

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://geoscript.io) package.

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-30e429b1_15e9cda07d8_-7fe4"
    },
    {
      "type": "Feature",
      "geometry": {
        "type": "Point",
        "coordinates": [
          -122.4842,
          47.2619
        ]
      },
      "properties": {
        "id": 2,
        "name": "Tacoma"
      },
      "id": "fid-30e429b1_15e9cda07d8_-7fe2"
    }
  ]
}

```