

FOSS4G SEOUL 2015

OGC GeoPackage in practice: Implementing a new OGC specification with open-source tools

BY STEVEN D. LANDER, RGi®

ABOUT ME

- <https://github.com/stevendlander>
- Software Engineer @ RGi
- Experienced in caching and storing large raster images
- Early implementer of the [OGC GeoPackage Encoding Standard](#)
- Work in Java, Android, Python, and some others

ABOUT OUR PROJECTS

- Software to Aggregate Geospatial Data (SWAGD)
 - Full implementation of the GeoPackage raster spec
 - Java 1.8
 - Uses GDAL 1.11.1
- [geopackage-python](#)
 - Naïve implementation of the GeoPackage raster spec
 - Python 2.7 & 3.4
 - Improvements to gdal2tiles.py along with separate script to package tiles

OTHER PRESENTATIONS AT FOSS4G SEOUL

- *Geopackage and how open source is changing the way governments think about standards*
 - (2015/09/16) Nathan Frantz, Ben Tuttle, 11:25 PT1-05
- *Building Continuous Integration within your open source project*
 - (2015/09/18) Steven Lander, 11:25 PT7-09

AT A GLANCE

- What is the GeoPackage Encoding Standard?
- Software to Aggregate Geospatial Data (SWAGD)
- Our approach to the implementation of GeoPackage
- What we learned building the reference implementation

WHAT IS THE GEOPACKAGE ENCODING STANDARD?

A set of conventions for storing data in a SQLite database

- vector features
- tile matrix sets of imagery and raster maps at various scales
- schema
- metadata
- extensions

HOW STANDARDS PROLIFERATE:
(SEE: A/C CHARGERS, CHARACTER ENCODINGS, INSTANT MESSAGING, ETC.)

SITUATION:
THERE ARE
14 COMPETING
STANDARDS.

14?! RIDICULOUS!
WE NEED TO DEVELOP
ONE UNIVERSAL STANDARD
THAT COVERS EVERYONE'S
USE CASES.



YEAH!

SOON:

SITUATION:
THERE ARE
15 COMPETING
STANDARDS.

EXISTING TECHNOLOGY

	Tile sets	Vector	Schema	Metadata	Extensions
Shapefile		✓		✓	
KML	✓	✓			
GeoJSON		✓			
MBTiles	✓*				

COMMUNITY IMPLEMENTATIONS

- GDAL
 - Features in 1.11.0, raster tiles in 2.0
- GeoServer
 - Community supported plugin
- SpatiaLite
 - Version 4.2.0
- ESRI
 - Feature support, raster support forthcoming
- DigitalGlobe
 - Export imagery to raster GeoPackage*

OUR APPROACH TO THE IMPLEMENTATION OF GEOPACKAGE:

Software to Aggregate Geospatial Data (SWAGD)

OVERVIEW

- Create an API from which other developers could pick and choose the functionality they need
- Build an UI around the API to demonstrate
- Java 1.8 with 1.6 backport

CORE API

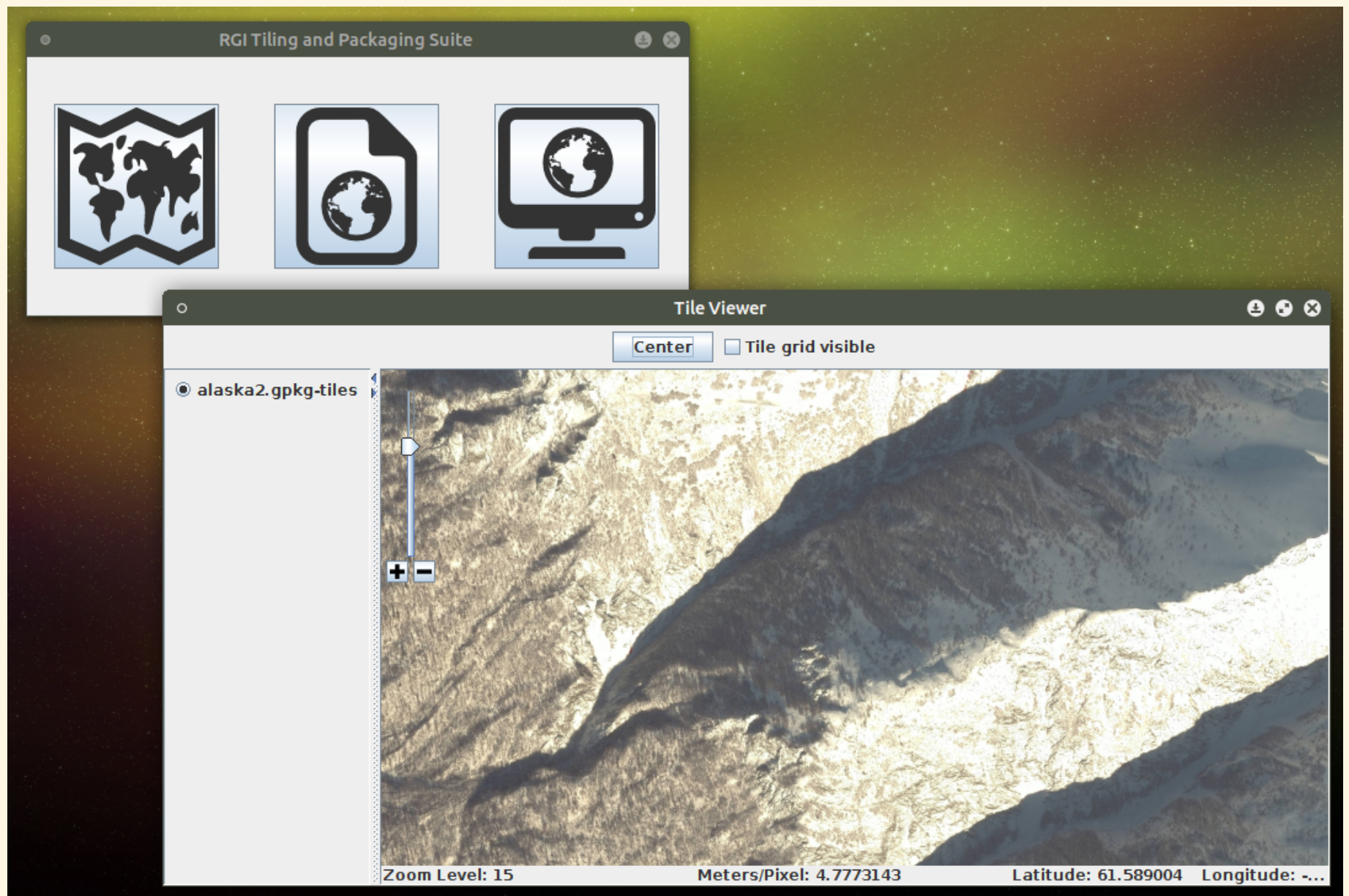
- Common - Geospatial functionality dependency
- DataStore - Tile store manipulation classes
- Gdal2Tiles - Creates tiles, heavily influenced by GDAL python functionality
- GeoPackage - Classes and validation for creating and reading GeoPackages
- NetworkExtension - GeoPackage extension for supporting networks of different kinds

INTEROPERABILITY SUPPORT

- GeoPackage verification component
 - Each opened GeoPackage is verified to be specification compliant
 - Made into a separate verification program
- Flexibility of the JVM
- Outputs Web Mercator (3857), Global Geodetic (4326)*, and Global Mercator (3395)*

REFERENCE UI

- Swing based UI with workflow buttons for tiling, packaging, and viewing
- JMapView handles viewing tiled imagery of all types
- Future UI redesign will use JavaFX



Tiling Settings

Input

File: ...

Native reference System: EPSG:3857 - WGS 84 / Pseudo-Mercator

Output reference System:

Output

Format:

Tile width:

Tile height:


Clear color: 

Image format:

Compression type:

Compression quality:

Tile set name:

Tile set description:

File name: ...

OK

Cancel

Packaging Settings

Input

File: /data/tiles/mercator/alaska2 ...

Reference system: EPSG:3857 - WGS84 / Web Mercator ▼

Output

Format: GeoPackage ▼

Image format: image/jpeg ▼

Compression type: JPEG ▼

Compression quality: 0.75 ▼

Tile set name: alaska2

Tile set description: Tile store alaska2 packaged by steven.lander at 2015-0

File name: /data/geopackage/alaska2.gpkg ...

OK Cancel

VERIFIER

- JavaFX
- Drag-n-Drop!
- Tests the following in the GeoPackage Encoding Standard:
 - core
 - tiles
 - schema
 - extensions
 - metadata





WHAT WE LEARNED BUILDING THE REFERENCE IMPLEMENTATION

JAVA

- There are few choices for Java native viewing of tiled data sets
- We wish we knew about JavaFX early in development
- Python still tiles raster images tremendously faster than our Java implementation
- Database optimization is more important than code optimization
 - SQLite pragma
 - The number of queries you process

ANDROID

- Android Java compatibility is stuck in 1.6 (official) or 1.7 (unofficial)
- Desktop development ill-suits android
 - SQLite implementations differ (SQLDroid)
 - BufferedImage vs. BitmapImage
 - Differing testing frameworks
- Issues with globally available android jar

SPECIFICATION ISSUES

- Even a verbose and precise specification can be interpreted differently by different actors
- Compromises are key to adoption
- Lingo can differ

GENERAL

- Think critically about who would want to use your code and how
- Constrain the breadth of choices to your users where appropriate

THANKS

