_dd
```

```
geo-shell> layer style set --name mars:geo_units_oc_dd --style mars/units.sld
```

```
Style /Users/jericks/Projects/geo-shell/mars/units.sld set on mars:geo_units_oc_dd
```

```
geo-shell> map open --name mars
```

```
Map mars opened!
```

```
geo-shell> map add layer --name mars --layer mars:geo_units_oc_dd
```

```
Added mars:geo_units_oc_dd layer to map mars
```

```
geo-shell> map draw --name mars
```

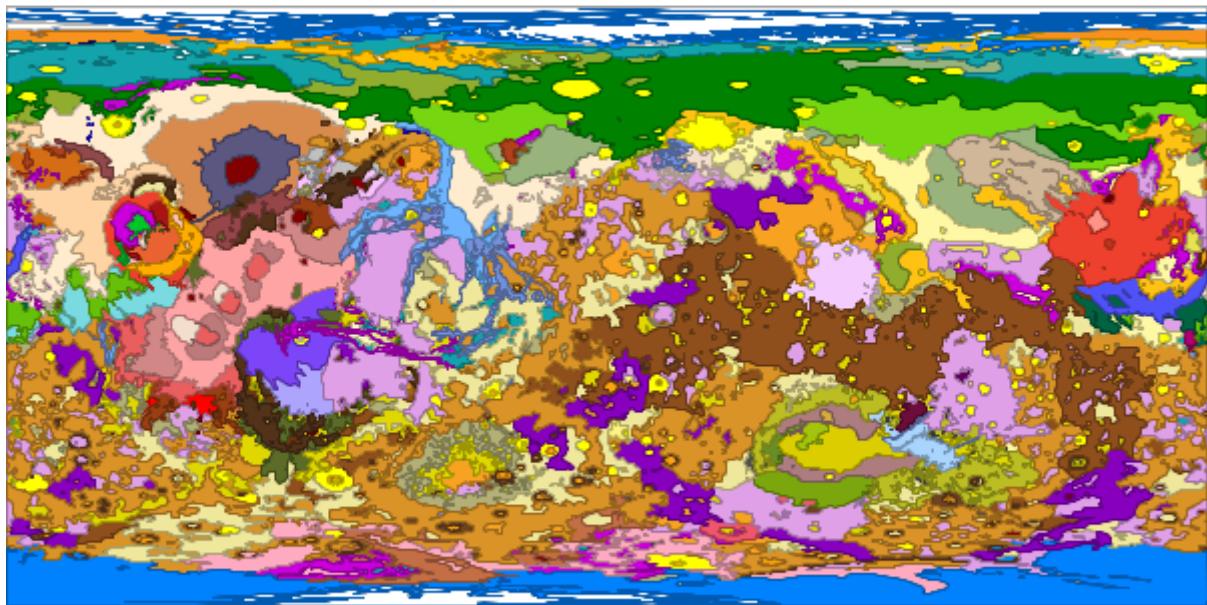
```
Done drawing /Users/jericks/Projects/geo-shell/image.png!
```

```
geo-shell> map close --name mars
```

```
Map mars closed!
```

```
geo-shell> open --file image.png
```

```
Opening /Users/jericks/Projects/geo-shell/image.png...
```



Unzip

Unzip a file

```
geo-shell> unzip --file mars.zip --directory mars
```

Name	Description	Mandatory	Specified Default	Unspecified Default
file	The zip file	true		
directory	The directory	true		

```
geo-shell> unzip --file mars.zip --directory mars
Unzipping /Users/jericks/Projects/geo-shell/mars.zip to /Users/jericks/Projects/geo-shell/mars
```

Open

Open a File.

```
geo-shell> open --file image.png
```

Name	Description	Mandatory	Specified Default	Unspecified Default
file	The File	true		

```
geo-shell> open --file image.png
Opening /Users/jericks/Projects/geo-shell/image.png...
```