GXF

Contents

Chapter 1

index

<title>GXF-3</title>

Introduction

The GXF-3 library is intended to make correct implementation of GXF-3 file format readers easy. It consists of free (OpenSource) C source code for functions to read GXF-3 raster files, and an example program using them to convert GXF data to GeoTIFF format.

GXF (Grid eXchange File) is a standard ASCII file format for exchanging gridded data among different software systems. Software that supports the GXF standard will be able to import properly formatted GXF files and export grids in GXF format. GXF-3 is an adopted standard format of the Gravity/Magnetics Committee of the Society of Exploration Geophysicists (SEG). GXF-3 is the primary gridded data interchange format for Geosoft.

Resources

- GXF-3 API Documentation
- GXF-3 API in .tar.gz
- GXF-3 API in ZIP
- GXF-3 File Format Specification (pdf)

Licensing

This library is offered as Open Source. In particular, it is offered under the X Consortium license which doesn't attempt to impose any copyleft, or credit requirements on users of the code.

The precise license text is:

Building the Source

Unix developers should be able to unpack the .tar.gz file, run configure, and type make to build the library (libgxf3.a), and a simple test program (gxftest.c).

Windows developers should unpack the .zip file, and type `nmake /f makefile.vc` to build with VC++.

Author and Acknowledgements

The primary author of the GXF3 library is Frank Warmerdam, and I can be reached at warmerdam@pobox.com. I am open to bug reports, and suggestions.

I would like to thank:

- Global Geomatics who funded development the majority of the work on this library, and agreed for it to be Open Source.
- Ian Macleod of Geosoft who answered a number of questions I had, and provided sample files.

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 4

Class Documentation

4.1 GXFInfo_t Struct Reference

The documentation for this struct was generated from the following file:

- gxfopen.h

Chapter 5

File Documentation

5.1 gxfopen.h File Reference

```
#include "cpl_conv.h"
#include "cpl_string.h"
```

Classes

- struct GXFInfo_t

Functions

- GXFHandle GXFOpen (const char *pszFilename)
- CPLErr GXFGetRawInfo (GXFHandle hGXF, int *pnXSize, int *pnYSize, int *pnSense, double *pdfZMin, double *pdfZMax, double *pdfDummy)
- CPLErr GXFGetRawScanline (GXFHandle, int iScanline, double *padfLineBuf)
- CPLErr GXFGetScanline (GXFHandle, int iScanline, double *padfLineBuf)
- char ** GXFGetMapProjection (GXFHandle)
- char ** GXFGetMapDatumTransform (GXFHandle)
- char * GXFGetMapProjectionAsPROJ4 (GXFHandle)
- char * GXFGetMapProjectionAsOGCWKT (GXFHandle)
- CPLErr GXFGetRawPosition (GXFHandle, double *, double *, double *, double *, double *)
- CPLErr GXFGetPosition (GXFHandle, double *, double *, double *, double *, double *)
- void GXFClose (GXFHandle hGXF)

5.1.1 Detailed Description

Public GXF-3 function definitions.

5.1.2 Function Documentation

5.1.2.1 GXFClose()

```
void GXFClose (
            GXFHandle hGXF )
```

Close GXF file opened with GXFOpen().

Parameters

hGXF	handle to GXF file.

5.1.2.2 GXFGetMapDatumTransform()

```
char** GXFGetMapDatumTransform (
            GXFHandle hGXF )
```

Return the lines related to the datum transformation. It is up to the caller to parse them and interpret. The return result will be NULL if no #MAP_DATUM_TRANSFORM line was found in the header.

Parameters

hGXF	the GXF file handle.

Returns

a NULL terminated array of string pointers containing the datum, or NULL. The strings remained owned by the GXF API, and should not be modified or freed by the caller.

5.1.2.3 GXFGetMapProjection()

```
char** GXFGetMapProjection (
            GXFHandle hGXF )
```

Return the lines related to the map projection. It is up to the caller to parse them and interpret. The return result will be NULL if no #MAP_PROJECTION line was found in the header.

Parameters

hGXF	the GXF file handle.

Returns

a NULL terminated array of string pointers containing the projection, or NULL. The strings remained owned by the GXF API, and should not be modified or freed by the caller.

5.1.2.4 GXFGetMapProjectionAsOGCWKT()

```
char* GXFGetMapProjectionAsOGCWKT (
            GXFHandle hGXF )
```

Return the GXF Projection in OpenGIS Well Known Text format.

The returned string becomes owned by the caller, and should be freed with CPLFree() or VSIFree(). The return value will be "" if no projection information is passed.

The mapping of GXF projections to OGC WKT format is not complete. Please see the gxf_ogcwkt.c code to better understand limitations of this translation. More information about OGC WKT format can be found in the Open↩ GIS Simple Features specification for OLEDB/COM found on the OpenGIS web site at www.opengis.org. The translation uses some code cribbed from the OGR library, about which more can be learned from http↩ ://gdal.velocet.ca/projects/opengis/.

For example, the following GXF definitions:

```
#UNIT_LENGTH
m,1
#MAP_PROJECTION
"NAD83 / UTM zone 19N"
"GRS 1980",6378137,0.081819191,0
"Transverse Mercator",0,-69,0.9996,500000,0
```

Would translate to (without the nice formatting):

```
PROJCS["NAD83 / UTM zone 19N",
       GEOGCS["GRS 1980",
              DATUM["GRS_1980",
                  SPHEROID["GRS 1980",6378137,298.257222413684]],
              PRIMEM["unnamed",0],
              UNIT["degree",0.0174532925199433]],
       PROJECTION["Transverse_Mercator"],
       PARAMETER["latitude_of_origin",0],
       PARAMETER["central_meridian",-69],
       PARAMETER["scale_factor",0.9996],
       PARAMETER["false_easting",500000],
       PARAMETER["false_northing",0],
       UNIT["m",1]]
```

Parameters

hGXF	handle to GXF file, as returned by GXFOpen().

Returns

string containing OGC WKT projection.

5.1.2.5 GXFGetMapProjectionAsPROJ4()

```
char* GXFGetMapProjectionAsPROJ4 (
            GXFHandle hGXF )
```

Return the GXF Projection in PROJ.4 format.

The returned string becomes owned by the caller, and should be freed with CPLFree() or VSIFree(). The return value will be "unknown" if no projection information is passed.

The mapping of GXF projections to PROJ.4 format is not complete. Please see the gxf_proj4.c code to better understand limitations of this translation. In particular, PROJ.4 knows little about datums.

For example, the following GXF definitions:

```
#UNIT_LENGTH
m,1
#MAP_PROJECTION
"NAD83 / UTM zone 19N"
"GRS 1980",6378137,0.081819191,0
"Transverse Mercator",0,-69,0.9996,500000,0
```

Would translate to:

```
+proj=tmerc +lat_0=0 +lon_0=-69 +k=0.9996 +x_0=500000 +y_0=0 +ellps=GRS80
```

Parameters

hGXF	handle to GXF file, as returned by GXFOpen().

Returns

string containing PROJ.4 projection.

5.1.2.6 GXFGetPosition()

```
CPLErr GXFGetPosition (
            GXFHandle hGXF,
            double * pdfXOrigin,
            double * pdfYOrigin,
            double * pdfXPixelSize,
            double * pdfYPixelSize,
            double * pdfRotation )
```

Get the grid positioning information.

Note that these coordinates refer to the grid positioning after taking into account the #SENSE flag (as is done by the GXFGetScanline()) function.

Note that the pixel values are considered to be point values in GXF, and thus the origin is for the first point. If you consider the pixels to be areas, then the origin is for the center of the origin pixel, not the outer corner.

This function does not support vertically oriented images, nor does it properly transform rotation for images with a SENSE other than GXFS_UL_RIGHT.

Parameters

hGXF	the GXF file handle.
pdfXOrigin	X position of the origin in the base coordinate system.
pdfYOrigin	Y position of the origin in the base coordinate system.
pdfXPixelSize	X pixel size in base coordinates.
pdfYPixelSize	Y pixel size in base coordinates.
pdfRotation	rotation in degrees counter-clockwise from the base coordinate system.

Returns

Returns CE_None if successful, or CE_Failure if no posiitioning information was found in the file.

5.1.2.7 GXFGetRawInfo()

```
CPLErr GXFGetRawInfo (
            GXFHandle hGXF,
            int * pnXSize,
            int * pnYSize,
            int * pnSense,
            double * pdfZMin,
            double * pdfZMax,
            double * pdfDummy )
```

Fetch header information about a GXF file.

Note that the X and Y sizes are of the raw raster and don't take into account the #SENSE flag. If the file is column oriented (rows in the files are actually columns in the raster) these values would need to be transposed for the actual raster.

The legal pnSense values are:

- GXFS_LL_UP(-1): lower left origin, scanning up.
- GXFS_LL_RIGHT(1): lower left origin, scanning right.
- GXFS_UL_RIGHT(-2): upper left origin, scanning right.
- GXFS_UL_DOWN(2): upper left origin, scanning down.
- GXFS_UR_DOWN(-3): upper right origin, scanning down.
- GXFS_UR_LEFT(3): upper right origin, scanning left.
- GXFS_LR_LEFT(-4): lower right origin, scanning left.
- GXFS_LR_UP(4): lower right origin, scanning up.

Note that the GXFGetScanline() function attempts to provide a GXFS_UL_RIGHT view onto files, but doesn't handle the *_DOWN and *_UP oriented files.

The Z min and max values may not occur in the GXF header. If they are requested, and aren't available in the header the entire file is scanned in order to establish them. This can be expensive.

If no #DUMMY value was specified in the file, a default of -1e12 is used.

Parameters

hGXF	handle to GXF file returned by GXFOpen().
pnXSize	int to be set with the width of the raw raster. May be NULL.
pnYSize	int to be set with the height of the raw raster. May be NULL.
pnSense	int to set with #SENSE flag, may be NULL.
pdfZMin	double to set with minimum raster value, may be NULL.
pdfZMax	double to set with minimum raster value, may be NULL.
pdfDummy	double to set with dummy (nodata / invalid data) pixel value.

5.1.2.8 GXFGetRawPosition()

```
CPLErr GXFGetRawPosition (
            GXFHandle hGXF,
            double * pdfXOrigin,
            double * pdfYOrigin,
            double * pdfXPixelSize,
            double * pdfYPixelSize,
            double * pdfRotation )
```

Get the raw grid positioning information.

Note that these coordinates refer to the raw grid, and are in the units specified by the #UNITS field. See GXFGetPosition() for a similar function that takes into account the #SENSE values similarly to GXFGetScanline().

Note that the pixel values are considered to be point values in GXF, and thus the origin is for the first point. If you consider the pixels to be areas, then the origin is for the center of the origin pixel, not the outer corner.

Parameters

hGXF	the GXF file handle.
pdfXOrigin	X position of the origin in the base coordinate system.
pdfYOrigin	Y position of the origin in the base coordinate system.
pdfXPixelSize	X pixel size in base coordinates.
pdfYPixelSize	Y pixel size in base coordinates.
pdfRotation	rotation in degrees counter-clockwise from the base coordinate system.

Returns

Returns CE_None if successful, or CE_Failure if no posiitioning information was found in the file.

5.1.2.9 GXFGetRawScanline()

```
CPLErr GXFGetRawScanline (
            GXFHandle hGXF,
```

```
            int iScanline,
            double * padfLineBuf )
```

Read a scanline of raster data from GXF file.

This function will read a row of data from the GXF file. It is "Raw" in the sense that it doesn't attempt to account for the #SENSE flag as the GXFGetScanline() function does. Unlike GXFGetScanline(), this function supports column organized files.

Any dummy pixels are assigned the dummy value indicated by GXFGetRawInfo().

Parameters

hGXF	the GXF file handle, as returned from GXFOpen().
iScanline	the scanline to read, zero is the first scanline in the file.
padfLineBuf	a buffer of doubles into which the scanline pixel values are read. This must be at least as long as a scanline.

Returns

CE_None if access succeeds or CE_Failure if something goes wrong.

5.1.2.10 GXFGetScanline()

```
CPLErr GXFGetScanline (
            GXFHandle hGXF,
            int iScanline,
            double * padfLineBuf )
```

Read a scanline of raster data from GXF file.

This function operates similarly to GXFGetRawScanline(), but it attempts to mirror data horizontally or vertically based on the #SENSE flag to return data in a top to bottom, and left to right organization. If the file is organized in columns (#SENSE is GXFS_UR_DOWN, GXFS_UL_DOWN, GXFS_LR_UP, or GXFS_LL_UP) then this function will fail, returning CE_Failure, and reporting a sense error.

See GXFGetRawScanline() for other notes.

Parameters

hGXF	the GXF file handle, as returned from GXFOpen().
iScanline	the scanline to read, zero is the top scanline.
padfLineBuf	a buffer of doubles into which the scanline pixel values are read. This must be at least as long as a scanline.

Returns

CE_None if access succeeds or CE_Failure if something goes wrong.

5.1.2.11 GXFOpen()

```
GXFHandle GXFOpen (
            const char * pszFilename )
```

Open a GXF file, and collect contents of the header.

Parameters

pszFilename	the name of the file to open.

Returns

a handle for use with other GXF functions to access the file. This will be NULL if the access fails.

Index