Table of Contents

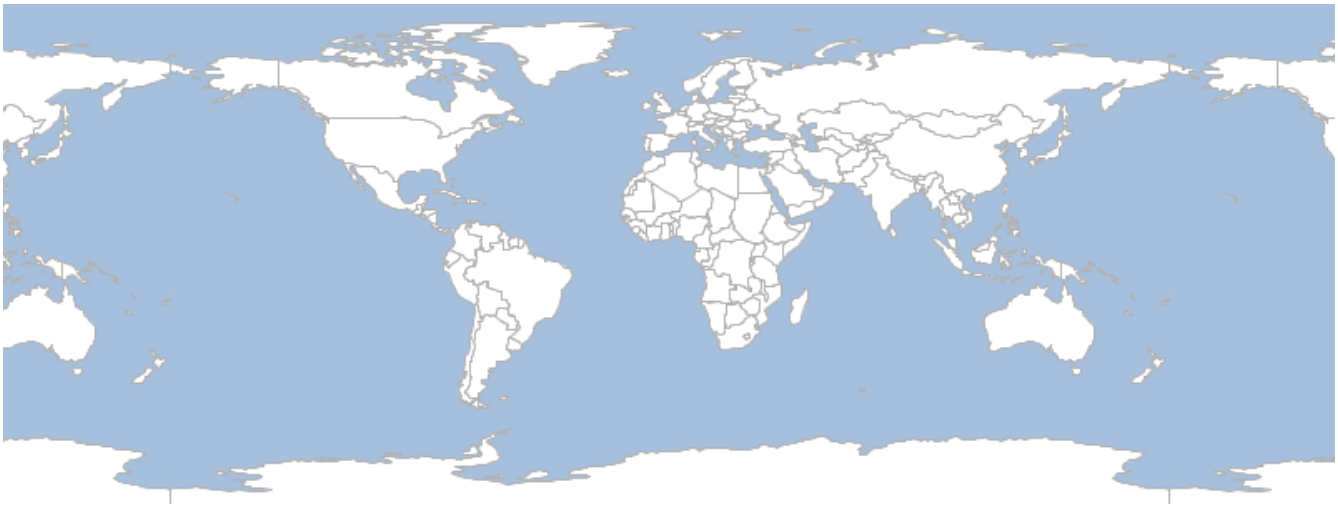
Render Recipes

The Render classes are in the **geoscript.render** package.

Creating Maps

Create a Map with Layers

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
    width: 800,
    height: 300,
    layers: [ocean, countries]
)
File file = new File("map.png")
map.render(file)
```



Rendering Maps

Finding Renderers

Get all Renderers

```
List<Renderer> renderers = Renderers.list()
renderers.each { Renderer renderer ->
    println renderer.class.simpleName
}
```

```
ASCII
Base64
GeoTIFF
GIF
JPEG
Pdf
PNG
Svg
```

Get a Renderer

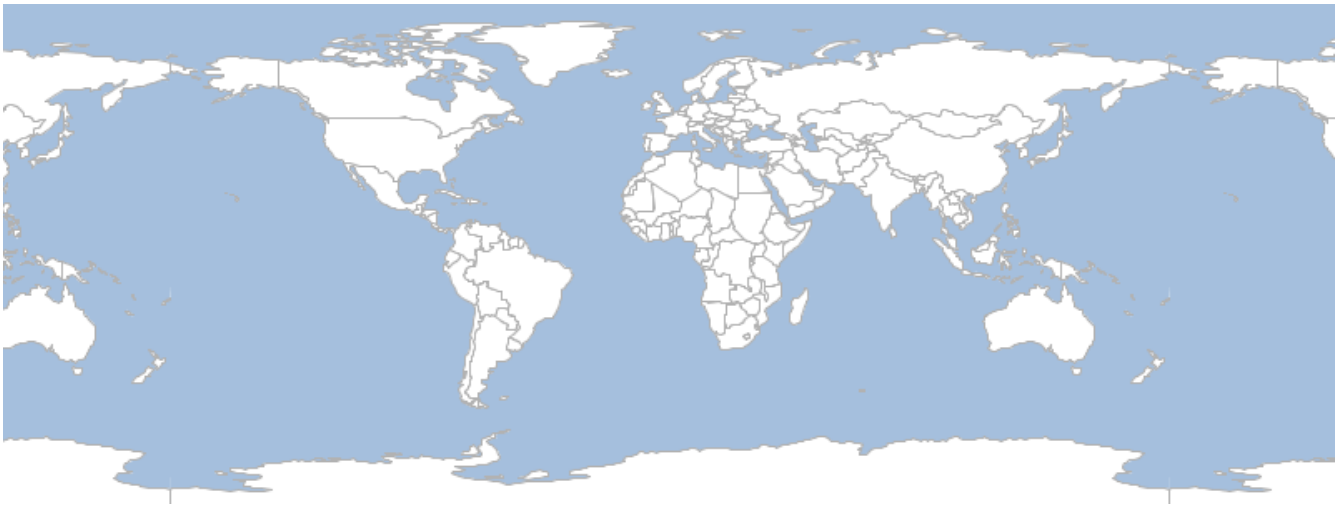
```
Renderer renderer = Renderers.find("png")
println renderer.class.simpleName
```

```
PNG
```

Image

Render a Map to an image using an Image Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Image png = new Image("png")
BufferedImage image = png.render(map)
```



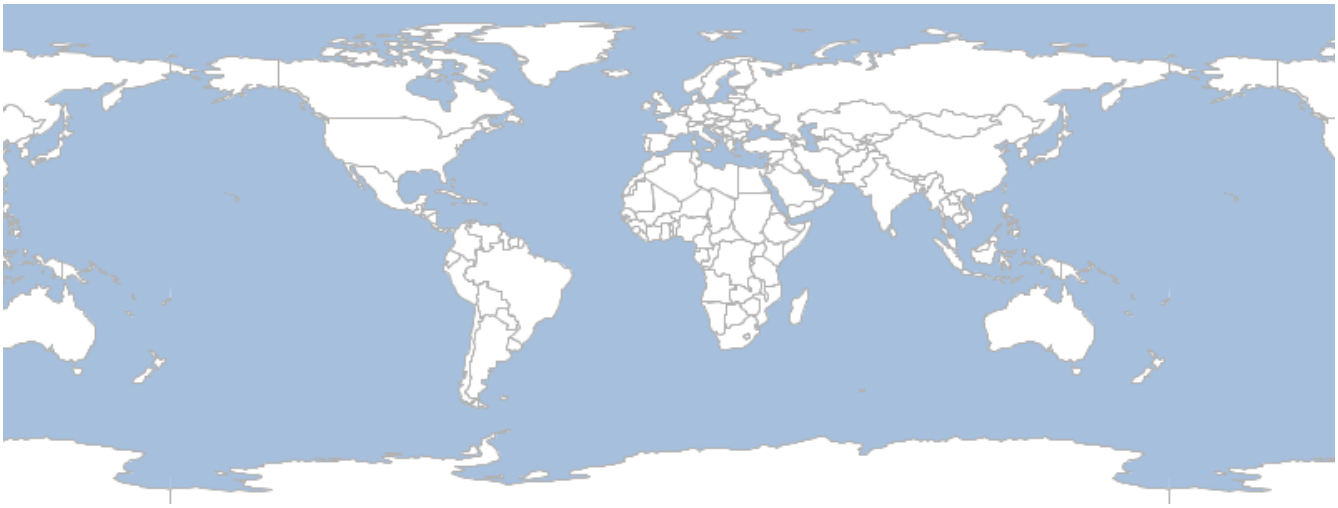
Render a Map to an OutputStream using the Image Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Image jpeg = new Image("jpeg")
File file = new File("map.jpeg")
FileOutputStream out = new FileOutputStream(file)
jpeg.render(map, out)
out.close()
```

PNG

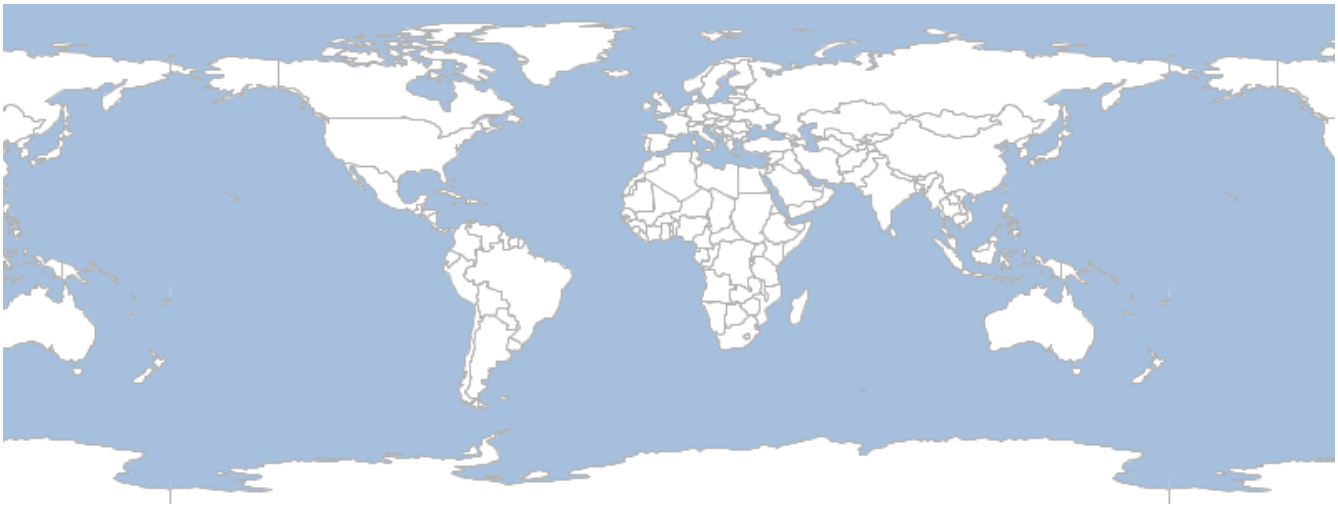
Render a Map to an Image using the PNG Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
    width: 800,
    height: 300,
    layers: [ocean, countries]
)
PNG png = new PNG()
BufferedImage image = png.render(map)
```



Render a Map to an OutputStream using the PNG Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
    width: 800,
    height: 300,
    layers: [ocean, countries]
)
PNG png = new PNG()
File file = new File("map.png")
FileOutputStream out = new FileOutputStream(file)
png.render(map, out)
out.close()
```



JPEG

Render a Map to an Image using the JPEG Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
JPEG jpeg = new JPEG()
BufferedImage image = jpeg.render(map)
```

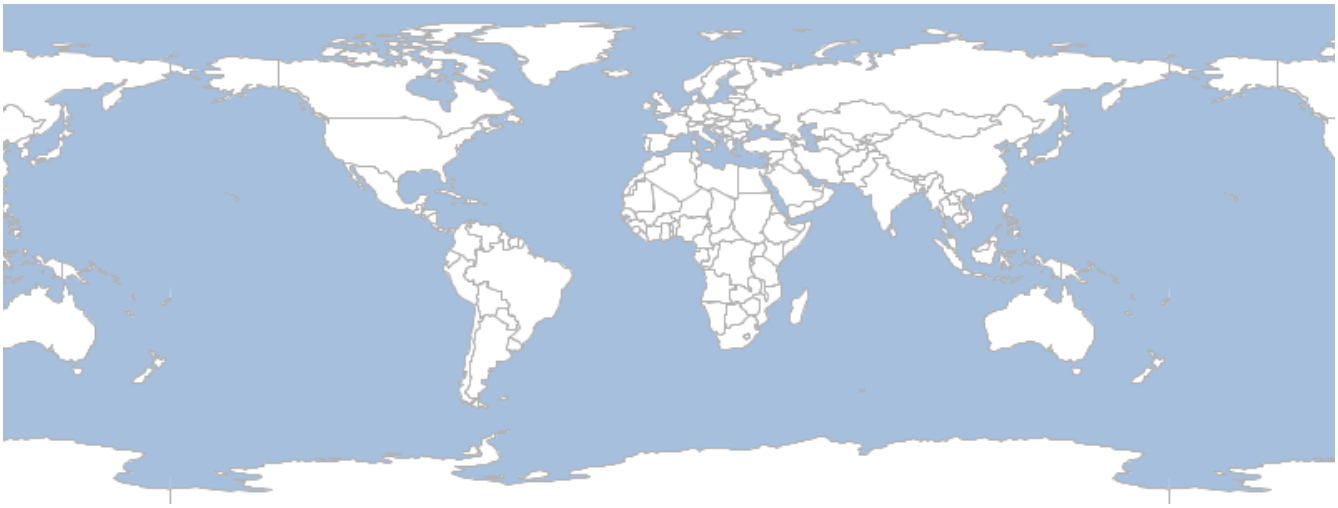
Render a Map to an OutputStream using the JPEG Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
JPEG jpeg = new JPEG()
File file = new File("map.jpeg")
FileOutputStream out = new FileOutputStream(file)
jpeg.render(map, out)
out.close()
```

GIF

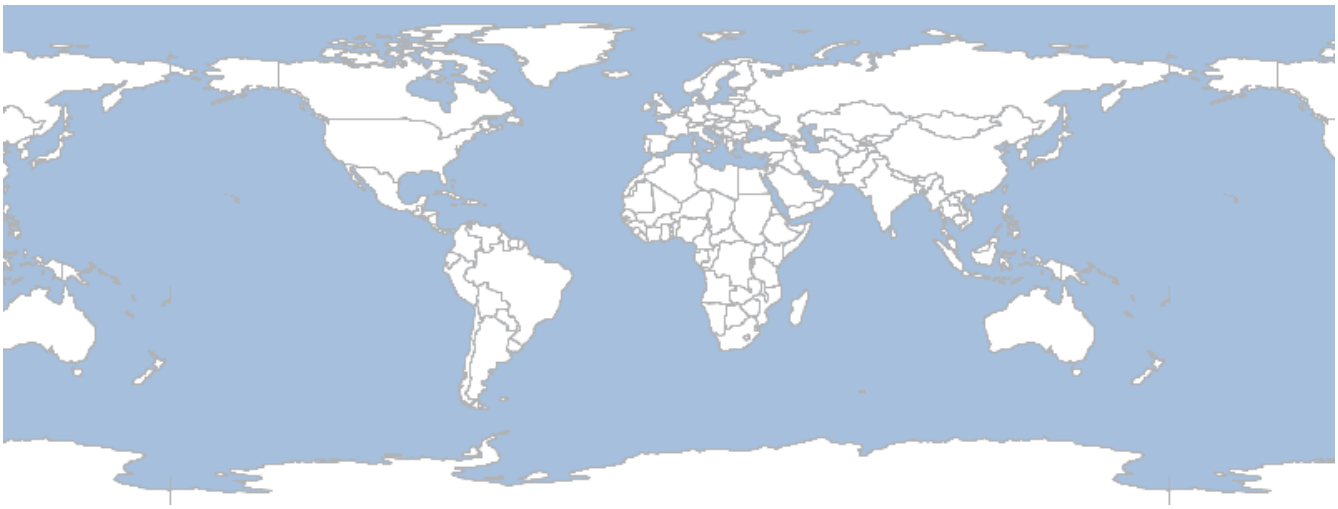
Render a Map to an Image using the GIF Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
GIF gif = new GIF()
BufferedImage image = gif.render(map)
```



Render a Map to an OutputStream using the GIF Renderer

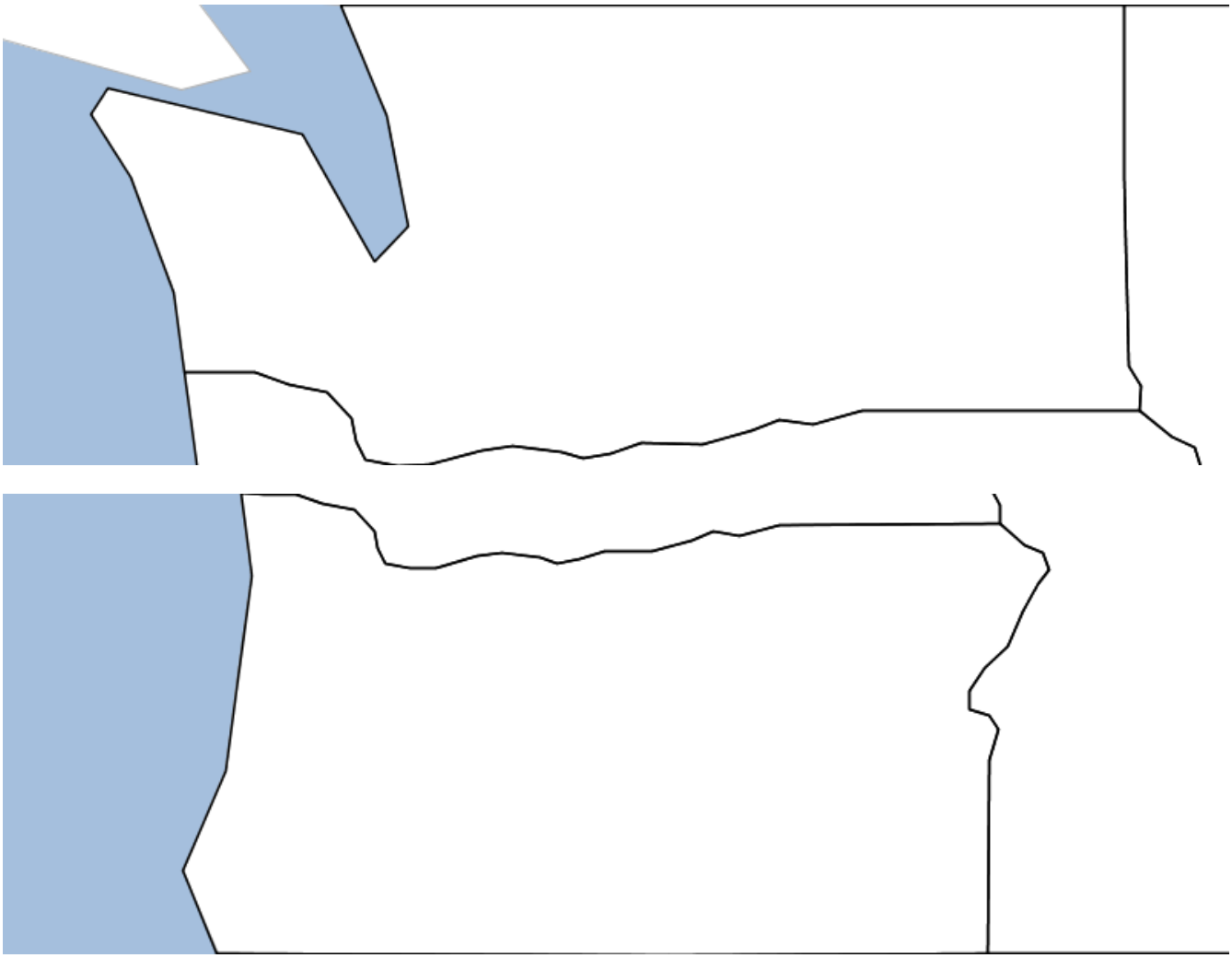
```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
GIF gif = new GIF()
File file = new File("map.gif")
gif.render(map, new FileOutputStream(file))
```

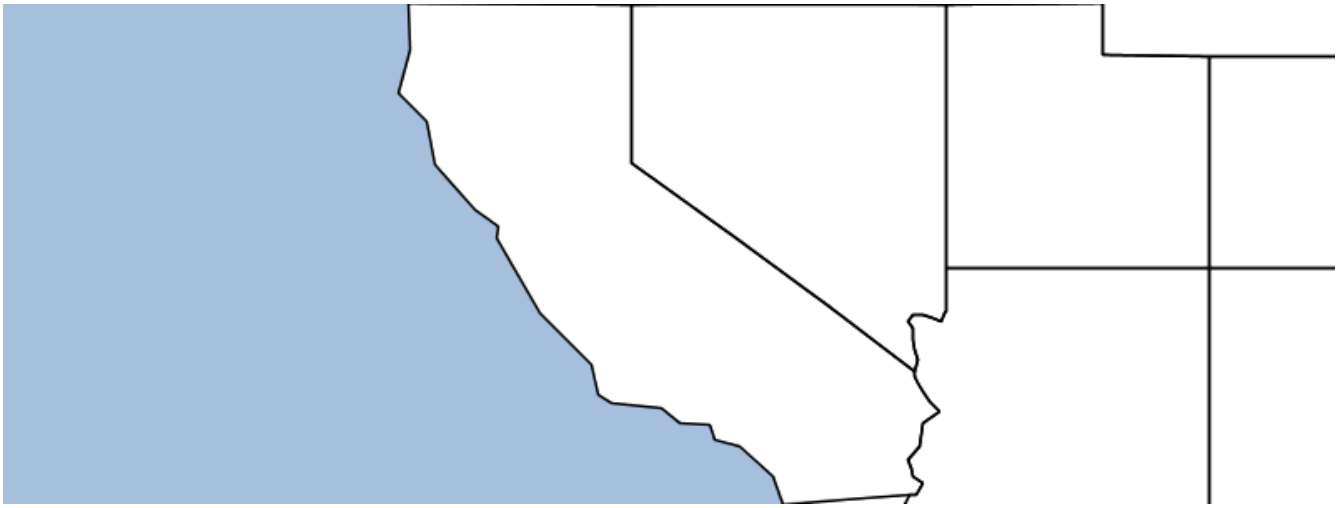


Render a Map to an animated GIF using the GIF Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer states = workspace.get("states")
states.style = new Fill("") + new Stroke("black", 1.0)
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries, states]
)

GIF gif = new GIF()
List images = ["Washington","Oregon","California"].collect { String state ->
    map.bounds = states.getFeatures("NAME_1 = '${state}'")[0].bounds
    def image = gif.render(map)
    image
}
File file = new File("states.gif")
gif.renderAnimated(images, file, 500, true)
```

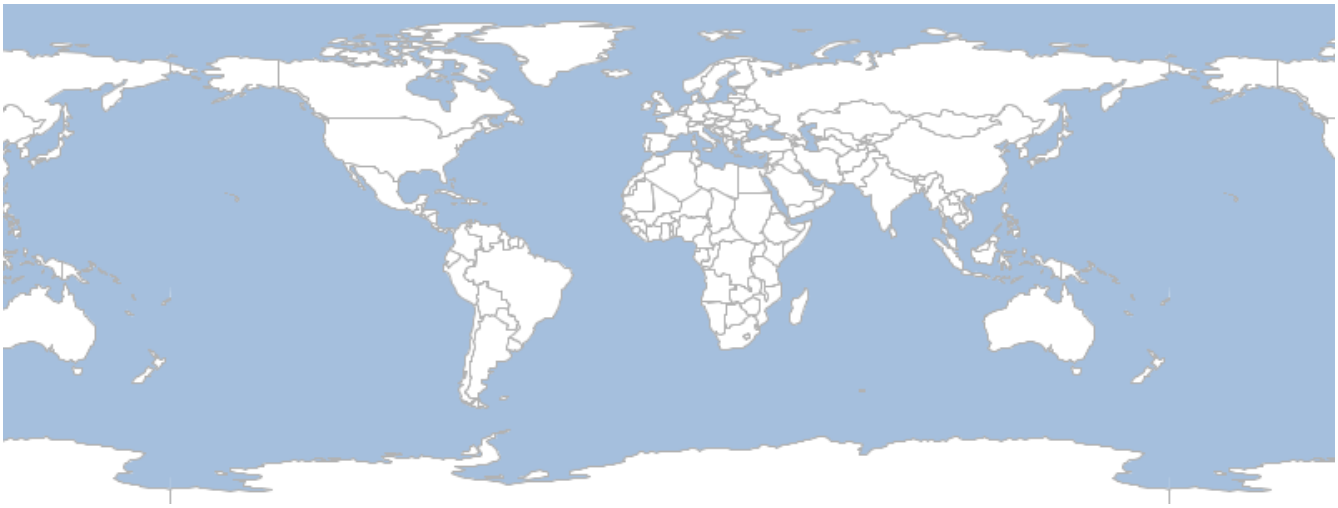




GeoTIFF

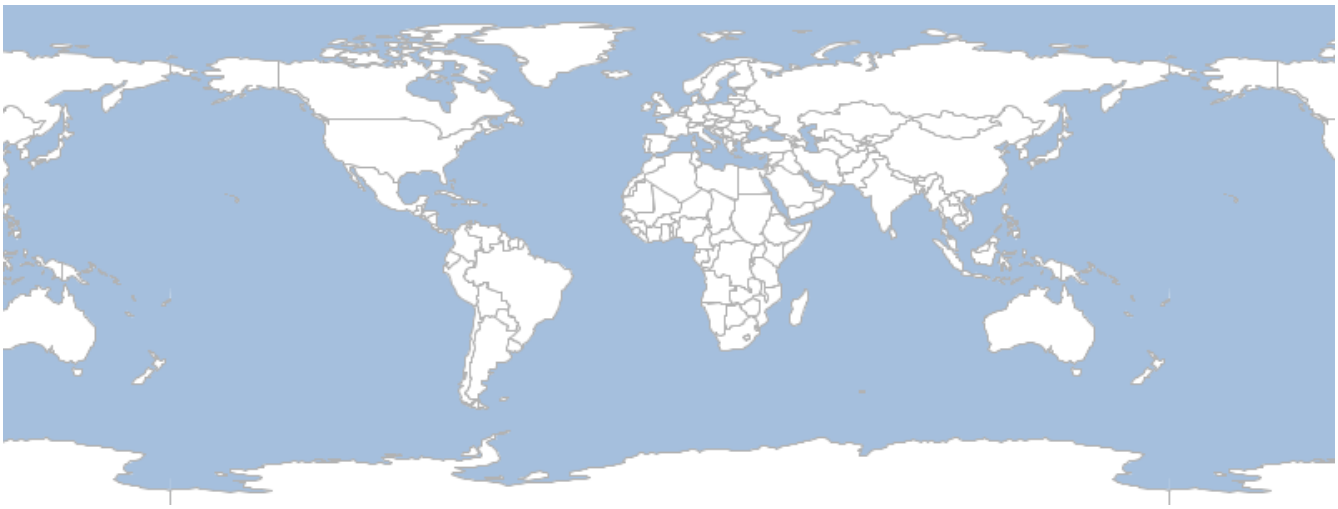
Render a Map to an Image using the GeoTIFF Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
GeoTIFF geotiff = new GeoTIFF()
BufferedImage image = geotiff.render(map)
```



Render a Map to an OutputStream using the GeoTIFF Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
GeoTIFF geotiff = new GeoTIFF()
File file = new File("map.tif")
geotiff.render(map, new FileOutputStream(file))
```



ASCII

Render a Map to an string using the ASCII Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
ASCII ascii = new ASCII(width: 60)
String asciiStr = ascii.render(map)
println asciiStr
```

```
((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
((((((((((((((^(^(.(?.....(((((((((((((^((((((((((((((((
.....:.^(^..??^^!^:..!(^...(((((^.^(((^...........:.^(^..??^
.....(:((.........((.(((.((((((:(.................(:((......
?..(^(((((((^.........((((((.(..............?..(^(((((((^.
.^((((((((((^.......^(((((((...:^.^?...^..:.^((((((((((^
((.((((((((((.....:(((((((^^((^:!^^^^.....((.((((((((((
((((((((((((((.!((((((((((.................((((((((((((
(((((((((^((((((.((.(((((((....:....((..^^((((((((((^((((
^(((((((((((((((((.(((((((^:^...^((((.((^^(^((((((((((((
((((((((((((((((((.!:(((((((.....((((((((((((((((((((((
((.((((((((((((((((.....(((((((...(((((((((^((.((((((((((
(^((((((((((((((((.....(((((((.^.((((((((((^((((((((((((
....(^((((((((((((((...(((((((.^(^(((((((((....(^(((((((((
....((((((((((((((...(((((((^:((((((((((.....((((((((((
((^.((((((((((((((..(((((((((((((((((((((^.((((((((((
((((((((((((((((((.((((((((((((((((((((((((((((((((((((
((((((((((((((((((^((((((((((((((((((((((((((((((((((((
((((((((((((((((((((((((((((((((((((((((((((((((((((((
....^((((((((((((((?((((((((((?....(.........^((((((((((
.....(((((...........((((.......................(((((.....
..........................................................
```

Render a Map to an text file using the ASCII Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
ASCII ascii = new ASCII(width: 60)
File file = new File("map.txt")
FileOutputStream out = new FileOutputStream(file)
ascii.render(map, out)
out.close()
```

```
((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
((((((((((((((^(^(.(?.....(((((((((((((^((((((((((((((((
.....:.^(^..??^^!^:..!(^...(((((^.^(((^...........:.^(^..??^
.....(:((.........((.(((.((((((:(.................(:((......
?..(^(((((((^.........((((((.(..............?..(^(((((((^.
.^((((((((((^.......^(((((((...:^.^?...^..:.^((((((((((^
((.((((((((((.....:(((((((^^((^:!^^^^.....((.((((((((((
((((((((((((((.!((((((((((.................((((((((((((
(((((((((^((((((.((.(((((((....:....((..^^((((((((((^((((
^(((((((((((((((((.(((((((^:^...^((((.((^^(^((((((((((((
((((((((((((((((((.!:(((((((.....((((((((((((((((((((((
((.((((((((((((((((.....(((((((...(((((((((^((.((((((((((
(^((((((((((((((((.....(((((((.^.((((((((((^((((((((((((
....(^((((((((((((((...(((((((.^(^(((((((((....(^(((((((((
....((((((((((((((...(((((((^:((((((((((.....((((((((((
((^.((((((((((((((..(((((((((((((((((((((^.((((((((((
((((((((((((((((((.((((((((((((((((((((((((((((((((((((
((((((((((((((((((^((((((((((((((((((((((((((((((((((((
((((((((((((((((((((((((((((((((((((((((((((((((((((((
....^((((((((((((((?((((((((((?....(.........^((((((((((
.....(((((...........((((.......................(((((.....
..........................................................
```

Base64

Render a Map to an string using the Base64 Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Base64 base64 = new Base64()
String base64Str = base64.render(map)
println base64Str
```

```
image/png;base64,iVBORw0KGgoAAAANSUhEUgAAAyAAAAEsC...
```

Render a Map to an text file using the Base64 Renderer

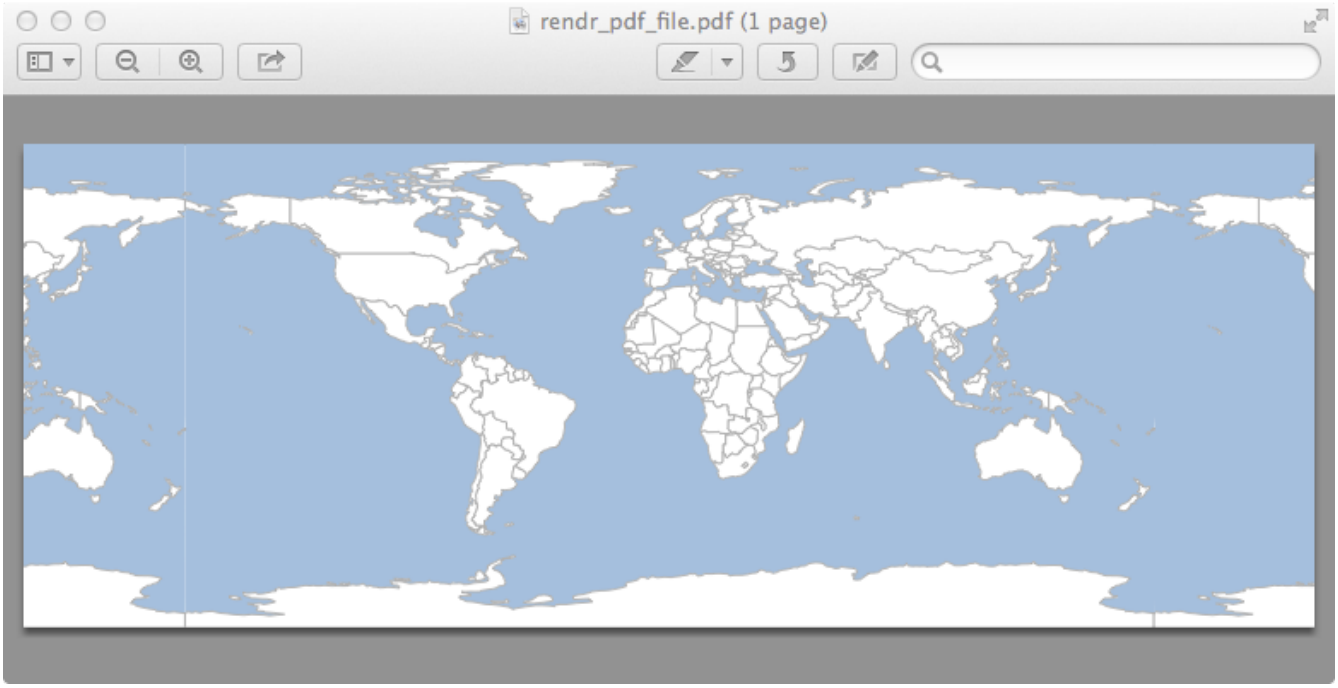
```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Base64 base64 = new Base64()
File file = new File("map.txt")
base64.render(map, new FileOutputStream(file))
```

```
iVBORw0KGgoAAAANSUhEUgAAAyAAAAEsCAYAAAA7Ldc6AACAAE...
```

PDF

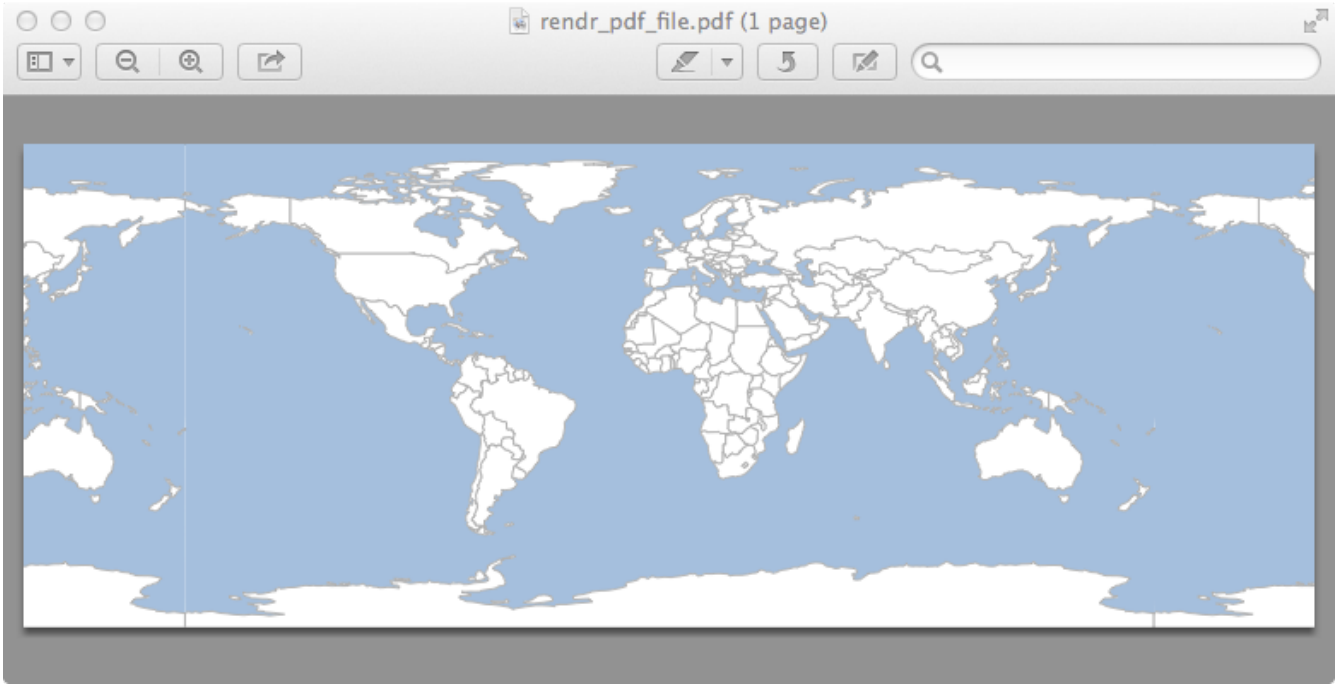
Render a Map to a PDF Document using the PDF Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Pdf pdf = new Pdf()
com.lowagie.text.Document document = pdf.render(map)
```



Render a Map to a PDF file using the PDF Renderer

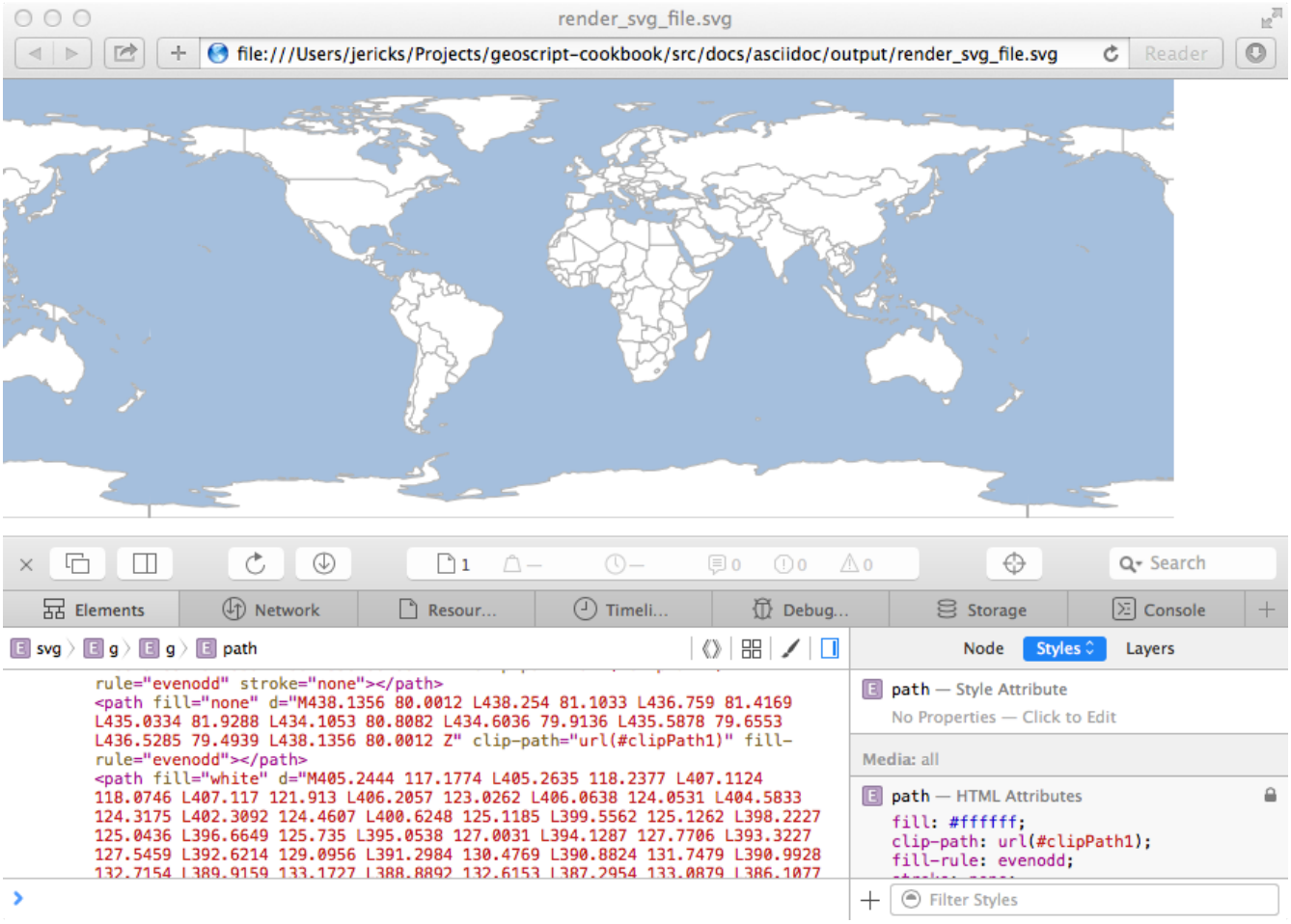
```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Pdf pdf = new Pdf()
File file = new File("map.pdf")
pdf.render(map, new FileOutputStream(file))
```



SVG

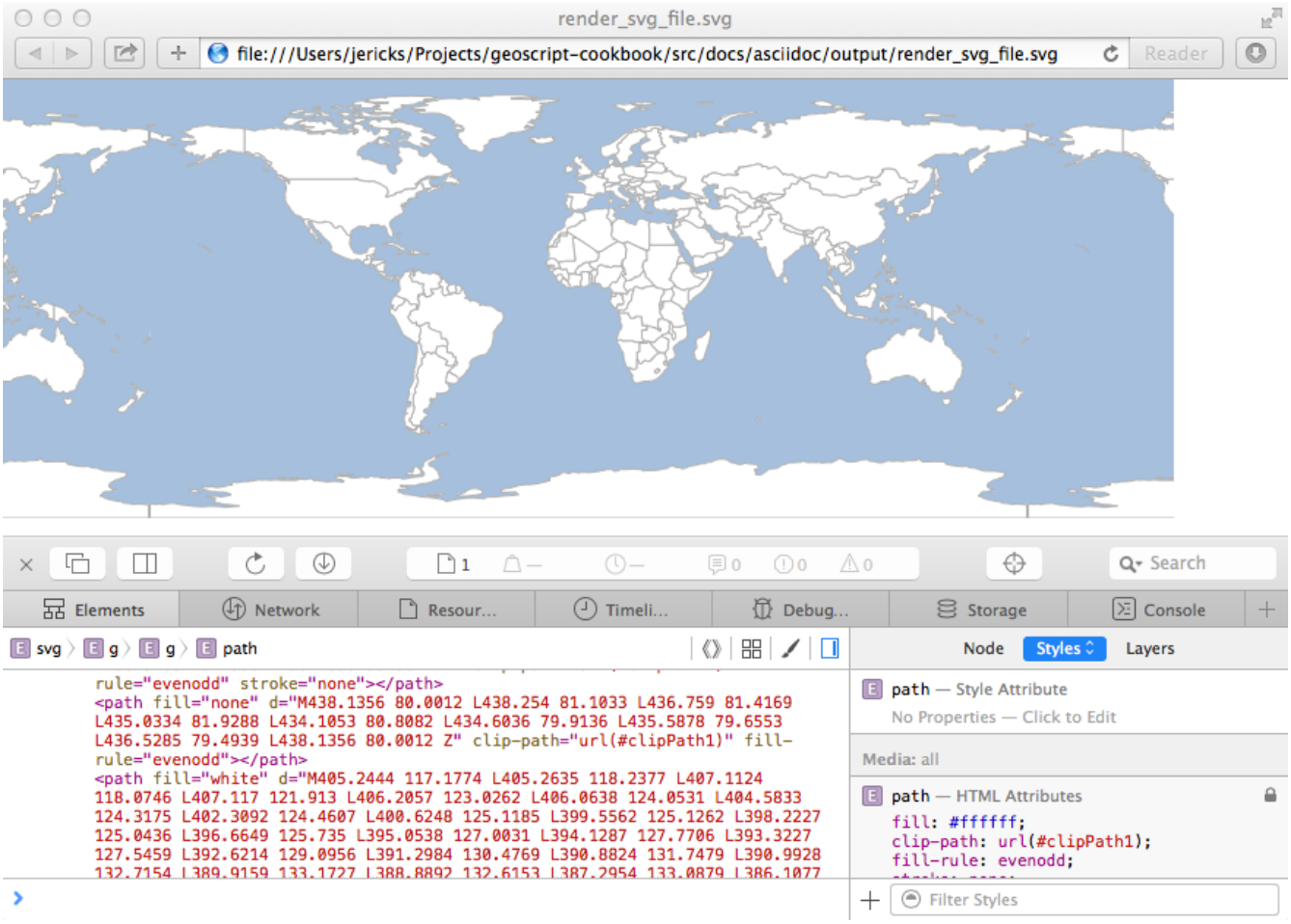
Render a Map to a SVG Document using the SVG Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Svg svg = new Svg()
org.w3c.dom.Document document = svg.render(map)
```



Render a Map to a SVG file using the SVG Renderer

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Svg svg = new Svg()
File file = new File("map.svg")
FileOutputStream out = new FileOutputStream(file)
svg.render(map, out)
out.close()
```



Displaying Maps

Finding Displayers

Get all Displayers

```
List<Displayer> displayers = Displayers.list()
displayers.each { Displayer displayer ->
    println displayer.class.simpleName
}
```

```
MapWindow
Window
```

Get a Displayer

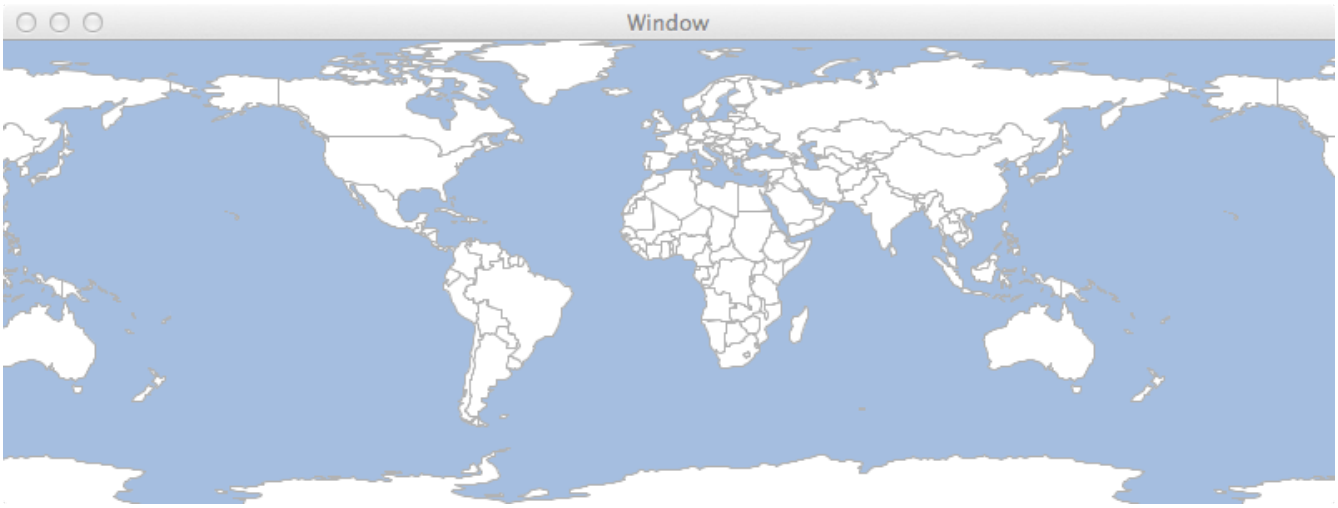
```
Displayer displayer = Displayers.find("window")
println displayer.class.simpleName
```

```
Window
```

Window

Display a Map in a simple GUI

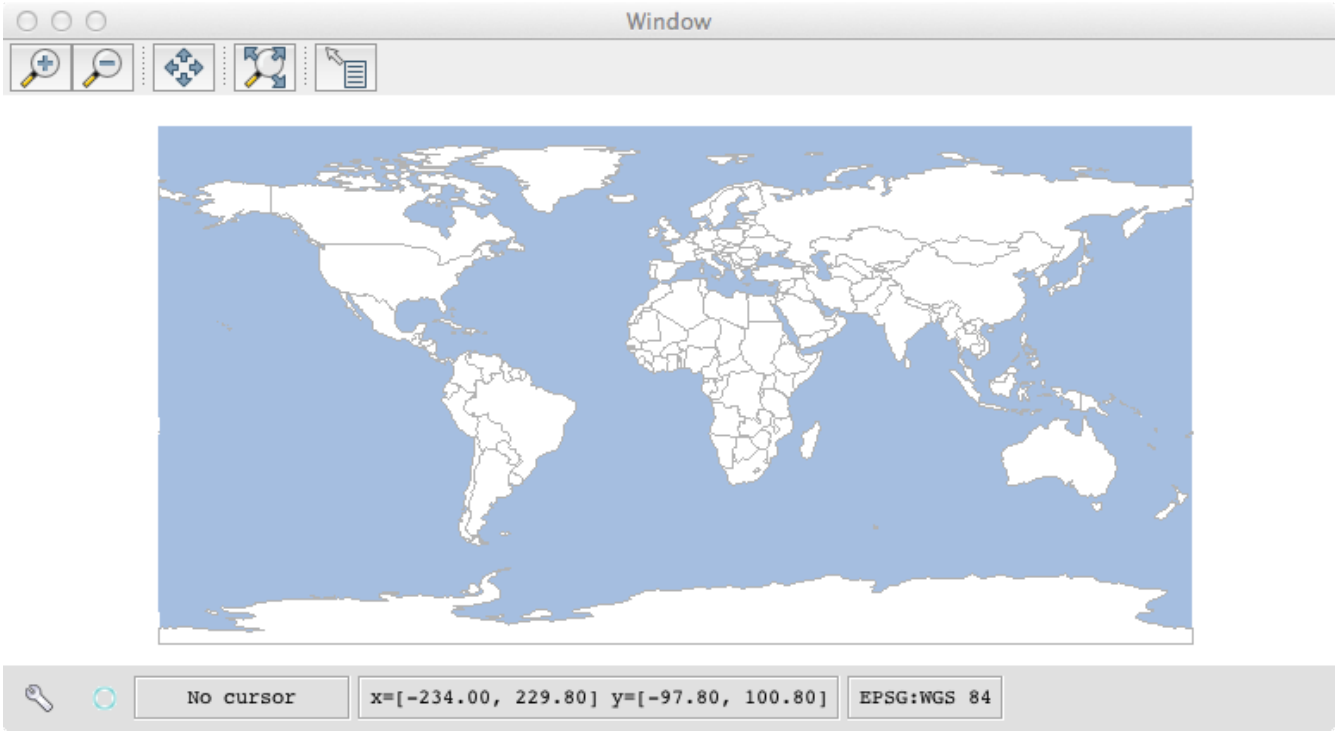
```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
Window window = new Window()
window.display(map)
```



MapWindow

Display a Map in a interactive GUI

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer countries = workspace.get("countries")
countries.style = new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5)
Layer ocean = workspace.get("ocean")
ocean.style = new Fill("#a5bfdd")
Map map = new Map(
        width: 800,
        height: 300,
        layers: [ocean, countries]
)
MapWindow window = new MapWindow()
window.display(map)
```



Drawing

The Draw class is an easy way to quickly render a Geometry, a List of Geometries, a Feature, or a Layer to an Image, a File, an OutputStream, or a GUI.

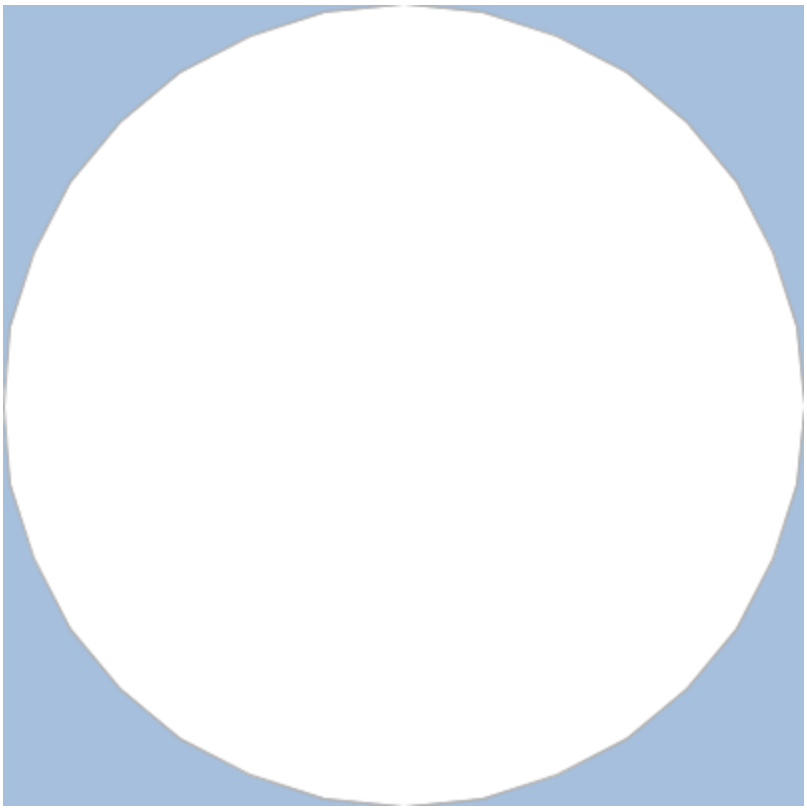
Drawing to a File or GUI

All of the draw methods take a single required parameter but can also take the following optional parameters:

- style = A Style
- bounds = The Bounds
- size = The size of the canvas ([400,350])
- out = The OutputStream, File, or File name. If null (which is the default) a GUI will be opened.
- format = The format ("jpeg", "png", "gif")
- proj = The Projection

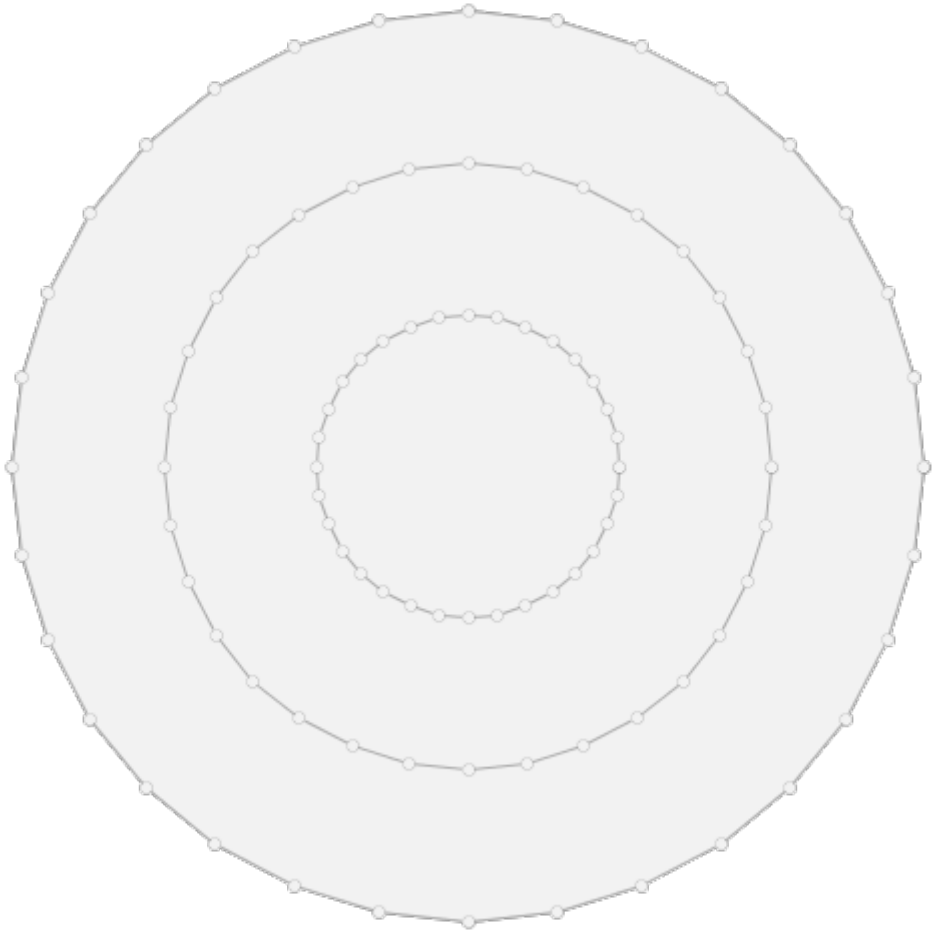
Draw a Geometry to a File

```
File file = new File("geometry.png")
Geometry geometry = new Point(-122.376, 47.587).buffer(0.5)
Draw.draw(geometry,
        style: new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5),
        bounds: new Bounds(-122.876,47.087,-121.876,48.087),
        size: [400,400],
        format: "png",
        proj: "EPSG:4326",
        backgroundColor: "#a5bfdd",
        out: file
)
```



Draw a List of Geometries to an OuputStream

```
Point point = new Point(-122.376, 47.587)
List geometries = [1.5, 1.0, 0.5].collect { double distance ->
    point.buffer(distance)
}
File file = new File("geometries.png")
OutputStream outputStream = new FileOutputStream(file)
Draw.draw(geometries, out: outputStream, format: "png")
outputStream.flush()
outputStream.close()
```

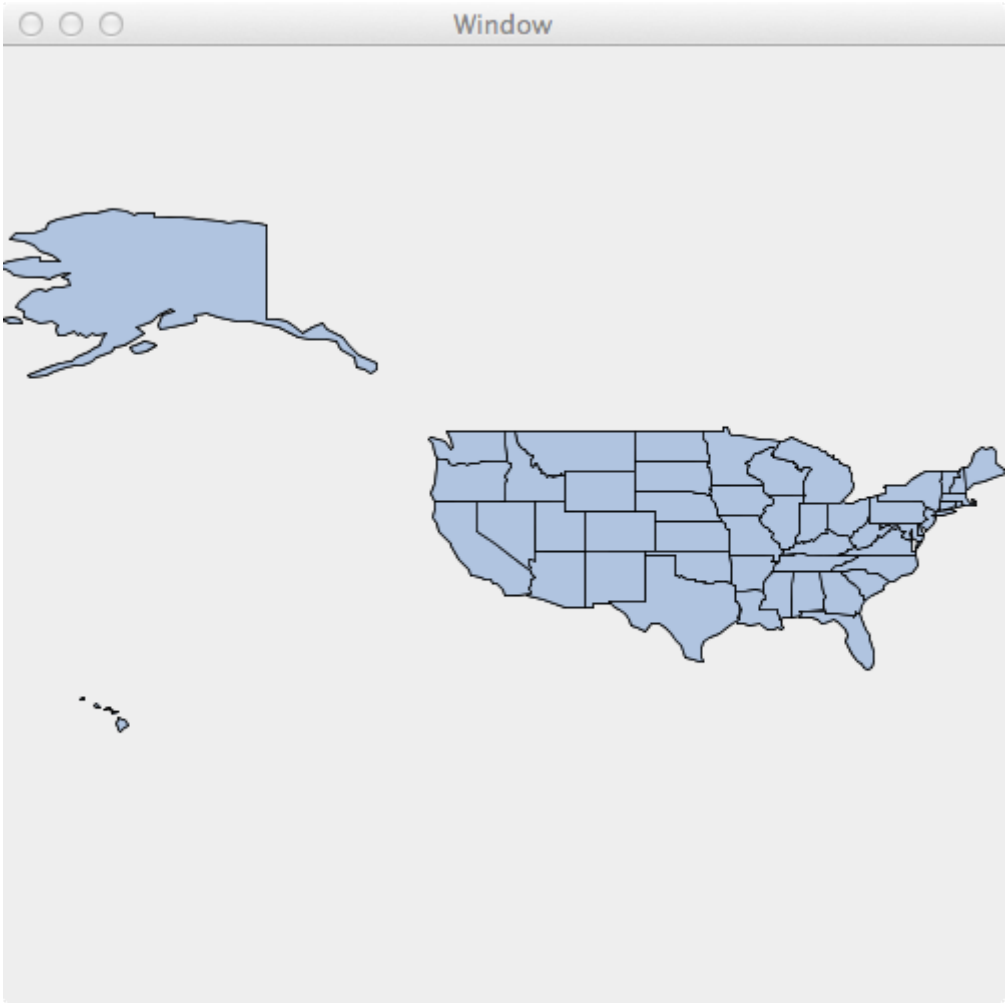


Draw a Feature to a file name

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer layer = workspace.get("states")
Feature feature = layer.first(filter: "NAME_1='Washington'")
Draw.draw(feature, bounds: feature.bounds, out: "feature.png")
```

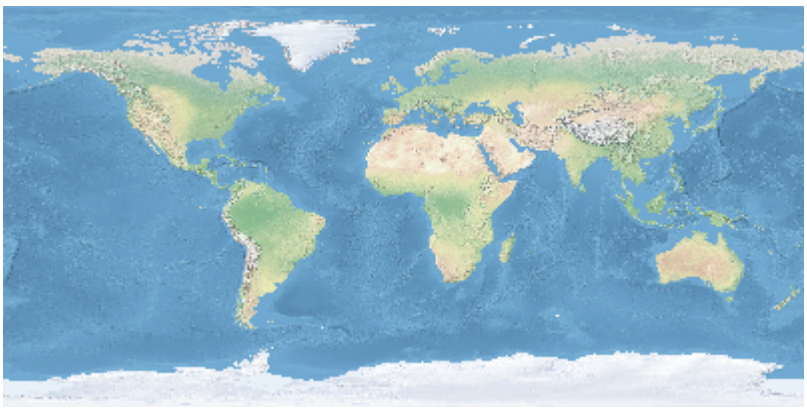
Draw a Layer to a GUI

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer layer = workspace.get("states")
layer.style = new Fill("LightSteelBlue") + new Stroke("LightSlateGrey", 0.25)
Draw.draw(layer, bounds: layer.bounds)
```



Draw a Raster to a File

```
File file = new File("earth.png")
Raster raster = new geoscript.layer.GeoTIFF(new File('src/main/resources/earth.tif'
)).read()
Draw.draw(raster, bounds: raster.bounds, size: [400,200], out: file)
```



Drawing to an Image

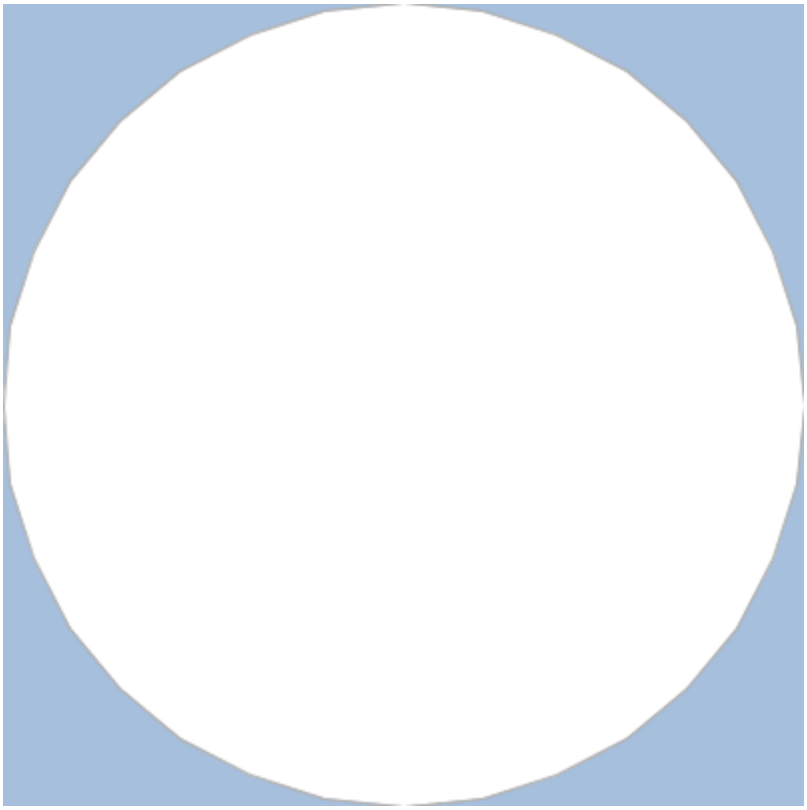
All of the drawToImage methods take a single required parameter but can also take the following optional parameters:

- style = A Style

- bounds = The Bounds
- size = The size of the canvas ([400,350])
- imageType = The format ("jpeg", "png", "gif")
- proj = The Projection

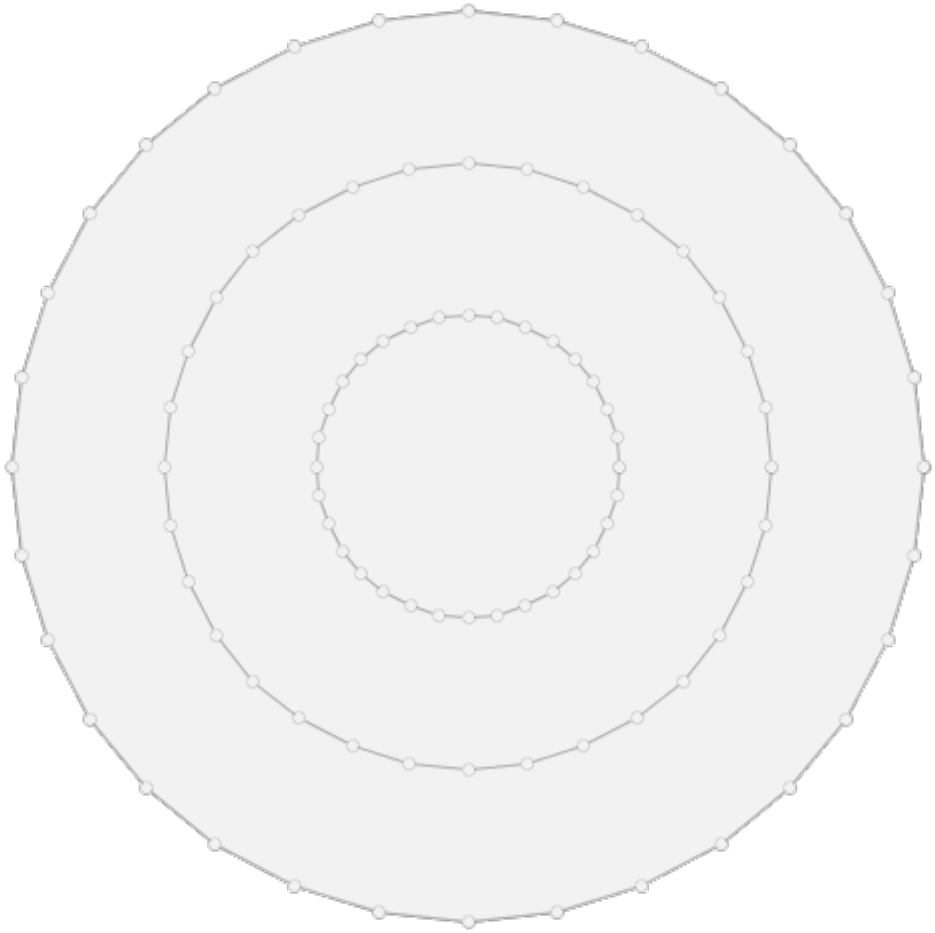
Draw a Geometry to an Image

```
Geometry geometry = new Point(-122.376, 47.587).buffer(0.5)
BufferedImage image = Draw.drawToImage(geometry,
        style: new Fill("#ffffff") + new Stroke("#b2b2b2", 0.5),
        bounds: new Bounds(-122.876,47.087,-121.876,48.087),
        size: [400,400],
        imageType: "png",
        proj: "EPSG:4326",
        backgroundColor: "#a5bfdd"
)
```



Draw a List of Geometries to an Image

```
Point point = new Point(-122.376, 47.587)
List geometries = [1.5, 1.0, 0.5].collect { double distance ->
    point.buffer(distance)
}
BufferedImage image = Draw.drawToImage(geometries)
```

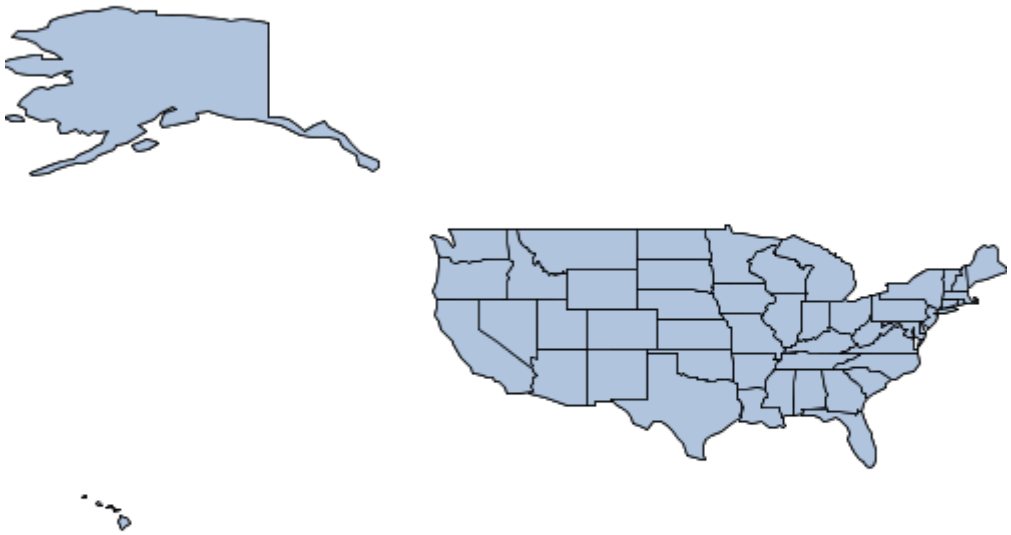


Draw a Feature to an Image

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer layer = workspace.get("states")
Feature feature = layer.first(filter: "NAME_1='Washington'")
BufferedImage image = Draw.drawToImage(feature, bounds: feature.bounds)
```

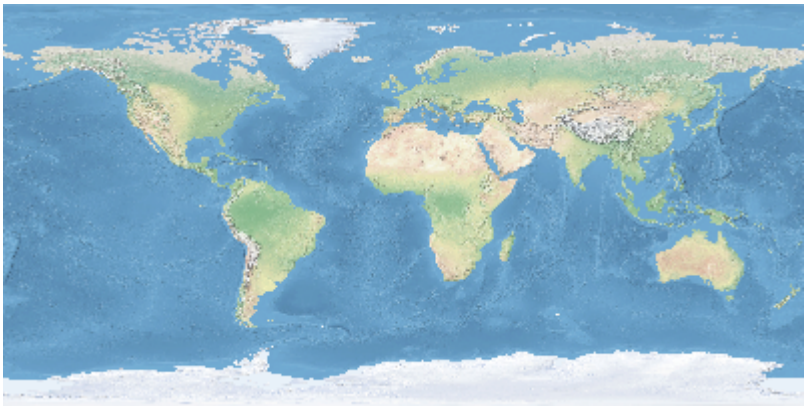
Draw a Layer to an Image

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer layer = workspace.get("states")
layer.style = new Fill("LightSteelBlue") + new Stroke("LightSlateGrey", 0.25)
BufferedImage image = Draw.drawToImage(layer, bounds: layer.bounds)
```



Draw a Raster to an Image

```
Raster raster = new geoscript.layer.GeoTIFF(new File('src/main/resources/earth.tif'
)).read()
BufferedImage image = Draw.drawToImage(raster, bounds: raster.bounds, size: [400,200])
```



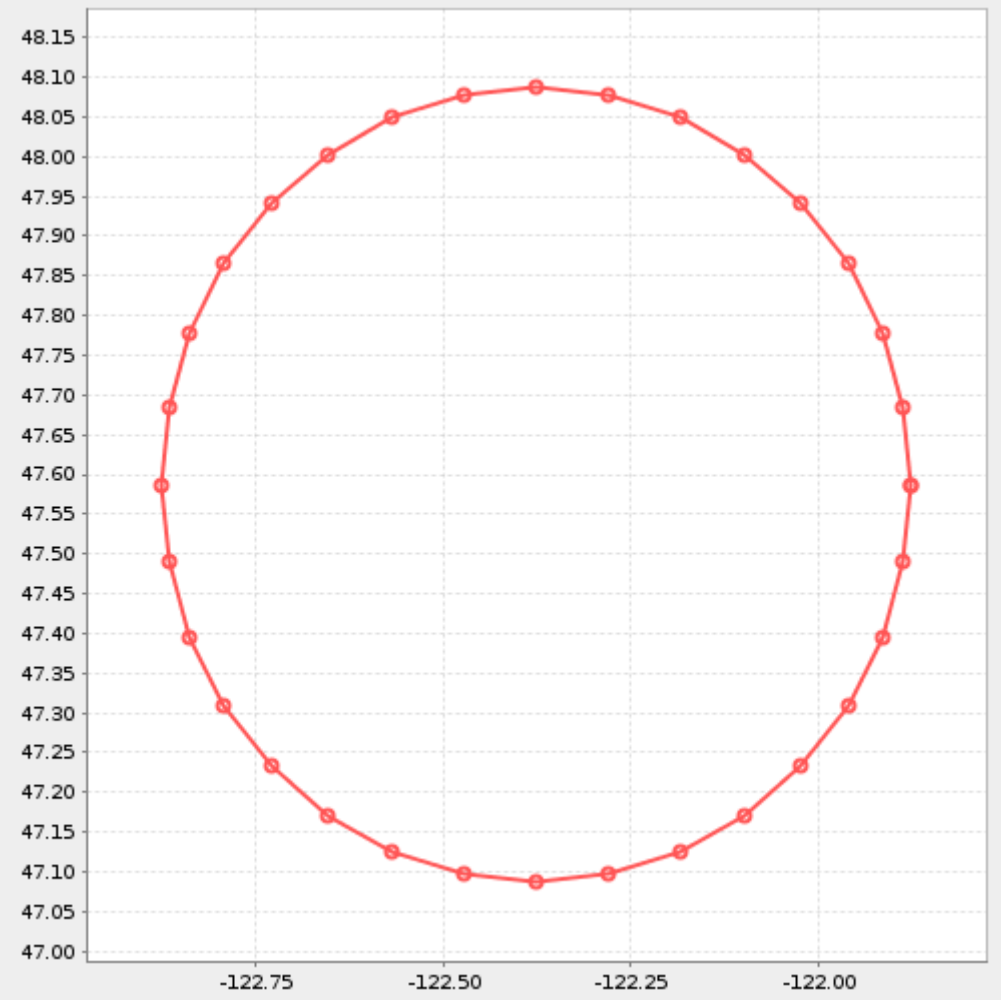
Plotting

Plotting to a File or GUI

The Plot module can plot a Geometry, a list of Geometries, a Feature, or a Layer to a File, a File name, an OutputStream, or a simple GUI.

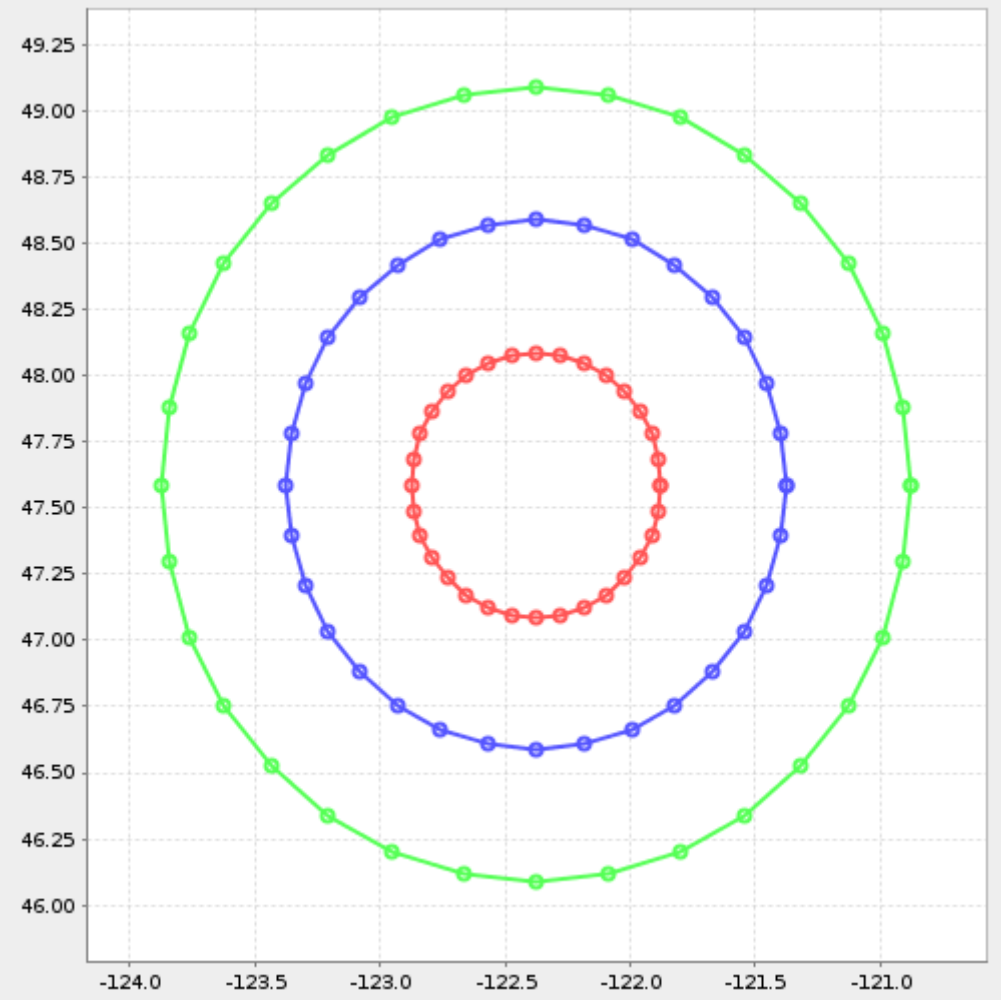
Plot a Geometry to a File

```
File file = new File("geometry.png")
Geometry geometry = new Point(-122.376, 47.587).buffer(0.5)
Plot.plot(geometry, out: file)
```



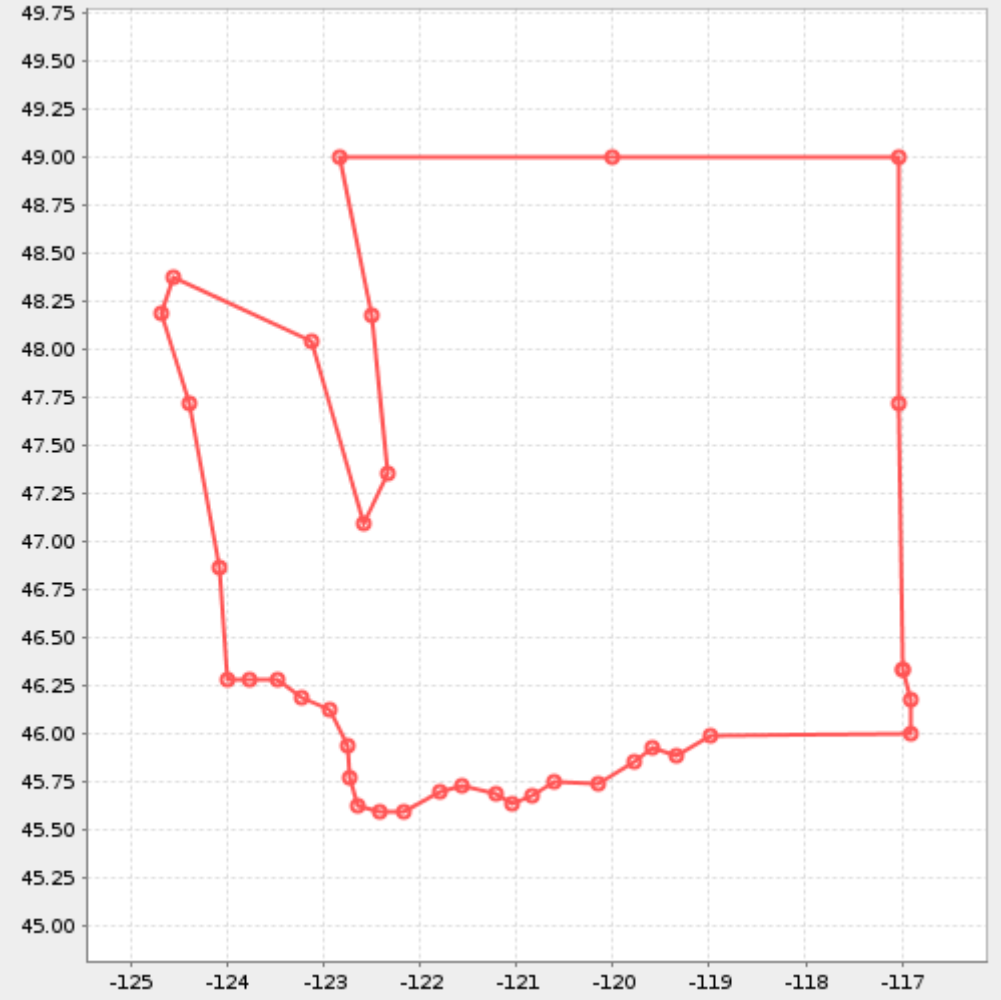
Plot a List of Geometries to an OutputStream

```
Point point = new Point(-122.376, 47.587)
List geometries = [1.5, 1.0, 0.5].collect { double distance ->
    point.buffer(distance)
}
File file = new File("geometries.png")
OutputStream outputStream = new FileOutputStream(file)
Plot.plot(geometries, out: outputStream)
outputStream.flush()
outputStream.close()
```



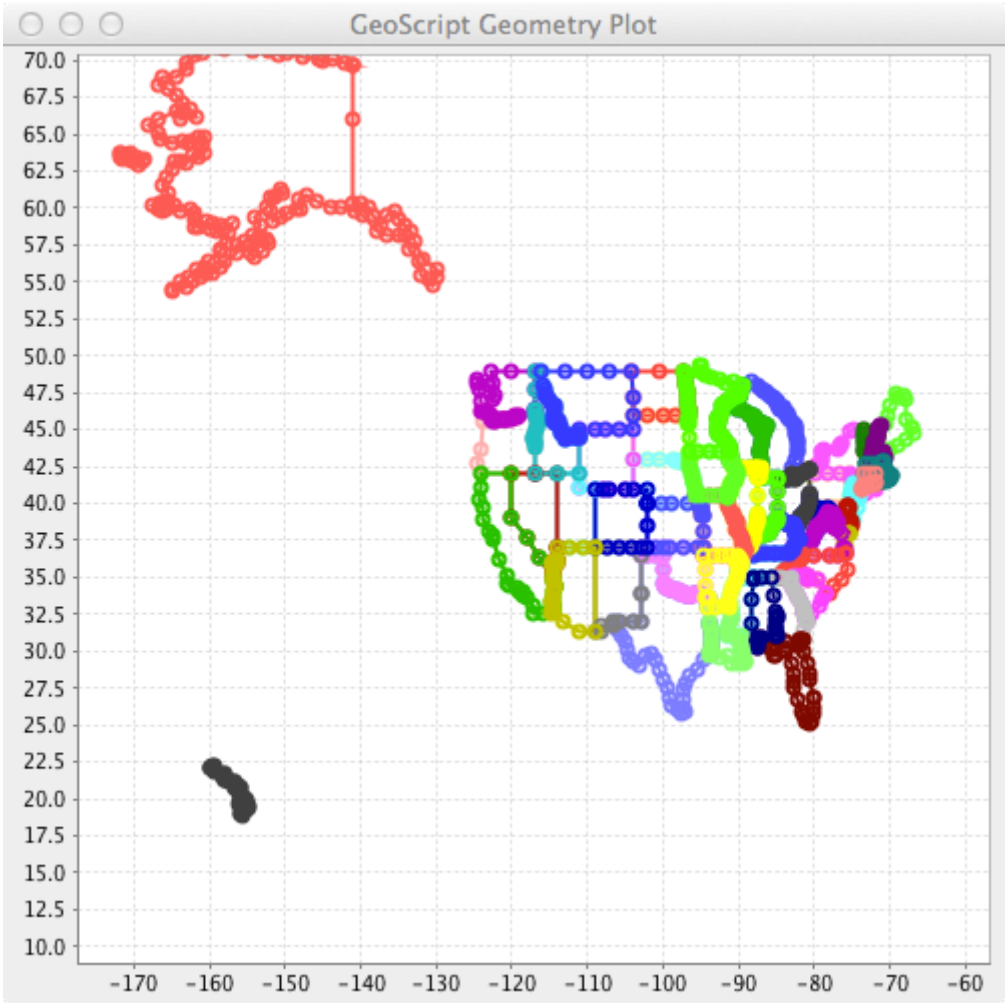
Plot a Feature to a File name

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer layer = workspace.get("states")
Feature feature = layer.first(filter: "NAME_1='Washington'")
Plot.plot(feature, out: "feature.png")
```



Plot a Layer to a GUI

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer layer = workspace.get("states")
Plot.plot(layer)
```

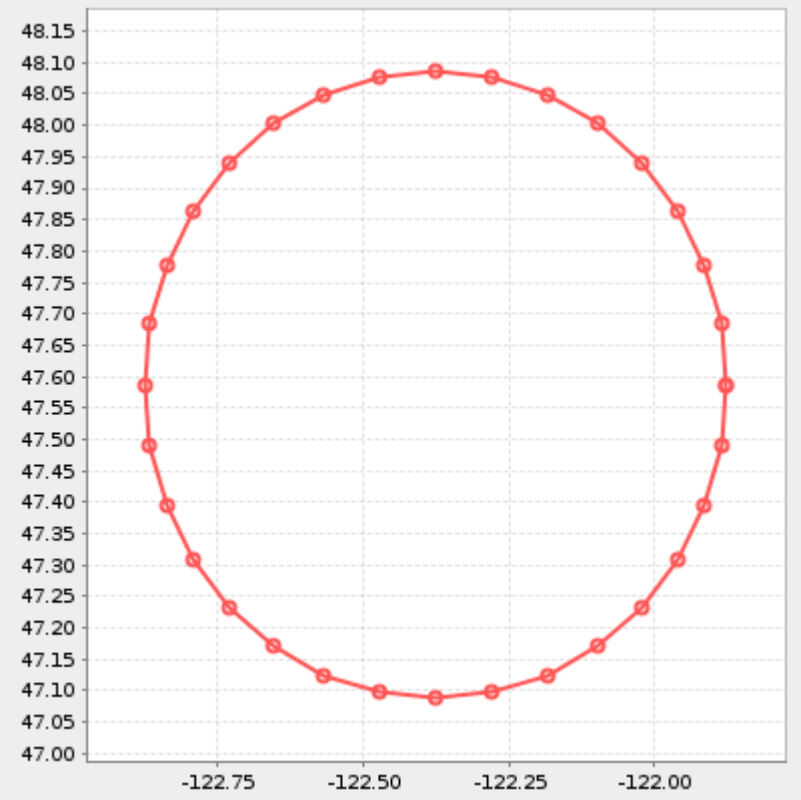


Plotting to an Image

The Plot module can plot a Geometry, a list of Geometries, a Feature, or a Layer to an image.

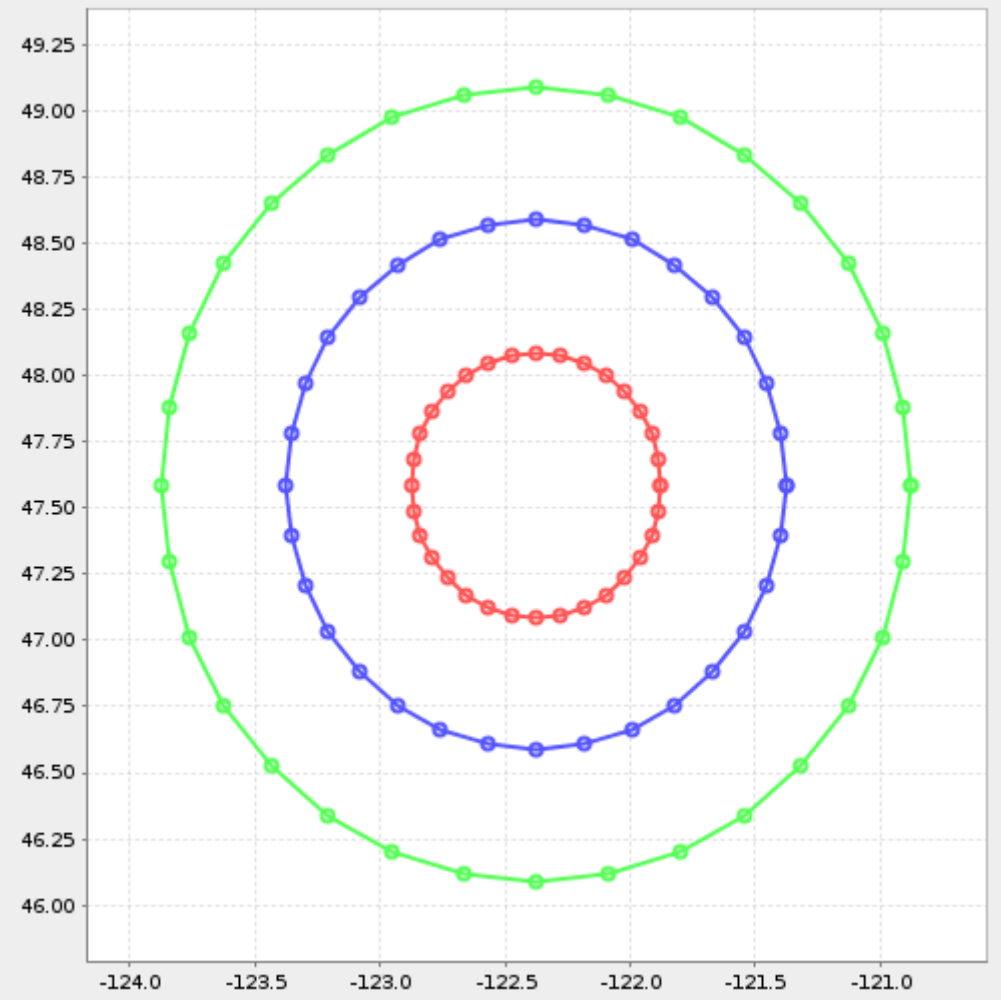
Plot a Geometry to an Image

```
Geometry geometry = new Point(-122.376, 47.587).buffer(0.5)
BufferedImage image = Plot.plotToImage(geometry, size: [400,400],)
```



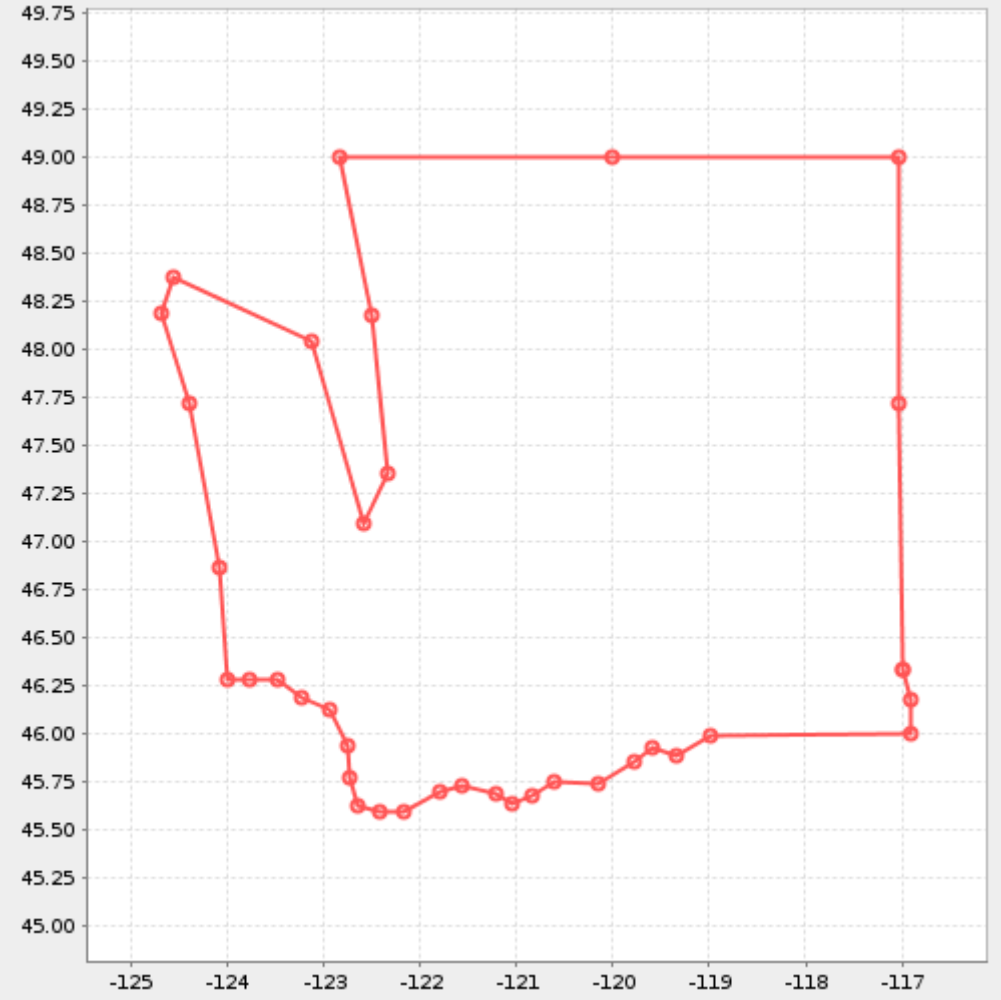
Plot a List of Geometries to an Image

```
Point point = new Point(-122.376, 47.587)
List geometries = [1.5, 1.0, 0.5].collect { double distance ->
    point.buffer(distance)
}
BufferedImage image = Plot.plotToImage(geometries)
```



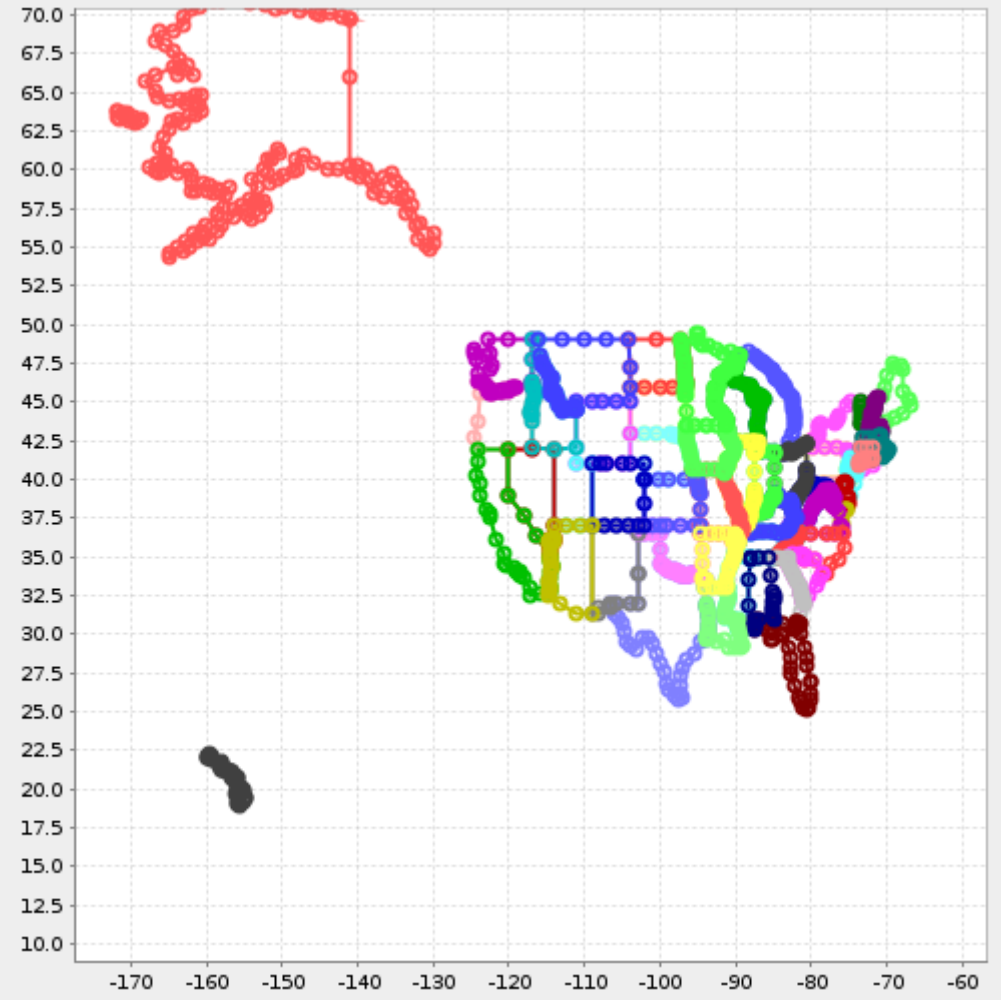
Plot a Feature to an Image

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer layer = workspace.get("states")
Feature feature = layer.first(filter: "NAME_1='Washington'")
BufferedImage image = Plot.plotToImage(feature, bounds: feature.bounds)
```



Plot a Layer to an Image

```
Workspace workspace = new GeoPackage('src/main/resources/data.gpkg')
Layer layer = workspace.get("states")
BufferedImage image = Plot.plotToImage(layer, bounds: layer.bounds)
```


70.0
67.5
65.0
62.5
60.0
57.5
55.0
52.5
50.0
47.5
45.0
42.5
40.0
37.5
35.0
32.5
30.0
27.5
25.0
22.5
20.0
17.5
15.0
12.5
10.0
-170
-160
-150
-140
-130
-120
-110
-100
-90
-80
-70
-60