

Table of Contents

Layer Recipes	1
Getting a Layer's Properties	1
Getting a Layer's Features	2
Reading and Writing Features	4

Layer Recipes

The Layer classes are in the [geoscript.layer](#) package.

A Layer is a collection of Features.

Getting a Layer's Properties

Get a Layer from a Workspace and it's name

```
Workspace workspace = new GeoPackage("src/main/resources/data.gpkg")
Layer layer = workspace.get("countries")
String name = layer.name
println "Name: ${name}"
```

```
Name: countries
```

The Layer's Format

```
String format = layer.format
println "Format: ${format}"
```

```
Format: GeoPackage
```

Count the number of Features

```
int count = layer.count
println "# of Features: ${count}"
```

```
# of Features: 177
```

Get the Layer's Projection

```
Projection proj = layer.proj
println "Projection: ${proj}"
```

```
Projection: EPSG:4326
```

Get the Bounds of the Layer

```
Bounds bounds = layer.bounds
println "Bounds: ${bounds}"
```

```
Bounds: (-179.99999999999997, -
90.000000000000003, 180.00000000000014, 83.64513000000002, EPSG:4326)
```

Getting a Layer's Features

Iterate over a Layer's Features

```
Workspace workspace = new GeoPackage("src/main/resources/data.gpkg")
Layer layer = workspace.get("states")
layer.eachFeature { Feature feature ->
    println feature["NAME_1"]
}
```

```
Minnesota
Montana
North Dakota
Hawaii
Idaho
Washington
Arizona
California
Colorado
Nevada
...
```

Iterate over a subset of a Layer's Features

```
Workspace workspace = new GeoPackage("src/main/resources/data.gpkg")
Layer layer = workspace.get("states")
layer.eachFeature("NAME_1 LIKE 'M%'") { Feature feature ->
    println feature["NAME_1"]
}
```

Minnesota
Montana
Missouri
Massachusetts
Mississippi
Maryland
Maine
Michigan

Iterate over a Layer's Features with parameters.

```
Workspace workspace = new GeoPackage("src/main/resources/data.gpkg")
Layer layer = workspace.get("states")
layer.eachFeature(sort: ["NAME_1"], start: 0, max: 5, fields: ["NAME_1"], filter:
"NAME_1 LIKE 'M%'" ) { Feature feature ->
    println feature["NAME_1"]
}
```

Maine
Maryland
Massachusetts
Michigan
Minnesota

Parameters

- filter: The Filter or Filter String to limit the Features. Defaults to null.
- sort: A List of Lists that define the sort order [[Field or Field name, "ASC" or "DESC"],...]. Not all Layers support sorting!
- max: The maximum number of Features to include
- start: The index of the record to start the cursor at. Together with maxFeatures this simulates paging. Not all Layers support the start index and paging!
- fields: A List of Fields or Field names to include. Used to select only a subset of Fields.

Read all Feature into a List

```
Workspace workspace = new GeoPackage("src/main/resources/data.gpkg")
Layer layer = workspace.get("states")
List<Feature> features = layer.features

println "# Features = ${features.size()}"
features.each { Feature feature ->
    println feature["NAME_1"]
}
```

```
# Features = 52
Minnesota
Montana
North Dakota
Hawaii
Idaho
Washington
Arizona
California
Colorado
Nevada
...
```

Reading and Writing Features

The Layer IO classes are in the [geoscript.layer.io](https://github.com/geoscript/layer.io) package.

GeoJSON

Get GeoJSON String from a Layer

```
Workspace workspace = new Memory()
Schema schema = new Schema("cities", [
    new Field("geom", "Point", "EPSG:4326"),
    new Field("id", "Integer"),
    new Field("name", "String")
])
Layer layer = workspace.create(schema)
layer.add([
    geom: new Point(-122.3204, 47.6024),
    id: 1,
    name: "Seattle"
])
layer.add([
    geom: new Point(-122.48416, 47.2619),
    id: 2,
    name: "Tacoma"
])

String geojson = layer.toJSONString()
println geojson
```

```

{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "geometry": {
        "type": "Point",
        "coordinates": [
          -122.3204,
          47.6024
        ]
      },
      "properties": {
        "id": 1,
        "name": "Seattle"
      },
      "id": "fid--5a9ddf2_15eb0ce1308_-7fd8"
    },
    {
      "type": "Feature",
      "geometry": {
        "type": "Point",
        "coordinates": [
          -122.4842,
          47.2619
        ]
      },
      "properties": {
        "id": 2,
        "name": "Tacoma"
      },
      "id": "fid--5a9ddf2_15eb0ce1308_-7fd6"
    }
  ]
}

```

GeoBuf

Get GeoBuf String from a Layer

```
Workspace workspace = new Memory()
Schema schema = new Schema("cities", [
    new Field("geom", "Point", "EPSG:4326"),
    new Field("id", "Integer"),
    new Field("name", "String")
])
Layer layer = workspace.create(schema)
layer.add([
    geom: new Point(-122.3204, 47.6024),
    id: 1,
    name: "Seattle"
])
layer.add([
    geom: new Point(-122.48416, 47.2619),
    id: 2,
    name: "Tacoma"
])

String geobuf = layer.toGeobufString()
println geobuf
```

```
0a0269640a046e616d6510021806223d0a1d0a0c08001a089fd8d374c0ebb22d6a0218016a090a07536561
74746c650a1c0a0c08001a08ff6e77498a3892d6a0218026a080a065461636f6d61
```

GML

Get GML String from a Layer

```
Workspace workspace = new Memory()
Schema schema = new Schema("cities", [
    new Field("geom", "Point", "EPSG:4326"),
    new Field("id", "Integer"),
    new Field("name", "String")
])
Layer layer = workspace.create(schema)
layer.add([
    geom: new Point(-122.3204, 47.6024),
    id: 1,
    name: "Seattle"
])
layer.add([
    geom: new Point(-122.48416, 47.2619),
    id: 2,
    name: "Tacoma"
])

String gml = layer.toGMLString()
println gml
```

```
<wfs:FeatureCollection xmlns:wfs="http://www.opengis.net/wfs">
  <gml:boundedBy xmlns:gml="http://www.opengis.net/gml">
    <gml:Box srsName="http://www.opengis.net/gml/srs/epsg.xml#4326">
      <gml:coord>
        <gml:X>
          -122.48416
        </gml:X>
        <gml:Y>
          47.2619
        </gml:Y>
      </gml:coord>
      <gml:coord>
        <gml:X>
          -122.3204
        </gml:X>
        <gml:Y>
          47.6024
        </gml:Y>
      </gml:coord>
    </gml:Box>
  </gml:boundedBy>
  <gml:featureMember xmlns:gml="http://www.opengis.net/gml">
    <gsf:cities xmlns:gsf="http://geoscript.org/feature" fid="fid--5a9ddf2_15eb0ce1308_-7fe0">
      <gml:name>
        Seattle
      </gml:name>
    </gsf:cities>
  </gml:featureMember>
</wfs:FeatureCollection>
```



```

    <gsf:geom>
      <gml:Point>
        <gml:coord>
          <gml:X>
            -122.3204
          </gml:X>
          <gml:Y>
            47.6024
          </gml:Y>
        </gml:coord>
      </gml:Point>
    </gsf:geom>
    <gsf:id>
      1
    </gsf:id>
  </gsf:cities>
</gml:featureMember>
<gml:featureMember xmlns:gml="http://www.opengis.net/gml">
  <gsf:cities xmlns:gsf="http://geoscript.org/feature" fid="fid--
5a9ddf2_15eb0ce1308_-7fde">
    <gml:name>
      Tacoma
    </gml:name>
    <gsf:geom>
      <gml:Point>
        <gml:coord>
          <gml:X>
            -122.48416
          </gml:X>
          <gml:Y>
            47.2619
          </gml:Y>
        </gml:coord>
      </gml:Point>
    </gsf:geom>
    <gsf:id>
      2
    </gsf:id>
  </gsf:cities>
</gml:featureMember>
</wfs:FeatureCollection>

```

KML

Get KML String from a Layer

```
Workspace workspace = new Memory()
Schema schema = new Schema("cities", [
    new Field("geom", "Point", "EPSG:4326"),
    new Field("id", "Integer"),
    new Field("name", "String")
])
Layer layer = workspace.create(schema)
layer.add([
    geom: new Point(-122.3204, 47.6024),
    id: 1,
    name: "Seattle"
])
layer.add([
    geom: new Point(-122.48416, 47.2619),
    id: 2,
    name: "Tacoma"
])

String kml = layer.toKMLString()
println kml
```

```
<kml:kml xmlns:kml="http://www.opengis.net/kml/2.2">
  <kml:Document>
    <kml:Folder>
      <kml:name>
        cities
      </kml:name>
      <kml:Schema kml:name="cities" kml:id="cities">
        <kml:SimpleField kml:name="id" kml:type="Integer"/>
        <kml:SimpleField kml:name="name" kml:type="String"/>
      </kml:Schema>
      <kml:Placemark>
        <kml:name>
          fid--5a9ddf2_15eb0ce1308_-7fdc
        </kml:name>
        <kml:Style>
          <kml:IconStyle>
            <kml:color>
              ff0000ff
            </kml:color>
          </kml:IconStyle>
        </kml:Style>
        <kml:ExtendedData>
          <kml:SchemaData kml:schemaUrl="#cities">
            <kml:SimpleData kml:name="id">
              1
            </kml:SimpleData>
            <kml:SimpleData kml:name="name">
```

```

        Seattle
      </kml:SimpleData>
    </kml:SchemaData>
  </kml:ExtendedData>
  <kml:Point>
    <kml:coordinates>
      -122.3204,47.6024
    </kml:coordinates>
  </kml:Point>
</kml:Placemark>
<kml:Placemark>
  <kml:name>
    fid--5a9ddf2_15eb0ce1308_-7fda
  </kml:name>
  <kml:Style>
    <kml:IconStyle>
      <kml:color>
        ff0000ff
      </kml:color>
    </kml:IconStyle>
  </kml:Style>
  <kml:ExtendedData>
    <kml:SchemaData kml:schemaUrl="#cities">
      <kml:SimpleData kml:name="id">
        2
      </kml:SimpleData>
      <kml:SimpleData kml:name="name">
        Tacoma
      </kml:SimpleData>
    </kml:SchemaData>
  </kml:ExtendedData>
  <kml:Point>
    <kml:coordinates>
      -122.48416,47.2619
    </kml:coordinates>
  </kml:Point>
</kml:Placemark>
</kml:Folder>
</kml:Document>
</kml:kml>

```