

Google Earth Engine and machine learning for Earth monitoring

Alexander Uzhinskiy

Joint Institute for Nuclear Research, Joliot-Curie 6, 141980 Dubna, Russia, auzhinskiy@jinr.ru

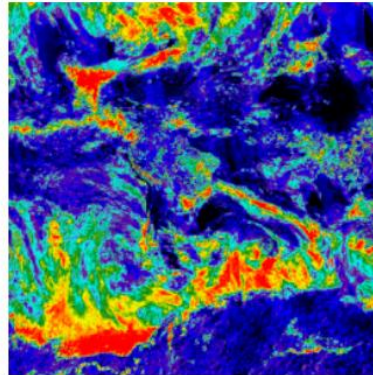
auzhinskiy@jinr.ru; Tel.: +79057766865

The Earth Engine Data Catalog

Datasets tagged climate in Earth Engine

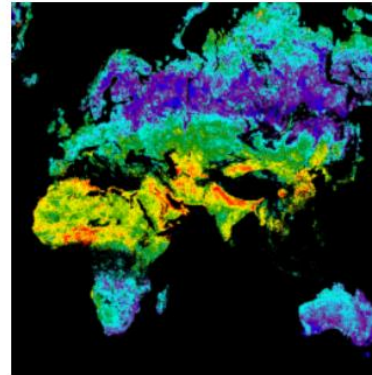
Filter list of datasets

Sentinel-5P NRTI CLOUD: Near Real-Time Cloud



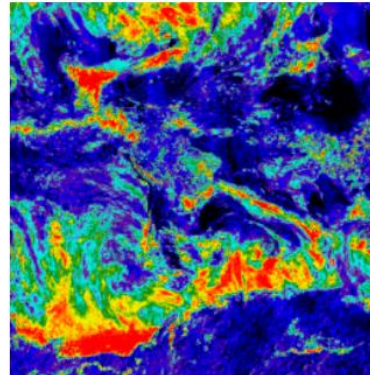
NRTI/L3_CLOUD This dataset provides near real-time high-resolution imagery of cloud parameters. The TROPOMI/S5P cloud properties retrieval is based on the OCRA and ROCINN algorithms currently being used in the operational GOME and GOME-2 products. OCRA retrieves the cloud fraction using measurements in the UV/VIS spectral ...

Sentinel-5P OFFL CH4: Offline Methane



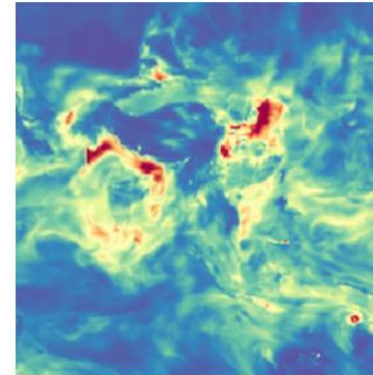
OFFL/L3_CH4 This dataset provides offline high-resolution imagery of methane concentrations. Methane (CH4) is, after carbon dioxide (CO2), the most important contributor to the anthropogenically enhanced greenhouse effect. Roughly three-quarters of methane emissions are anthropogenic and as such it is important to continue

Sentinel-5P OFFL CLOUD: Near Real-Time Cloud



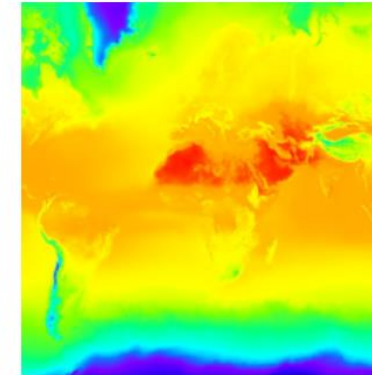
OFFL/L3_CLOUD This dataset provides offline high-resolution imagery of cloud parameters. The TROPOMI/S5P cloud properties retrieval is based on the OCRA and ROCINN algorithms currently being used in the operational GOME and GOME-2 products. OCRA retrieves the cloud fraction using measurements in the UV/VIS spectral regions ...

Copernicus Atmosphere Monitoring Service (CAMS) Global Near-Real-Time



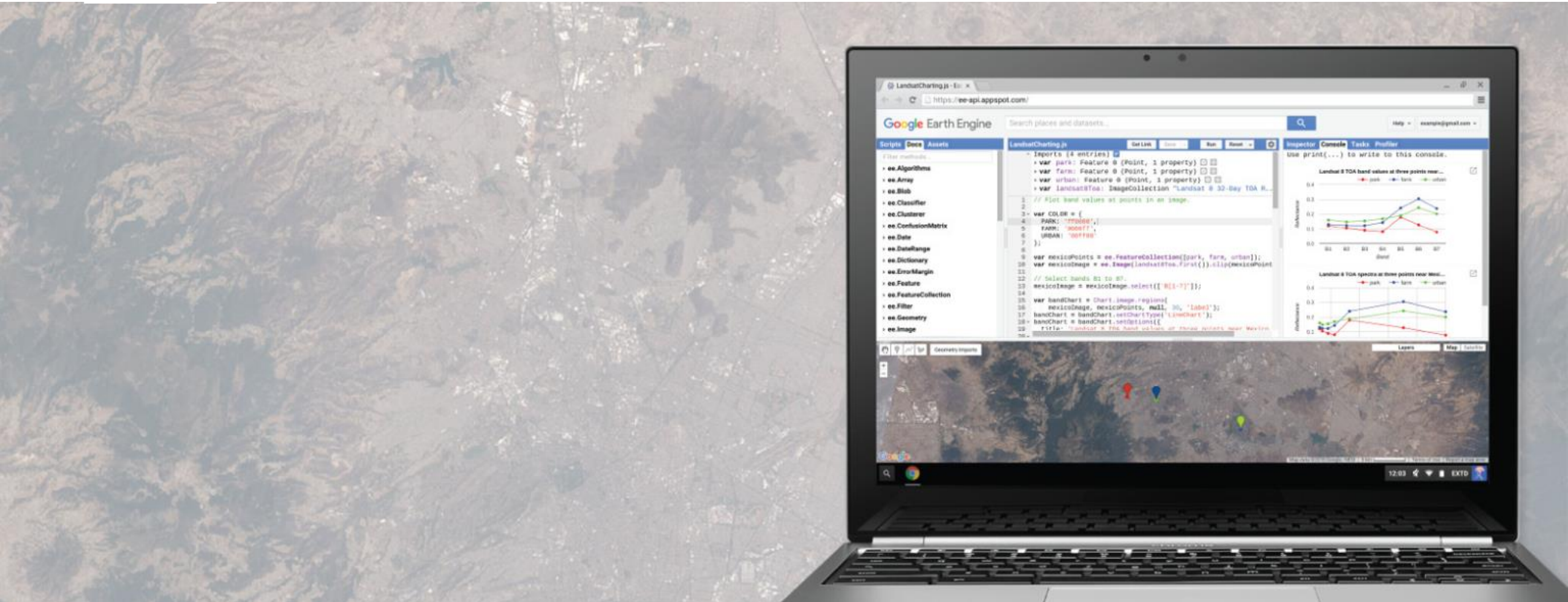
The Copernicus Atmosphere Monitoring Service provides the capacity to continuously monitor the composition of the Earth's atmosphere at global and regional scales. The main global near-real-time production system is a data assimilation and forecasting suite providing two 5-day forecasts per day for aerosols and chemical ...

ERA5 Daily Aggregates - Latest Climate Reanalysis Produced by ECMWF / Copernicus Climate



ERA5 is the fifth generation ECMWF atmospheric reanalysis of the global climate. Reanalysis combines model data with observations from across the world into a globally complete and consistent dataset. ERA5 replaces its predecessor, the ERA-Interim reanalysis. ERA5 DAILY provides aggregated values for each day for ...

JavaScript code editor <https://code.earthengine.google.com/>



* python too!

<https://developers.google.com/>

<https://developers.google.com/earth-engine/datasets/catalog/>

<https://code.earthengine.google.com/>

<https://explorer.earthengine.google.com/>

The image shows a screenshot of the Google Earth Engine playground interface. The interface is divided into several sections:

- Search for datasets or places:** A search bar at the top with a magnifying glass icon.
- Script manager:** A sidebar on the left with a search bar and a 'NEW' button. It lists various scripts under categories like 'Owner (16)', 'Writer', 'Reader', and 'Examples'.
- API documentation:** A link in the top navigation bar.
- Asset manager:** A link in the top navigation bar.
- Code Editor:** A central area with a code editor showing JavaScript code for a Sentinel-2 cloud mask script. The code includes comments and function definitions for cloud mask generation and data masking.
- Inspector:** A panel on the right for inspecting map elements.
- Console:** A panel on the right for viewing console output.
- Tasks:** A panel on the right for managing tasks.
- Map:** A satellite map at the bottom showing a coastal area. It includes a 'Layers' panel, a 'Map' button, and a 'Satellite' button.
- Geometry Tools:** A toolbar on the left side of the map.
- Zoom:** A zoom control on the left side of the map.

Arrows point from text labels to these specific components in the interface.

Basics

<https://code.earthengine.google.com/6b24f55c3c60991a09352ce552fd310a>

Scripts Docs Assets

Filter scripts... NEW Refresh

Owner (1)

- users/zmeijska/default
 - Image Collection
 - experiment
 - Ivanovo
 - Poland
 - Sweden
 - ndvi
 - ndvi_calculation
 - razm2
 - razmeri
 - romania

New Script * Get Link Save Run Reset Apps Settings

```

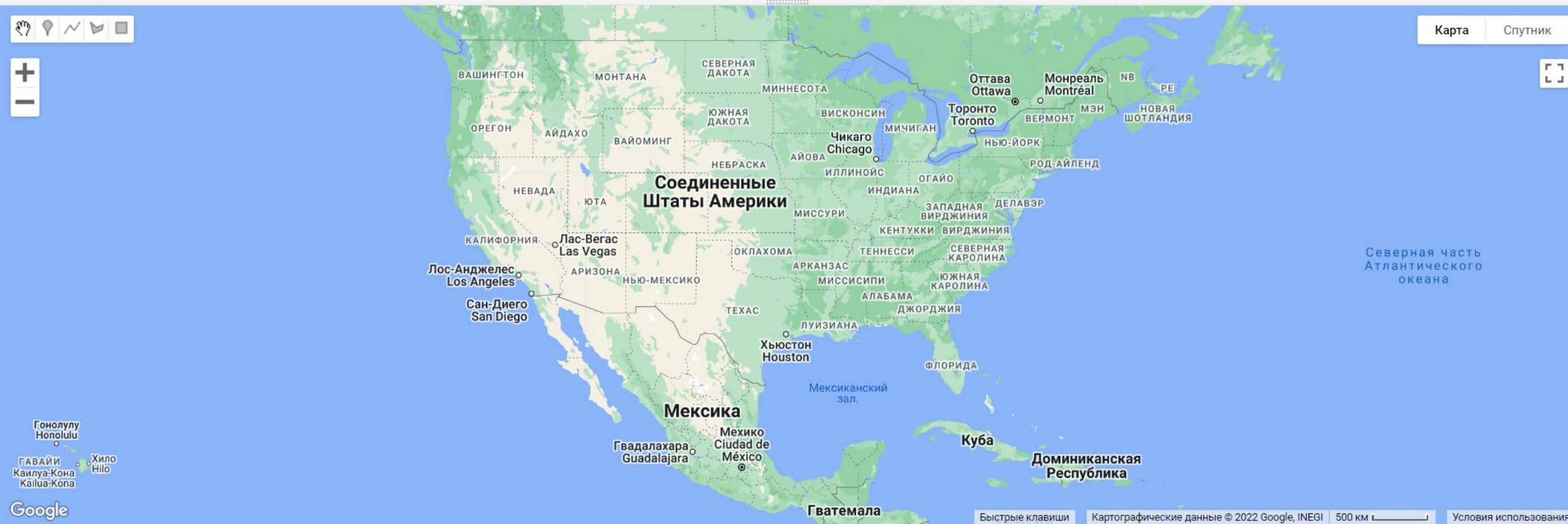
1 // Use single (or double) quotes to make a string.
2 var greetString = 'Ahoy there!';
3 // Use parentheses to pass arguments to functions.
4 print(greetString);
5
6 // Store a number in a variable.
7 var number = 42;
8 print('The answer is:', number);
9
10 // Use square brackets [] to make a list.
11 var listOfNumbers = [0, 1, 1, 2, 3, 5];
12 print('List of numbers:', listOfNumbers);
13
14

```

Inspector Console Tasks

Use print(...) to write to this console.

Ahoy there!	JSON
The answer is: 42	JSON
List of numbers: ▶ [0,1,1,2,3,5]	JSON



Scripts Docs Assets

Filter scripts... NEW Refresh

Owner (1)

- users/zmeijska/default
 - Image Collection
 - experiment
 - Ivanovo
 - Poland
 - Sweden
 - ndvi
 - ndvi_calculation
 - razm2
 - razmeri
 - romania
 - tutorial

New Script * Get Link Save Run Reset Apps Settings

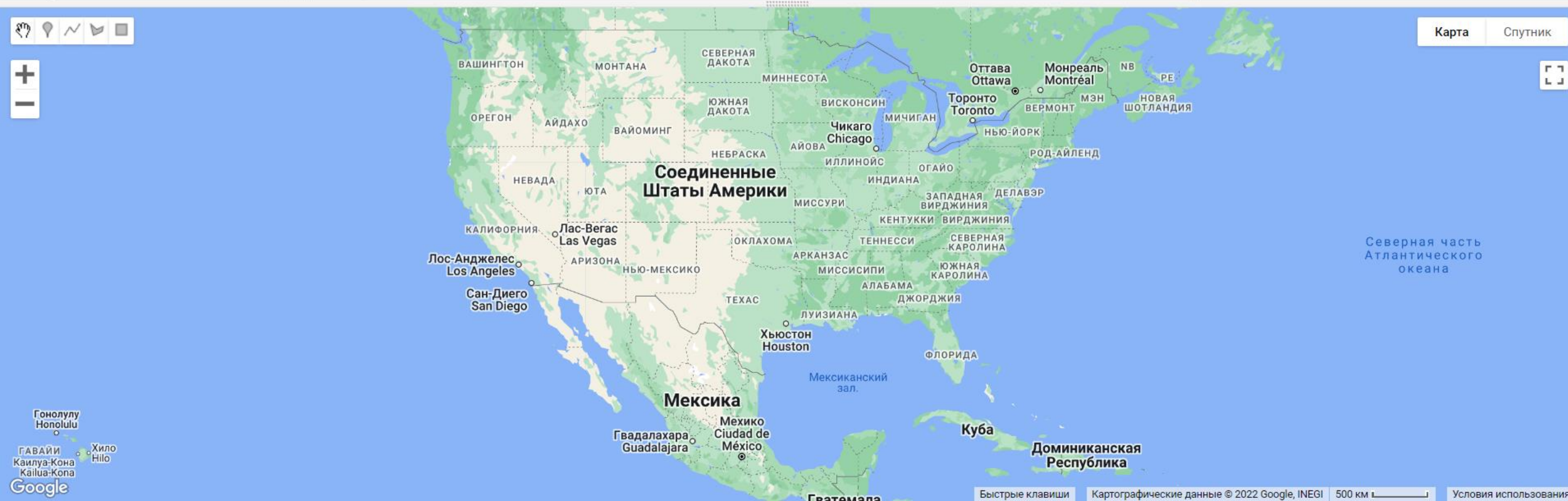
```

1 // Use curly brackets {} to make a dictionary of key:value pairs.
2 var object = {
3   foo: 'bar',
4   baz: 13,
5   stuff: ['this', 'that', 'the other thing']
6 };
7 print('Dictionary:', object);
8 // Access dictionary items using square brackets.
9 print('Print foo:', object['foo']);
10 // Access dictionary items using dot notation.
11 print('Print stuff:', object.stuff);
  
```

Inspector Console Tasks

Use print(...) to write to this console.

Dictionary:	JSON
Object (3 properties)	JSON
Print foo:	JSON
bar	JSON
Print stuff:	JSON
["this","that","the other thing"]	JSON



Scripts Docs Assets

Filter scripts... NEW Refresh

Owner (1)

- users/zmeijska/default
 - Image Collection
 - experiment
 - Ivanovo
 - Poland
 - Sweden
 - ndvi
 - ndvi_calculation
 - razm2
 - razmeri
 - romania
 - tutorial

New Script *

Get Link Save Run Reset Apps

```

1 // The reflect function takes a single parameter: element.
2 var reflect = function(element) {
3   // Return the argument.
4   return element;
5 };
6 print('A good day to you!', reflect('Back at you!'));

```

Inspector Console Tasks

Use print(...) to write to this console.

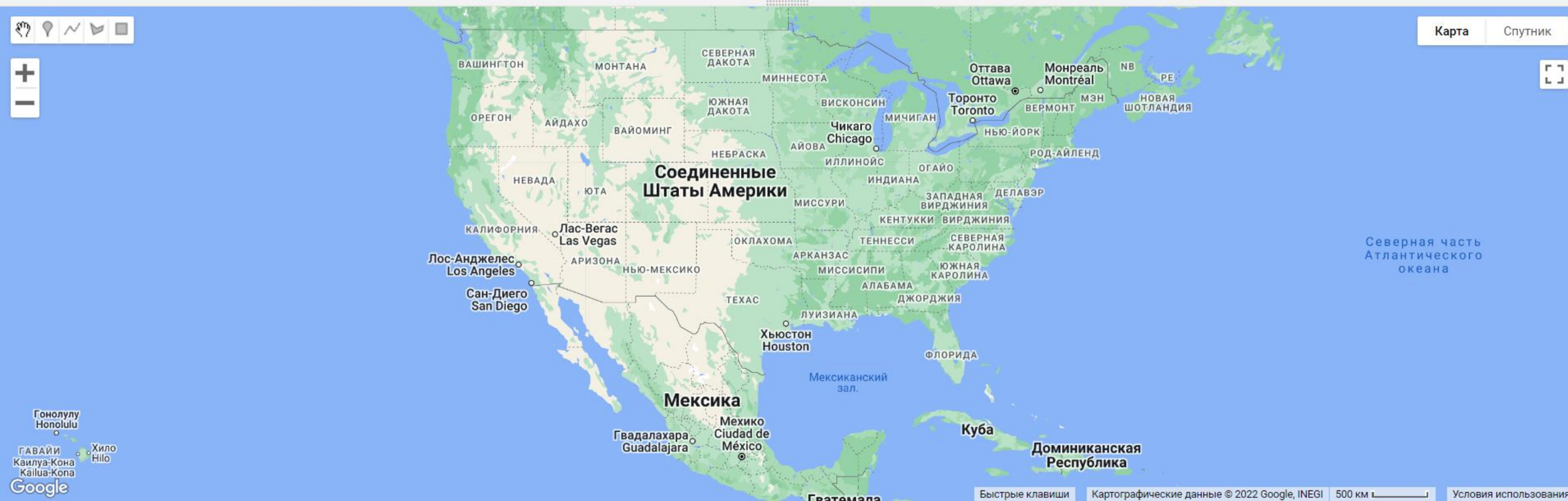
```

A good day to you!
Back at you!

```

JSON

JSON



Scripts Docs Assets

Filter scripts... NEW Refresh

Owner (1)

- users/zmeijska/default
 - Image Collection
 - experiment
 - Ivanovo
 - Poland
 - Sweden
 - ndvi
 - ndvi_calculation
 - razm2
 - razmeri
 - romania
 - tutorial

New Script * Get Link Save Run Reset Apps Settings

```

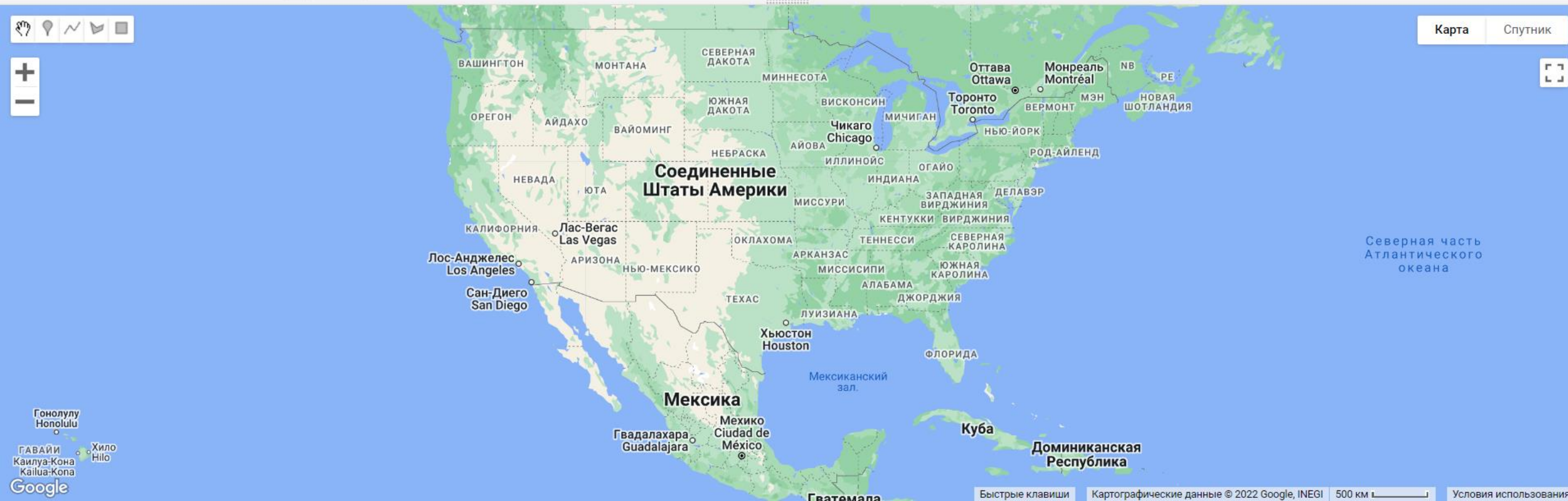
1 // Define a date in Earth Engine.
2 var date = ee.Date('2015-12-31');
3 print('Date:', date);
4
5 // Get the current time using the JavaScript Date.now() method.
6 var now = Date.now();
7 print('Milliseconds since January 1, 1970', now);
8
9 // Initialize an ee.Date object.
10 var eeNow = ee.Date(now);
11 print('Now:', eeNow);

```

Inspector Console Tasks

Use print(...) to write to this console.

Date:	JSON
▶ Date (2015-12-31 00:00:00)	JSON
Milliseconds since January 1, 1970	JSON
1657653540108	
Now:	JSON
▶ Date (2022-07-12 19:19:00)	JSON



Data Catalog

<https://developers.google.com/earth-engine/datasets/catalog/>

Earth Engine Data Catalog

Search Language

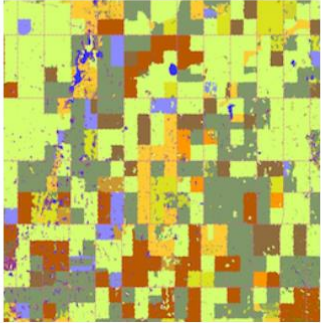

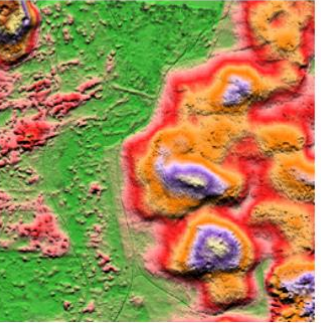
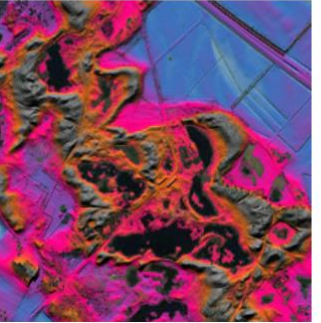

Home View all datasets Browse by tags Landsat MODIS Sentinel API Docs

Earth Engine Data Catalog

Earth Engine's public data catalog includes a variety of standard Earth science raster datasets. You can import these datasets into your script environment with a single click. You can also upload your own [raster data](#) or vector data for private use or sharing in your scripts.

Looking for another dataset not in Earth Engine yet? Let us know by [suggesting a dataset](#).

Filter list of datasets

Canada AAFC Annual Crop Inventory	Allen Coral Atlas (ACA) - Geomorphic Zonation and Benthic Habitat - v1.0	AHN Netherlands 0.5m DEM, Interpolated	AHN Netherlands 0.5m DEM, Non-Interpolated	AHN Netherlands 0.5m DEM, Raw Samples
				
Starting in 2009, the Earth Observation Team of the Science and Technology Branch (STB) at Agriculture and Agri-Food Canada (AAFC) began the process of generating annual crop type digital maps. Focusing on the Prairie Provinces	The Allen Coral Atlas dataset maps the geomorphic zonation and benthic habitat for the world's shallow coral reefs at 5m pixel resolution. The underlying satellite image data are temporal composites of PlanetScope	The AHN DEM is a 0.5m DEM covering the Netherlands. It was generated from LIDAR data taken in the spring between 2007 and 2012. It contains ground level samples with all other items above ground (such as buildings, bridges, trees	The AHN DEM is a 0.5m DEM covering the Netherlands. It was generated from LIDAR data taken in the spring between 2007 and 2012. It contains ground level samples with all other items above ground (such as buildings, bridges, trees	The AHN DEM is a 0.5m DEM covering the Netherlands. It was generated from LIDAR data taken in the spring between 2007 and 2012. This version contains both ground level samples and items above ground level (such as buildings,

Images

https://developers.google.com/earth-engine/datasets/catalog/CGIAR_SRTM90_V4?hl=en

<https://code.earthengine.google.com/31f672aec5b2a3f8bc7d1fbf23450935>

```
// Instantiate an image with the Image constructor.
```

```
var image = ee.Image('CGIAR/SRTM90_V4');
```

```
// Zoom to a location.
```

```
Map.setCenter(37.02540746271644,56.766160331658845, 12);
```

```
Map.addLayer(image);
```

```
//Map.addLayer(image, {min: 50, max: 180});
```

```
//Map.addLayer(image, {min: 50, max: 180, palette: ['blue', 'green', 'red']}, 'custom palette');
```

Scripts Docs Assets

Filter scripts... NEW Refresh

Owner (1)

- users/zmeiyska/default
 - Image Collection
 - experiment
 - lip
 - basics
 - image Refresh Edit Delete
 - Ivanovo
 - Poland

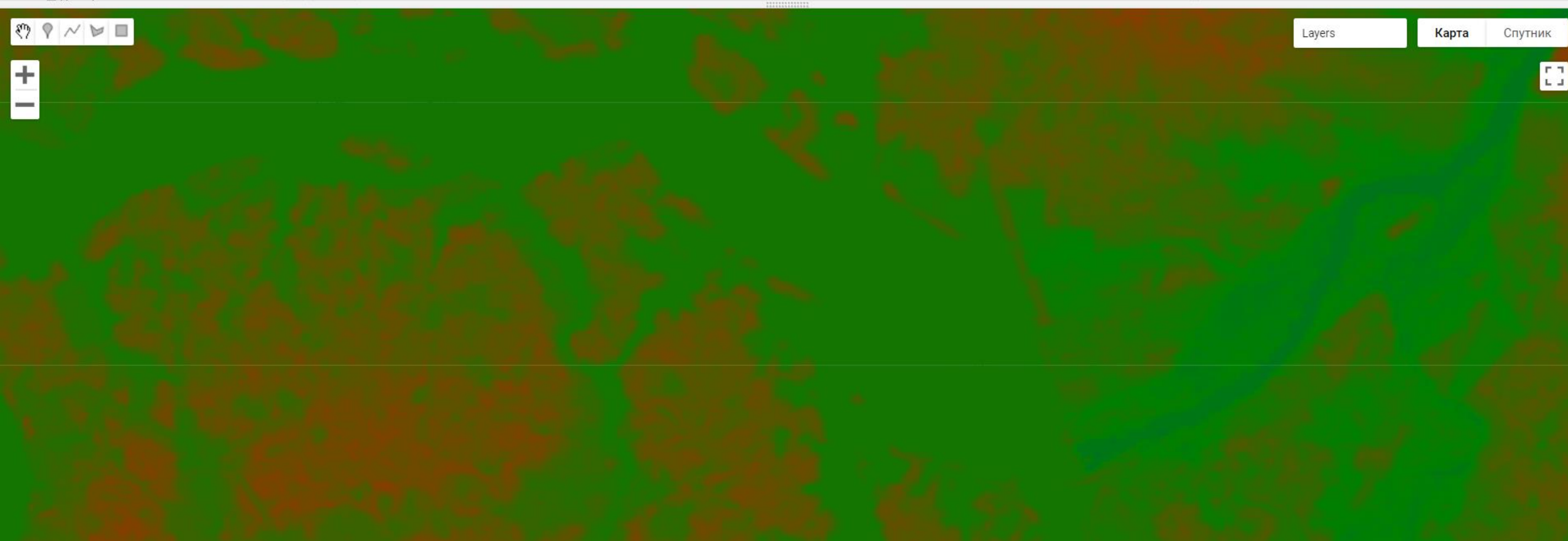
lip/image * Get Link Save Run Reset Apps Settings

```

1 // Instantiate an image with the Image constructor.
2 var image = ee.Image('CGIAR/SRTM90_V4');
3
4 // Zoom to a location.
5 Map.setCenter(37.02540746271644,56.766160331658845, 12);
6
7 //Map.addLayer(image);
8 //Map.addLayer(image, {min: 50, max: 180});
9 Map.addLayer(image, {min: 50, max: 180, palette: ['blue', 'green', 'red']}, 'custom palette');
```

Inspector Console Tasks

Use print (...) to write to this console.



Images computation

<https://code.earthengine.google.com/a4b27505dfd223c7f5878e97ca42d06b>

<https://developers.google.com/earth-engine/apidocs/ee-terrain-slope>

```
// Instantiate an image with the Image constructor.  
var image = ee.Image('CGIAR/SRTM90_V4');  
  
// Apply an algorithm to an image.  
var slope = ee.Terrain.slope(image);  
  
// Zoom to a location.  
Map.setCenter(37.02540746271644, 56.766160331658845, 12);  
  
Map.addLayer(slope, {min: 0, max :5}, 'slope');
```

Scripts Docs Assets




- experiment
 - lip
 - basics
 - collections
 - image
 - Ivanovo
 - Poland
 - Sweden
 - image comp   
 - ndvi
 - ndvi_calculation
 - razm2

image comp * Get Link Save Run Reset Apps 

```

1 // Instantiate an image with the Image constructor.
2 var image = ee.Image('CGIAR/SRTM90_V4');
3
4 // Apply an algorithm to an image.
5 var slope = ee.Terrain.slope(image);
6
7 // Zoom to a location.
8 Map.setCenter(37.02540746271644,56.766160331658845, 12);
9
10 Map.addLayer(slope, {min: 0, max :5}, 'slope');|

```

Inspector Console Tasks

Use print(...) to write to this console.

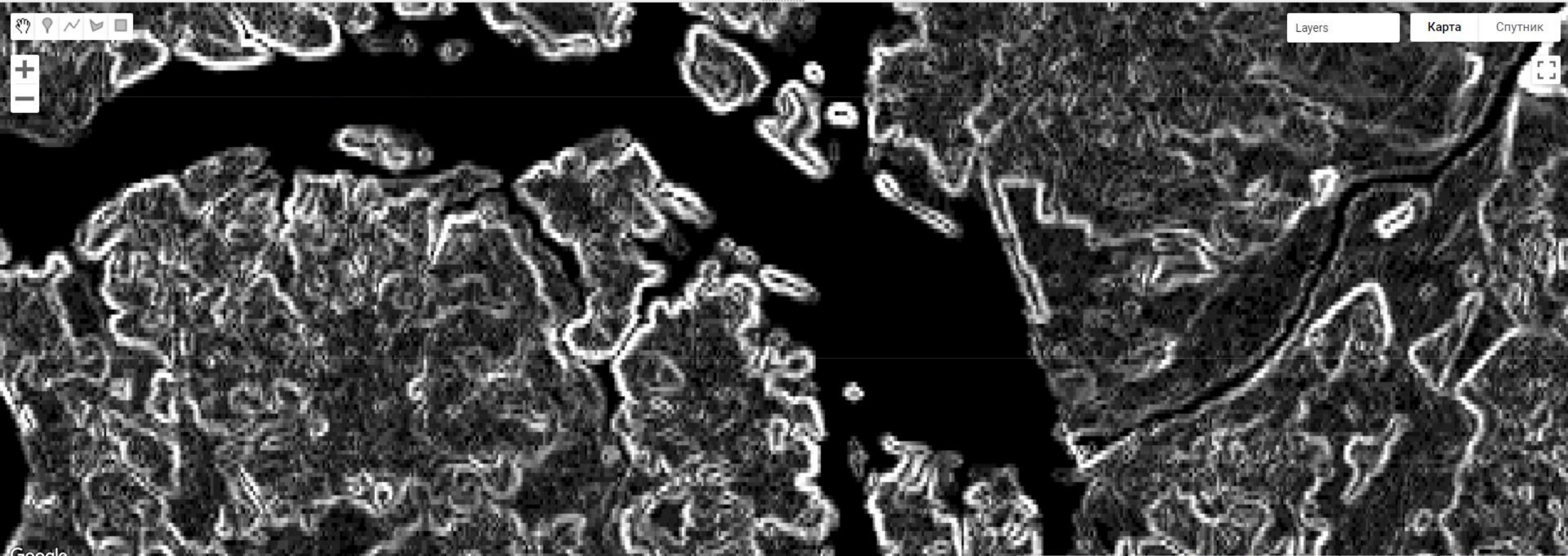


Image statistics

<https://code.earthengine.google.com/37d0c570ad54bf3d0232386d868be168>

```
// Instantiate an image with the Image constructor.
var image = ee.Image('CGIAR/SRTM90_V4');

// Zoom to a location.
Map.setCenter(37.02540746271644,56.766160331658845, 12);
var p1 = ee.Geometry.Point(37.02540746271644,56.766160331658845);
var p2 = ee.Geometry.Point(37.03502049982581,56.770864050772836);

var mean = image.reduceRegion({
  reducer: ee.Reducer.mean(),
  geometry: p1,
  scale: 90
});

print(mean);

var mean = image.reduceRegion({
  reducer: ee.Reducer.mean(),
  geometry: p2,
  scale: 90
});

print(mean);

Map.addLayer(image, {min: 50, max: 180, palette: ['blue', 'green', 'red']}, 'custom palette');
Map.addLayer(p1);
Map.addLayer(p2);
```


Scripts Docs Assets

- experiment
 - lip
 - basics
 - collections
 - image
 - image comp
 - image statistics
 - Ivanovo
 - Poland
 - Sweden
 - ndvi
 - ndvi calculation

```

image statistics *
1 // Instantiate an image with the Image constructor.
2 var image = ee.Image('CGIAR/SRTM90_V4');
3
4 // Zoom to a location.
5 Map.setCenter(37.02540746271644, 56.766160331658845, 12);
6 var p1 = ee.Geometry.Point(37.02540746271644, 56.766160331658845);
7 var p2 = ee.Geometry.Point(37.03502049982581, 56.770864050772836);
8
9 var mean = image.reduceRegion({
10   reducer: ee.Reducer.mean(),
11   geometry: p1,
12   scale: 30
13 });
14

```

Inspector Console Tasks

Use print(...) to write to this console.

- Object (1 property) JSON
 - elevation: 134
- Object (1 property) JSON
 - elevation: 120

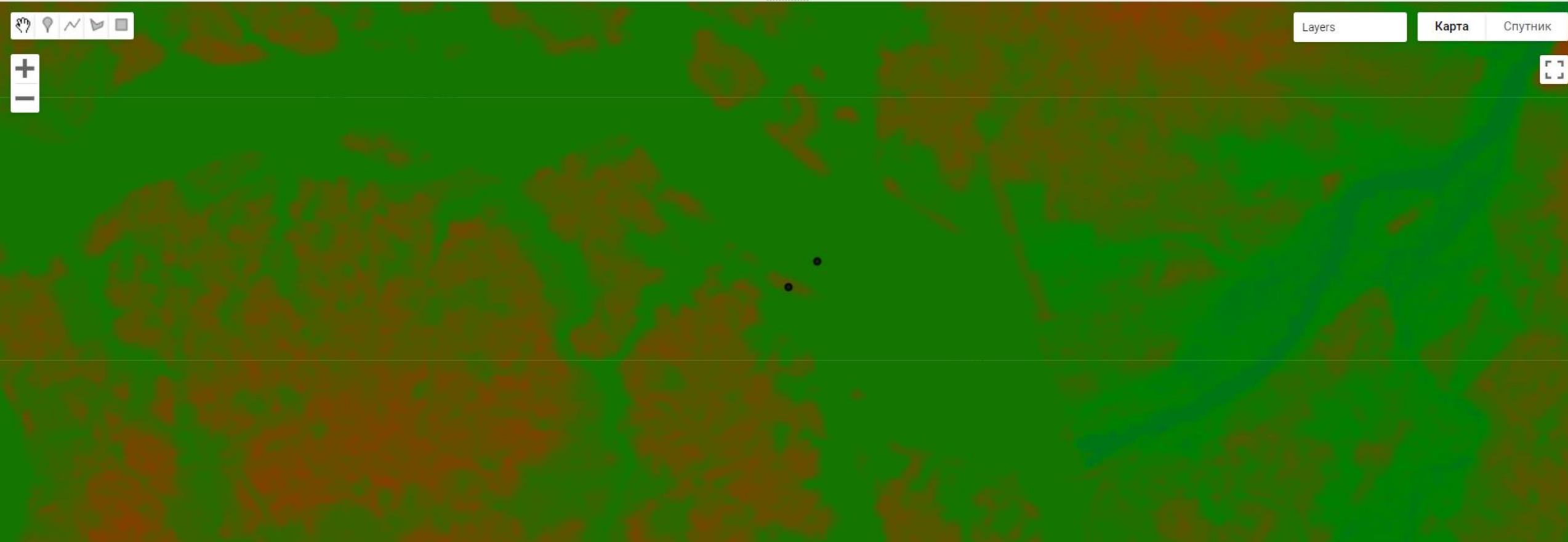


Image Collections

https://developers.google.com/earth-engine/datasets/catalog/LANDSAT_LC08_C02_T1_TOA

<https://code.earthengine.google.com/92fc966d883b11f9850b36504e6ea5eb>

```
var point = ee.Geometry.Point(37.02540746271644,56.766160331658845);  
var l8 = ee.ImageCollection('LANDSAT/LC08/C02/T1_TOA');
```

```
var spatialFiltered = l8.filterBounds(point);  
print('spatialFiltered', spatialFiltered);
```

```
var temporalFiltered = spatialFiltered.filterDate('2022-06-01', '2022-07-17');  
print('temporalFiltered', temporalFiltered);
```

```
// This will sort from least to most cloudy.  
var sorted = temporalFiltered.sort('CLOUD_COVER');
```

```
// Get the first (least cloudy) image.  
var scene = sorted.first();
```

```
Map.setCenter(37.02540746271644,56.766160331658845, 12);  
var visParams = {bands: ['B4', 'B3', 'B2'], max: 0.3};  
Map.addLayer(scene, visParams, 'true-color composite less clouds');  
Map.addLayer(temporalFiltered, visParams, 'true-color composite');
```

- Scripts
- Docs
- Assets
 - classification
 - collection clouds
 - collection mask
 - collection median
 - collection ndvi
 - collections
 - fires
 - image
 - image comp
 - image statistics
 - timeseries
- Ivanovo
- Poland
- Sweden

```

collections
  Get Link Save Run Reset Apps
1 var point = ee.Geometry.Point(37.02540746271644, 56.766160331658845);
2 var l8 = ee.ImageCollection('LANDSAT/LC08/C02/T1_TOA');
3
4 var spatialFiltered = l8.filterBounds(point);
5 print('spatialFiltered', spatialFiltered);
6
7 var temporalFiltered = spatialFiltered.filterDate('2022-06-01', '2022-07-17');
8 print('temporalFiltered', temporalFiltered);
9
10 // This will sort from least to most cloudy.
11 var sorted = temporalFiltered.sort('CLOUD_COVER');
12
13 // Get the first (least cloudy) image.
14 var scene = sorted.first();
15
16 Map.setCenter(37.02540746271644, 56.766160331658845, 12);
17 var visParams = {bands: ['B4', 'B3', 'B2'], max: 0.3};

```

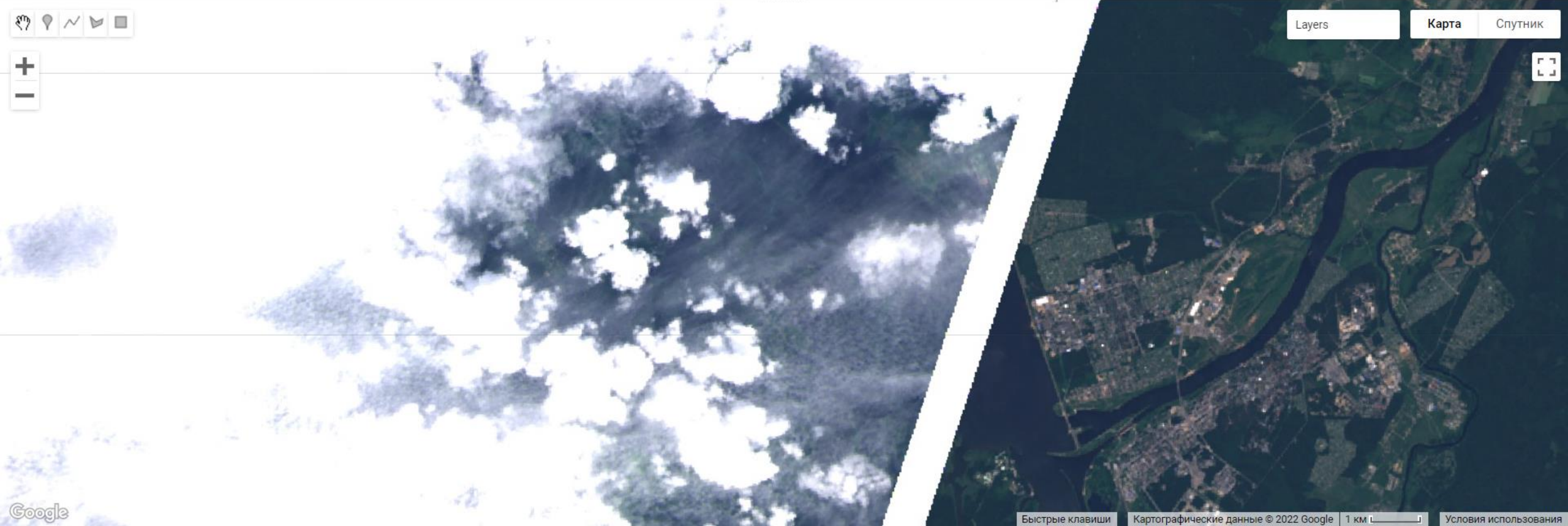
Inspector Console Tasks

Use print(...) to write to this console.

```

spatialFiltered
  ImageCollection LANDSAT/LC08/C02/T1_TOA (464 elements, 17 band...
temporalFiltered
  ImageCollection LANDSAT/LC08/C02/T1_TOA (5 elements, 17 bands)

```



Median image

<https://code.earthengine.google.com/e06d18ab4bda8ae5c6500407d8bd97d5>

```
var l8 = ee.ImageCollection('LANDSAT/LC08/C02/T1_TOA').filterDate('2022-06-01', '2022-07-17');
```

```
Map.addLayer(l8, {bands: ['B4', 'B3', 'B2'], max: 0.3}, 'l8 collection');
```

```
// Get the median over time, in each band, in each pixel.
```

```
var median = l8.median();
```

```
Map.addLayer(median, {bands: ['B4', 'B3', 'B2'], max: 0.3}, 'l8 median');
```

```
Map.setCenter(37.02540746271644, 56.766160331658845, 12);
```

Scripts Docs Assets

- lip
 - basics
 - collection composi...
 - collections
 - image
 - image comp
 - image statistics
- Ivanovo
- Poland
- Sweden
- ndvi

collection composite *

Get Link Save Run Reset Apps

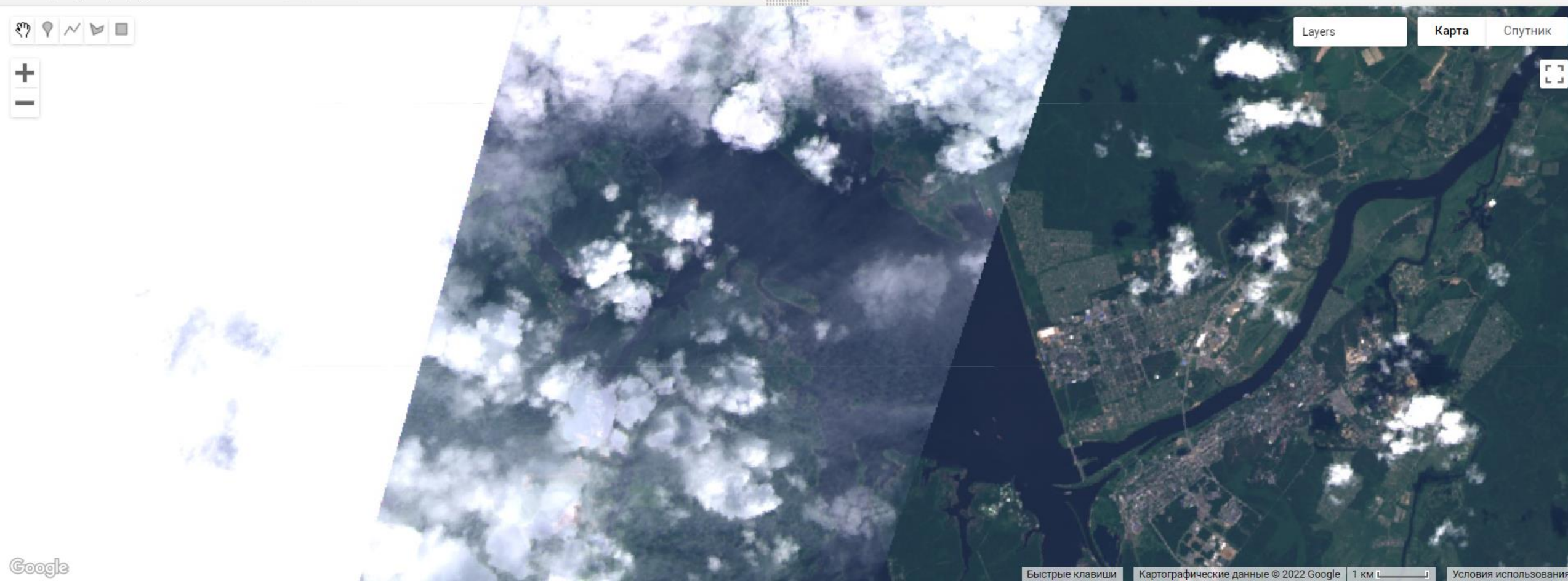
```

1 var l8 = ee.ImageCollection('LANDSAT/LC08/C02/T1_TOA').filterDate('2022-06-01', '2022-07-17');
2
3 Map.addLayer(l8, {bands: ['B4', 'B3', 'B2'], max: 0.3}, 'l8 collection');
4
5 // Get the median over time, in each band, in each pixel.
6 var median = l8.median();
7 Map.addLayer(median, {bands: ['B4', 'B3', 'B2'], max: 0.3}, 'l8 median');
8 Map.setCenter(37.02540746271644, 56.766160331658845, 12);
9

```

Inspector Console Tasks

Use print(...) to write to this console.



Masking

<https://developers.google.com/earth-engine/apidocs/ee-image-updatemask>

https://developers.google.com/earth-engine/datasets/catalog/UMD_hansen_global_forest_change_2021_v1_9?hl=en

<https://code.earthengine.google.com/976753b4bfb128c1cc8293ac60fe1b3f>

```
var l8 = ee.ImageCollection('LANDSAT/LC08/C02/T1_TOA').filterDate('2022-06-01', '2022-07-17');
```

```
// Get the median over time, in each band, in each pixel.
```

```
var median = l8.median();
```

```
Map.addLayer(median, {bands: ['B4', 'B3', 'B2'], max: 0.3}, 'l8 median');
```

```
Map.setCenter(37.02540746271644, 56.766160331658845, 12);
```

```
// Load or import the Hansen et al. forest change dataset.
```

```
var hansenImage = ee.Image('UMD/hansen/global_forest_change_2021_v1_9');
```

```
// Select the land/water mask.
```

```
var datamask = hansenImage.select('datamask');
```

```
// Create a binary mask.
```

```
var mask = datamask.eq(1);
```

```
// Update the composite mask with the water mask.
```

```
var maskedComposite = median.updateMask(mask);
```

```
Map.addLayer(maskedComposite, {bands: ['B4', 'B3', 'B2'], max: 0.3}, 'masked');
```

Scripts Docs Assets

- experiment
 - lip
 - basics
 - collection mask
 - collection median
 - collections
 - image
 - image comp
 - image statistics
 - Ivanovo
 - Poland

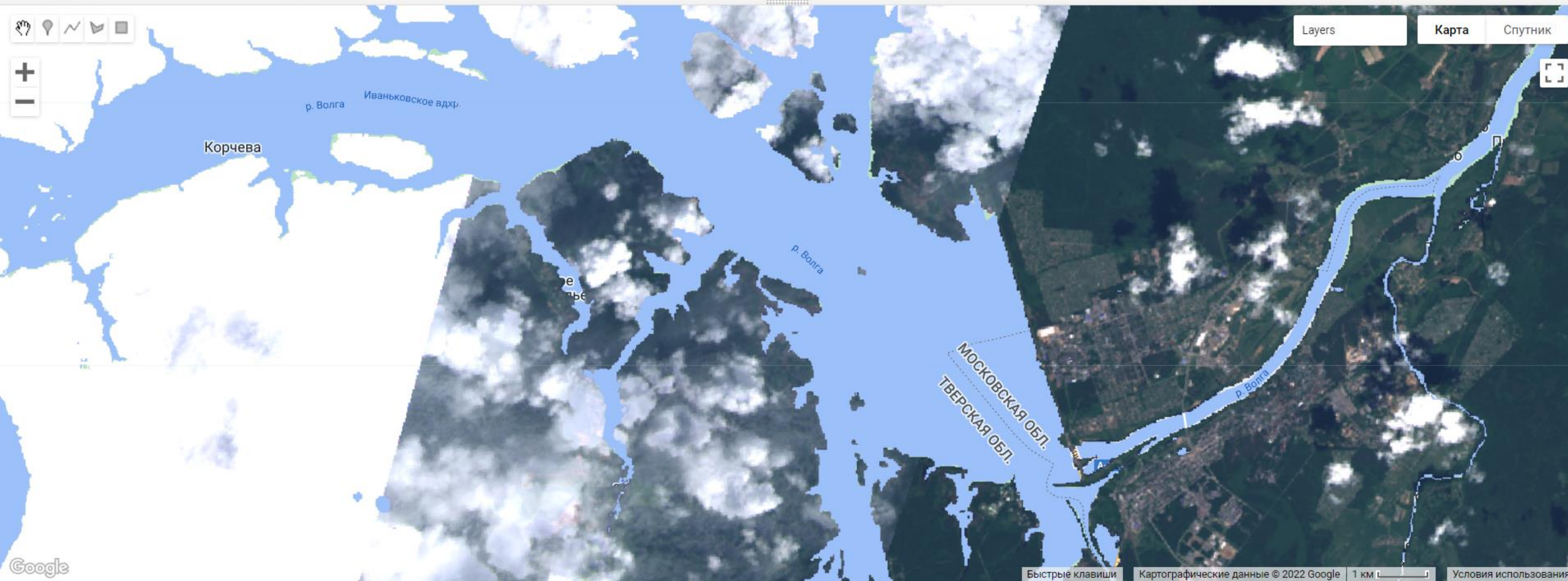
collection mask *

```

6 Map.setCenter(37.02540746271644, 56.766160331658845, 12);
7
8 // Load or import the Hansen et al. forest change dataset.
9 var hansenImage = ee.Image('UMD/hansen/global_forest_change_2021_v1_9');
10
11 // Select the land/water mask.
12 var datamask = hansenImage.select('datamask');
13
14 // Create a binary mask.
15 var mask = datamask.eq(1);
16
17 // Update the composite mask with the water mask.
18 var maskedComposite = median.updateMask(mask);
19 Map.addLayer(maskedComposite, {bands: ['B4', 'B3', 'B2'], max: 0.3}, 'masked');
```

Inspector Console Tasks

Use print(...) to write to this console.



Clouds

<https://code.earthengine.google.com/36ce292efce7e367f53aac53ed970fbf>

https://developers.google.com/earth-engine/datasets/catalog/LANDSAT_LC08_C02_T1_L2#description

```
var dataset = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')
  .filterDate('2022-05-01', '2022-07-17');

// Applies scaling factors.
function applyScaleFactors(image) {
  var opticalBands = image.select('SR_B.').multiply(0.0000275).add(-0.2);
  var thermalBands = image.select('ST_B.*').multiply(0.00341802).add(149.0);
  return image.addBands(opticalBands, null, true)
    .addBands(thermalBands, null, true);
}

dataset = dataset.map(applyScaleFactors);

var visualization = {
  bands: ['SR_B4', 'SR_B3', 'SR_B2'],
  min: 0.0,
  max: 0.3,
};

Map.setCenter(37.02540746271644, 56.766160331658845, 12);

Map.addLayer(dataset, visualization, 'clouds');
```

```
function maskL8sr(image) {
  // Bit 0 - Fill
  // Bit 1 - Dilated Cloud
  // Bit 2 - Cirrus
  // Bit 3 - Cloud
  // Bit 4 - Cloud Shadow
  var qaMask = image.select('QA_PIXEL').bitwiseAnd(parseInt('11111', 2)).eq(0);
  var saturationMask = image.select('QA_RADSAT').eq(0);

  // Apply the scaling factors to the appropriate bands.
  var opticalBands = image.select('SR_B.').multiply(0.0000275).add(-0.2);

  // Replace the original bands with the scaled ones and apply the masks.
  return image.addBands(opticalBands, null, true)
    .updateMask(qaMask)
    .updateMask(saturationMask);
}

// Map the function over one year of data.
var collection = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')
  .filterDate('2022-05-01', '2022-07-17')
  .map(maskL8sr);

var composite = collection.median();

// Display the results.
Map.addLayer(composite, {bands: ['SR_B4', 'SR_B3', 'SR_B2'], min: 0, max: 0.3}, 'no clouds');
```


Scripts Docs Assets

- experiment
 - lip
 - basics
 - collection clouds
 - collection mask
 - collection median
 - collections
 - image
 - image comp
 - image statistics
 - Ivanovo

collection clouds *

Get Link Save Run Reset Apps

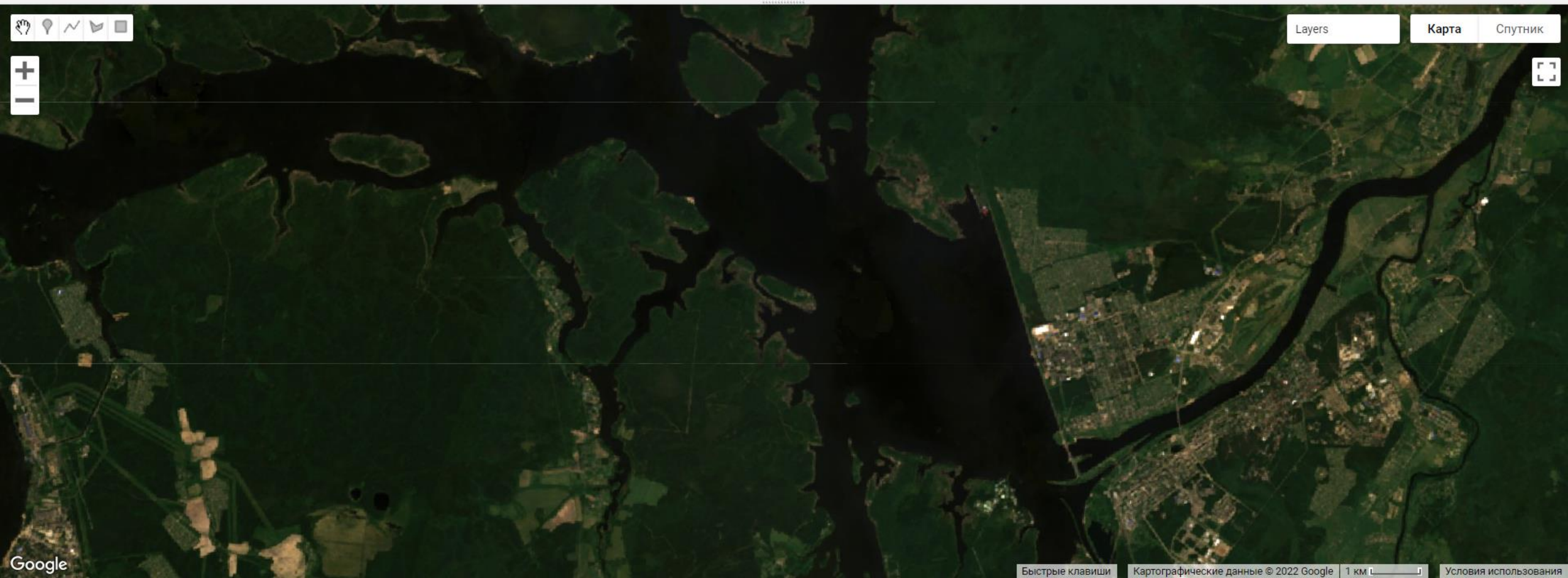
```

1 var dataset = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')
2   .filterDate('2022-05-01', '2022-07-17');
3
4
5 // Applies scaling factors.
6 function applyScaleFactors(image) {
7   var opticalBands = image.select('SR_B.').multiply(0.0000275).add(-0.2);
8   var thermalBands = image.select('ST_B.*').multiply(0.00341802).add(149.0);
9   return image.addBands(opticalBands, null, true)
10     .addBands(thermalBands, null, true);
11 }
12
13 dataset = dataset.map(applyScaleFactors);
14

```

Inspector Console Tasks

Use print(...) to write to this console.



NDVI

<https://code.earthengine.google.com/ec9e2f81b63ad006475b263f5f0cd661>

```
var point = ee.Geometry.Point(37.02540746271644,56.766160331658845);
var i8 = ee.ImageCollection('LANDSAT/LC08/C02/T1_TOA');

// Get the least cloudy image in 2015.
var image = ee.Image(
  i8.filterBounds(point)
    .filterDate('2022-06-01', '2022-07-17')
    .sort('CLOUD_COVER')
    .first()
);

// Compute the Normalized Difference Vegetation Index (NDVI).
var nir = image.select('B5');
var red = image.select('B4');
var ndvi = nir.subtract(red).divide(nir.add(red)).rename('NDVI');

// Display the result.
Map.setCenter(37.02540746271644,56.766160331658845, 12);
var ndviParams = {min: -1, max: 1, palette: ['blue', 'white', 'green']};
Map.addLayer(ndvi, ndviParams, 'NDVI image');
```

- Scripts
- Docs
- Assets
 - changes
 - classification
 - collection clouds
 - collection mask
 - collection median
 - collection ndvi
 - collections
 - fires
 - image
 - image comp
 - image statistics

```

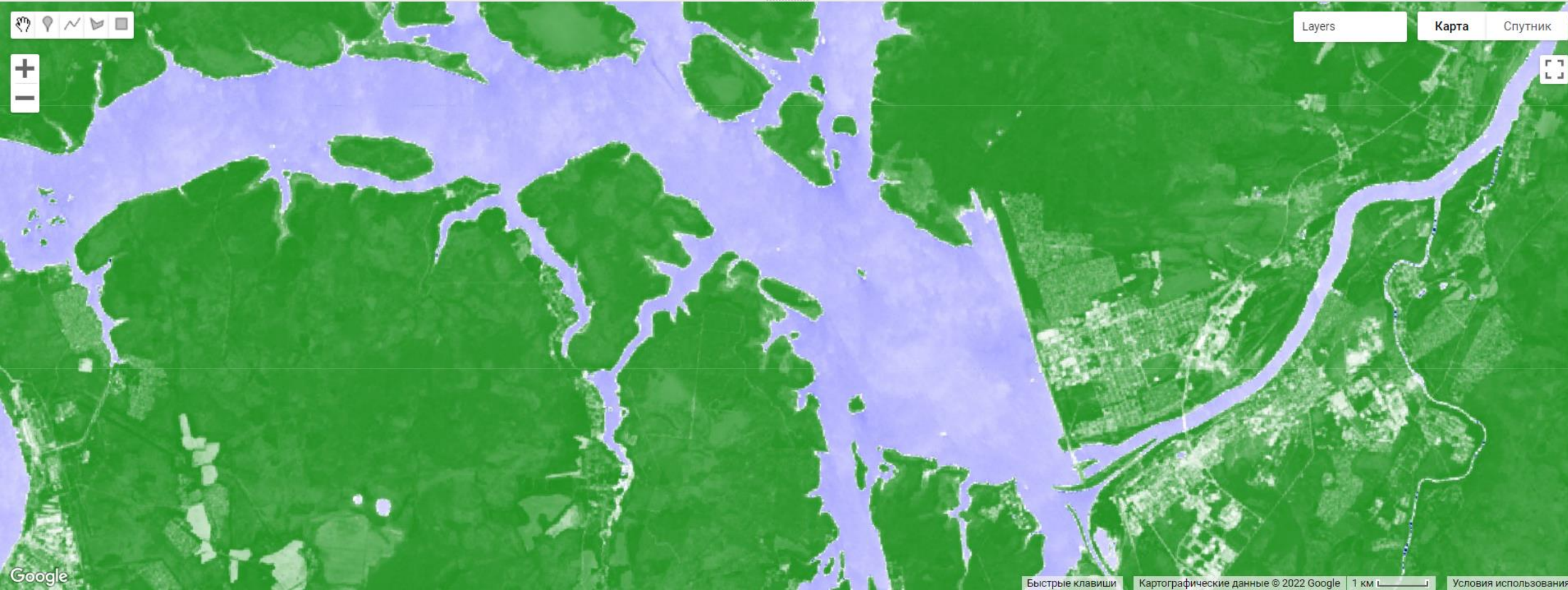
collection ndvi
Get Link Save Run Reset Apps

4 // Get the least cloudy image in 2015.
5 var image = ee.Image(
6   18.filterBounds(point)
7     .filterDate('2022-06-01', '2022-07-17')
8     .sort('CLOUD_COVER')
9     .first()
10 );
11
12 // Compute the Normalized Difference Vegetation Index (NDVI).
13 var nir = image.select('B5');
14 var red = image.select('B4');
15 var ndvi = nir.subtract(red).divide(nir.add(red)).rename('NDVI');
16

```

Inspector Console Tasks

Use print(...) to write to this console.



Charts

<https://code.earthengine.google.com/ec976be7da2129036839e1cf2f21cbbb>

```
//var cloud_perc = 60;//Max cloud percentile per scene.

// Import the Landsat 8 TOA image collection.
var l8 = ee.ImageCollection('LANDSAT/LC08/C01/T1_TOA')
  .filterDate('2021-03-01', '2021-12-01');
//   .filter(ee.Filter.lt('CLOUD_COVER', cloud_perc));

// Map a function over the Landsat 8 TOA collection to add an NDVI band.
var withNDVI = l8.map(function(image) {
  var ndvi = image.normalizedDifference(['B5', 'B4']).rename('NDVI');
  return image.addBands(ndvi);
});

var roi = ee.Geometry.Rectangle(37.01900746271644,56.76360331658845,37.0340746271644,56.76860331658845);
Map.setCenter(37.02540746271644,56.766160331658845, 12);
Map.addLayer(roi, {color: 'blue'}, 'clouds');

// Create a chart.
var chart = ui.Chart.image.series({
  imageCollection: withNDVI.select('NDVI'),
  region: roi,
  reducer: ee.Reducer.median(),
  scale: 30
}).setOptions({title: 'NDVI over time'});

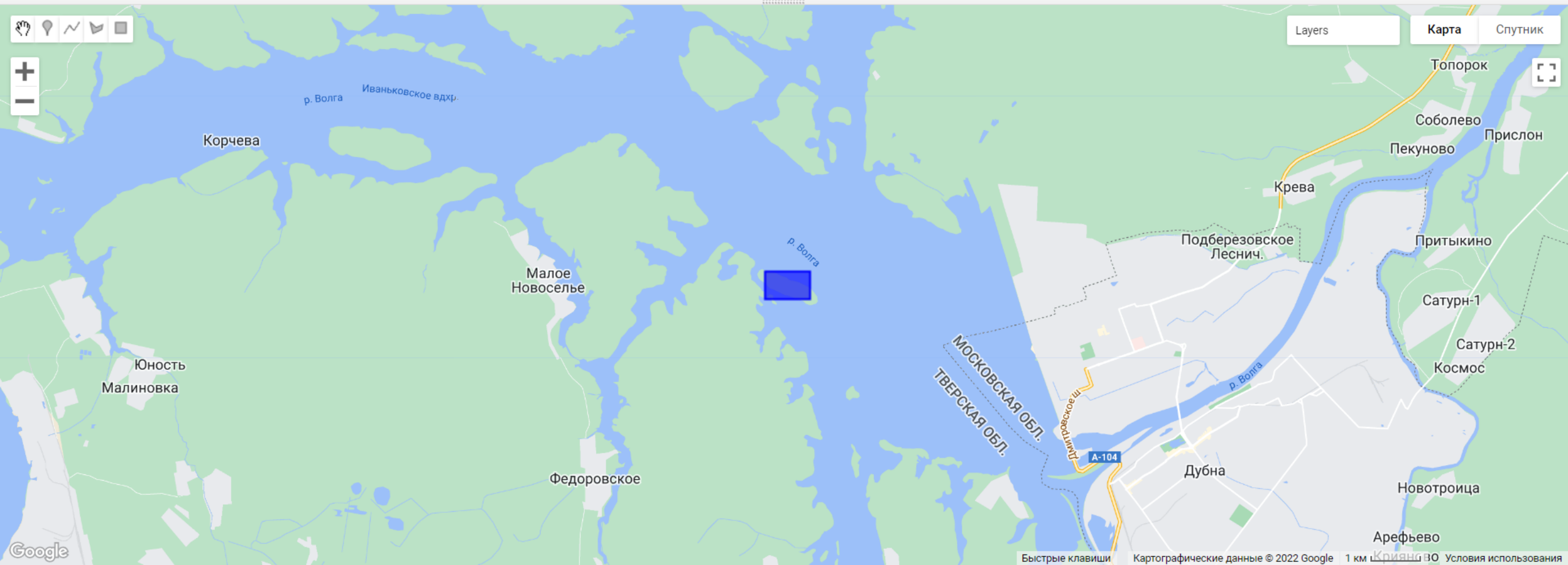
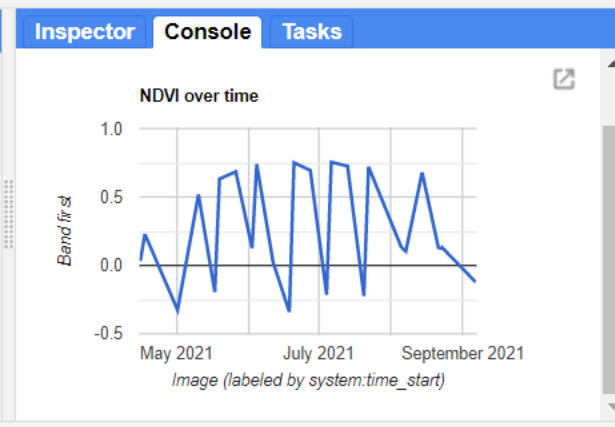
// Display the chart in the console.
print(chart);
```

- Scripts
- Docs
- Assets
- Layer Filters
- Linked Maps
- Manual Legend
- Mobile Friendly UI
- Mosaic Editor
- Ocean Timeseries Investigator
- Population Explorer
- Split Panel
- Two Chart Inspector
- Zoom Box
- Datasets
- Demos

```

lip/Charting
1 //var cloud_perc = 60;//Max cloud percentile per scene.
2
3 // Import the Landsat 8 TOA image collection.
4 var l8 = ee.ImageCollection('LANDSAT/LC08/C01/T1_TOA')
5   .filterDate('2021-03-01', '2021-10-01');
6   //   .filter(ee.Filter.lt('CLOUD_COVER', cloud_perc));
7
8 // Map a function over the Landsat 8 TOA collection to add an NDVI band.
9 var withNDVI = l8.map(function(image) {
10   var ndvi = image.normalizedDifference(['B5', 'B4']).rename('NDVI');
11   return image.addBands(ndvi);
12 });
13
14 var roi = ee.Geometry.Rectangle(37.01900746271644, 56.76360331658845, 37.0340746271644, 56.76860331658845);
15

```



Burn Areas

<https://developers.google.com/earth-engine/datasets/catalog/FIRMS?hl=en>

<https://code.earthengine.google.com/e9c2c3dd7cfb47bb55111557da6102d0>

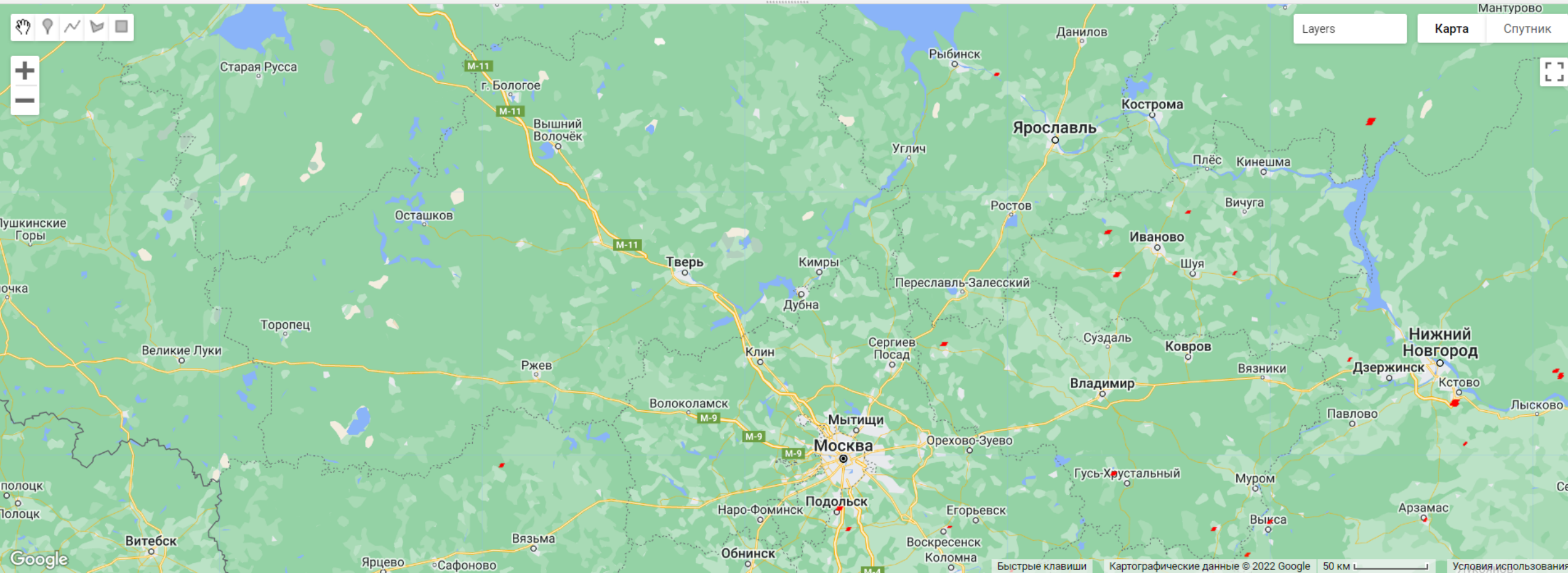
```
var dataset = ee.ImageCollection('FIRMS').filter(
  ee.Filter.date('2022-07-01', '2022-07-16'));
var fires = dataset.select('T21');
var firesVis = {
  min: 325.0,
  max: 400.0,
  palette: [
    'ff0000', 'fd4100', 'fb8200', 'f9c400', 'f2ff00', 'b6ff05',
    '7aff0a', '3eff0f', '02ff15', '00ff55', '00ff99', '00ffdd',
    '00ddff', '0098ff', '0052ff', '0210ff', '3a0dfb', '7209f6',
    'a905f1', 'e102ed', 'ff00cc', 'ff0089', 'ff0047', 'ff0004'
  ]
};
Map.setCenter(37.02540746271644, 56.766160331658845, 7);
Map.addLayer(fires, firesVis, 'Fires');
```

- Scripts
- Docs
- Assets
 - collection mask
 - collection median
 - collection ndvi
 - collections
 - fires
 - image
 - image comp
 - image statistics
 - Ivanovo
 - Poland
 - Sweden
 - TOA corrections

```
lip/fires *
3 var fires = dataset.select('T21');
4 var firesVis = {
5   min: 325.0,
6   max: 400.0,
7   palette: [
8     'ff0000', 'fd4100', 'fb8200', 'f9c400', 'f2ff00', 'b6ff05',
9     '7aff0a', '3eff0f', '02ff15', '00ff55', '00ff99', '00ffdd',
10    '00ddff', '0098ff', '0052ff', '0210ff', '3a0dfb', '7209f6',
11    'a905f1', 'e102ed', 'ff00cc', 'ff0089', 'ff0047', 'ff0004'
12  ]
13 };
14 Map.setCenter(37.02540746271644, 56.766160331658845, 7);
15 Map.addLayer(fires, firesVis, 'Fires');
16
```

Inspector Console Tasks

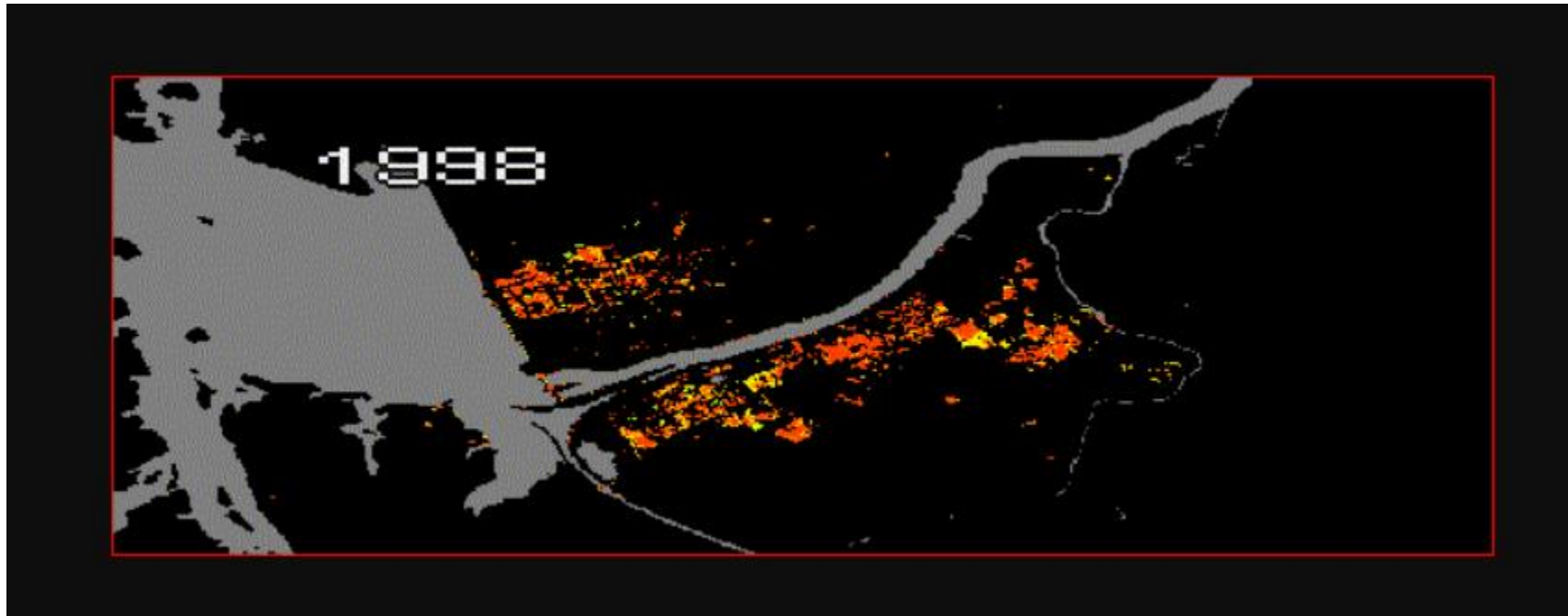
Use print(...) to write to this console.



Timeseries

https://developers.google.com/earth-engine/datasets/catalog/Tsinghua_FROM-GLC_GAIA_v10?hl=en#description

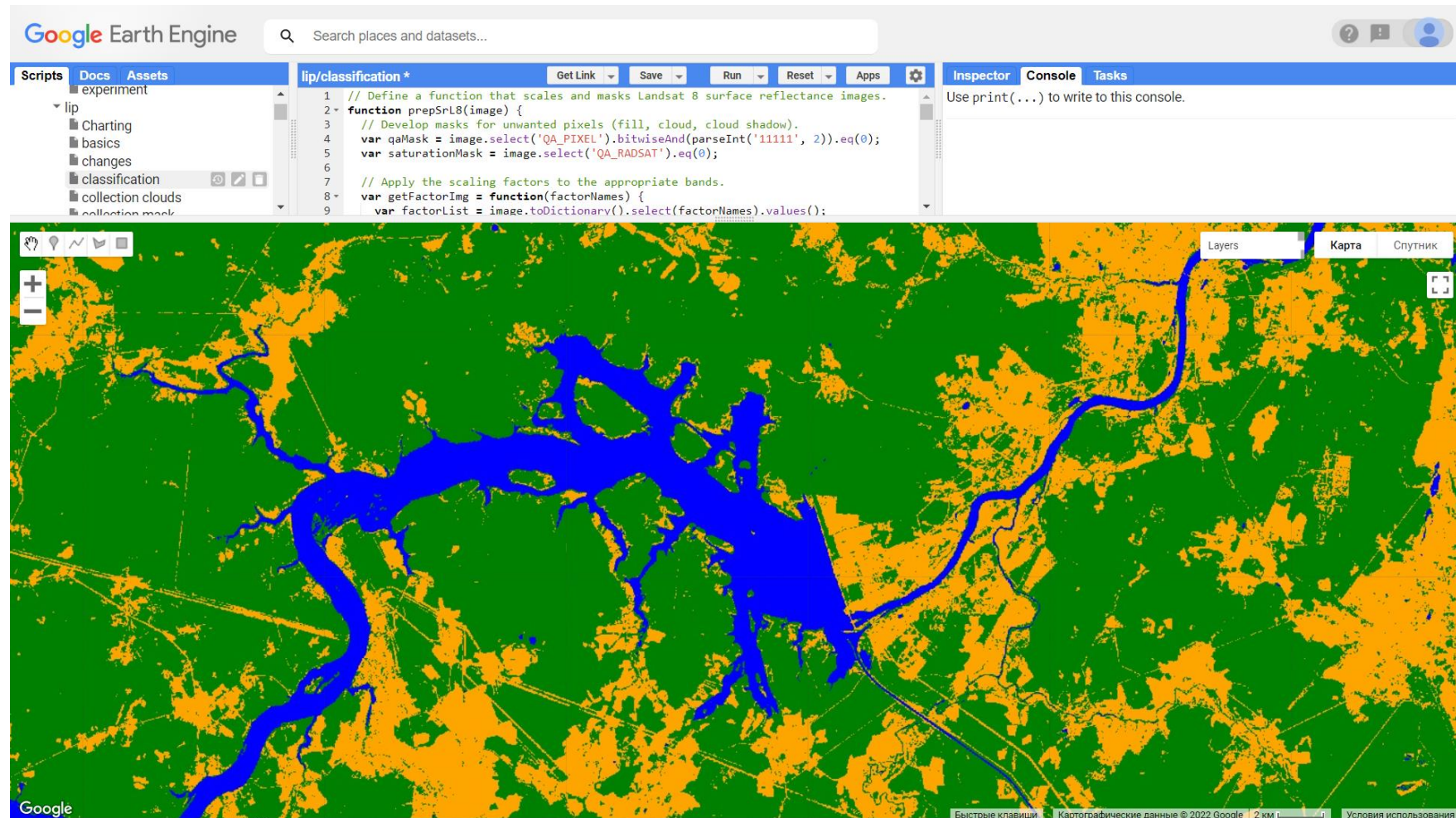
<https://code.earthengine.google.com/cdd574d1d7466208b1eb5bff5fb7c02e>



Land use classification

<https://developers.google.com/earth-engine/apidocs/ee-classifier-libsvm>

<https://code.earthengine.google.com/1828d8cf8bd900884f4a9746ee214252>



The screenshot displays the Google Earth Engine web interface. At the top, the Google Earth Engine logo and a search bar are visible. Below the search bar, there are tabs for 'Scripts', 'Docs', and 'Assets'. The 'Scripts' tab is active, showing a file tree on the left with folders like 'lip', 'lip/classification', and 'lip/classification *'. The main editor area shows a JavaScript script for land use classification. The script defines a function 'prepSrL8' that takes an image as input and returns a classified image. It uses 'QA_PIXEL' and 'QA_RADSAT' masks to filter out unwanted pixels and then applies scaling factors to the appropriate bands. The script is as follows:

```
1 // Define a function that scales and masks Landsat 8 surface reflectance images.
2 function prepSrL8(image) {
3   // Develop masks for unwanted pixels (fill, cloud, cloud shadow).
4   var qaMask = image.select('QA_PIXEL').bitwiseAnd(parseInt('11111', 2)).eq(0);
5   var saturationMask = image.select('QA_RADSAT').eq(0);
6
7   // Apply the scaling factors to the appropriate bands.
8   var getFactorImg = function(factorNames) {
9     var factorList = image.toDictionary().select(factorNames).values();
```

The map view below the script shows a satellite image of a landscape with a river network. The river network is highlighted in blue, indicating the result of the classification. The map includes standard navigation controls like zoom in (+) and zoom out (-) buttons, and a 'Layers' panel on the right. The bottom of the interface shows the Google logo, keyboard shortcuts, and copyright information for 2022.

