/\*\*

\* @license

\* Copyright 2019 Google LLC

\*

\* Licensed under the Apache License, Version 2.0 (the "License");

\* you may not use this file except in compliance with the License.

\* You may obtain a copy of the License at

\*

\* http://www.apache.org/licenses/LICENSE-2.0

\*

\* Unless required by applicable law or agreed to in writing, software

\* distributed under the License is distributed on an "AS IS" BASIS,

\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

\* See the License for the specific language governing permissions and

\* limitations under the License.

\*/

// Fetch a MODIS NDVI collection and select NDVI.

var col = ee.ImageCollection('MODIS/006/MOD13A2').select('NDVI');

// Define a mask to clip the NDVI data by.

var mask = ee.FeatureCollection('USDOS/LSIB\_SIMPLE/2017')

.filter(ee.Filter.eq('wld\_rgn', 'Africa'));

// Define the regional bounds of animation frames.

var region = ee.Geometry.Polygon(

[[[-18.698368046353494, 38.1446395611524],

[-18.698368046353494, -36.16300755581617],

[52.229366328646506, -36.16300755581617],

[52.229366328646506, 38.1446395611524]]],

null, false

);

// Add day-of-year (DOY) property to each image.

col = col.map(function(img) {

var doy = ee.Date(img.get('system:time\_start')).getRelative('day', 'year');

return img.set('doy', doy);

});

// Get a collection of distinct images by 'doy'.

var distinctDOY = col.filterDate('2013-01-01', '2014-01-01');

// Define a filter that identifies which images from the complete

// collection match the DOY from the distinct DOY collection.

var filter = ee.Filter.equals({leftField: 'doy', rightField: 'doy'});

// Define a join.

var join = ee.Join.saveAll('doy\_matches');

// Apply the join and convert the resulting FeatureCollection to an

// ImageCollection.

var joinCol = ee.ImageCollection(join.apply(distinctDOY, col, filter));

// Apply median reduction among matching DOY collections.

var comp = joinCol.map(function(img) {

var doyCol = ee.ImageCollection.fromImages(

img.get('doy\_matches')

);

return doyCol.reduce(ee.Reducer.median());

});

// Define RGB visualization parameters.

var visParams = {

min: 0.0,

max: 9000.0,

palette: [

'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163', '99B718', '74A901',

'66A000', '529400', '3E8601', '207401', '056201', '004C00', '023B01',

'012E01', '011D01', '011301'

],

};

// Create RGB visualization images for use as animation frames.

var rgbVis = comp.map(function(img) {

return img.visualize(visParams).clip(mask);

});

// Define GIF visualization arguments.

var gifParams = {

'region': region,

'dimensions': 600,

'crs': 'EPSG:3857',

'framesPerSecond': 10,

'format': 'gif'

};

// Print the GIF URL to the console.

print(rgbVis.getVideoThumbURL(gifParams));

// Render the GIF animation in the console.

print(ui.Thumbnail(rgbVis, gifParams));