//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//

// License: LGPL

//

// See LICENSE.txt file in the top level directory for more details.

//

// Author: John Stastny

//

// Main function to call ossim filter which tiles image and runs something on it

//

//

#include <ossim/base/ossimArgumentParser.h>

#include <ossim/base/ossimApplicationUsage.h>

#include <ossim/base/ossimContainerProperty.h>

#include <ossim/base/ossimDatum.h>

#include <ossim/base/ossimDatumFactoryRegistry.h>

#include <ossim/base/ossimDrect.h>

#include <ossim/base/ossimEllipsoid.h>

#include <ossim/base/ossimException.h>

#include <ossim/base/ossimFilename.h>

#include <ossim/imaging/ossimFilterResampler.h>

#include <ossim/base/ossimGeoidManager.h>

#include <ossim/base/ossimGpt.h>

#include <ossim/init/ossimInit.h>

#include <ossim/base/ossimNotify.h>

#include <ossim/base/ossimPreferences.h>

#include <ossim/base/ossimProperty.h>

#include <ossim/base/ossimString.h>

#include <ossim/base/ossimTrace.h>

#include <ossim/base/ossimXmlDocument.h>

#include <ossim/elevation/ossimElevManager.h>

#include <ossim/imaging/ossimFilterResampler.h>

#include <ossim/imaging/ossimImageHandlerRegistry.h>

#include <ossim/imaging/ossimImageWriterFactoryRegistry.h>

#include <ossim/init/ossimInit.h>

#include <ossim/plugin/ossimSharedPluginRegistry.h>

#include <ossim/projection/ossimProjectionFactoryRegistry.h>

#include <ossim/support\_data/ossimInfoBase.h>

#include <ossim/support\_data/ossimInfoFactoryRegistry.h>

#include <ossim/support\_data/ossimSupportFilesList.h>

#include <ossim/base/ossimStdOutProgress.h>

#include <ossim/base/ossimTimer.h>

#include <ossim/imaging/ossimImageHandler.h>

#include <ossim/imaging/ossimImageRenderer.h>

#include <ossim/imaging/ossimSingleImageChain.h>

#include <ossim/imaging/ossimTiffWriter.h>

#include <vector>

#include <iostream>

#include "ossimTileToIplFilter.h"

using namespace std;

int main(int argc, char \*argv[])

{

if(argc != 2){

cout << "ossimTileToIplFilterApp.exe <input\_file>"<< endl;

return 0;

}

cout << "In main function" << endl;

// Initialize ossim

ossimInit::instance()->initialize(argc, argv);

// the image name

ossimFilename image\_file = argv[1];

// Declare ossim ImageHandler and open input image

ossimRefPtr<ossimImageHandler> ih = ossimImageHandlerRegistry::instance()->open(image\_file);

// Check to see if image handler is valid

if (ih.valid())

{

// Instantiate the rigorous model:

ossimRefPtr<ossimImageGeometry> geom = ih->getImageGeometry();

// Now print the geometry information for the file

cout << "Geometry: " << geom->print(cout) << endl;

cout << "Connecting handler to TileToIplFilter " << endl;

ossimRefPtr<ossimTileToIplFilter> TileToIpl = new ossimTileToIplFilter();

TileToIpl->connectMyInputTo(0,ih.get());

cout << "Connected!" << endl;

// Declare writer

ossimRefPtr<ossimImageSourceSequencer> sequencer = new ossimImageSourceSequencer();

sequencer->setToStartOfSequence();

// Connect to the TileToIpl

sequencer->connectMyInputTo(TileToIpl.get());

ossimRefPtr<ossimImageData> dataObject;

sequencer->setTileSize(ossimIpt(512,512));

cout << "Executing the chain..." << endl;

// Run through image tile by tile (execute filter chain.

while( (dataObject=sequencer->getNextTile()).valid() );

/\*while(1){

if(!sequencer->getNextTile().valid())

break;

}\*/

TileToIpl->disconnect();

sequencer->disconnect();

TileToIpl = 0;

sequencer = 0;

//delete TileToIpl ;

// osssim does this

//for us in the above code

}

cout << "Done!" << endl;

return 0;

} // End of main...