[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

PROCESSING **FEATURE COLLECTION** [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

A feature collection is is an EE variable object that represents a set of features. Feature collections can be processed by using operations of the types listed below,

which vary according to the nature of that processing. Each operation name is linked to a separate page describing that operation.

**UPLOADING** FEATURE COLLECTIONS [Accessing Public Sources of Feature Data](#PUBLIC)

[Creating Fusion Tables from Private Shapefiles with *Shpescape*](#SHPSCAPE)

[Creating Fusion Tables from Private Shapefiles with ArcGIS, Google Earth, and Google Drive](#ARCGIS)

[Creating Fusion Tables from Private Files of Delimited Text](#PRIVATE)

[Creating Feature Collections from Fusion Tables](#FUSION)

**ACCESSING** FEATURE COLLECTIONS [ee.FeatureCollection(fusionTableID)](#FeatureCollection) [ee.Collection.loadTable](#loadTable)

**CREATING** FEATURE COLLECTIONS [ee.FeatureCollection(geometry)](#FeatureCollection_geometry) [ee.FeatureCollection.randomPoints](#randomPoints)

[Onscreen Drawing](#Drawing)

**EDITING** FEATURE COLLECTIONS

BY **LIMITING** FEATURES [featureCollection.limit](#limit)

BY **FILTERING** FEATURES [featureCollection.filterMetadata](#filterMetadata) [featureCollection.filterDate](#filterDate)

[featureCollection.filterBounds](#filterBounds) [featureCollection.filter](#filter)

BY **SELECTING** FEATURES [featureCollection.select](#select) [featureCollection.distinct](#distinct)

BY **COMBINING** FEATURES [featureCollection.union](#union) [featureCollection.merge](#merge)

BY **JOINING** COLLECTIONS [ee.Join.apply](#apply)

BY **RESETTING** VALUES [featureCollection.set](#set) [featureCollection.setMulti](#setMulti)

[featureCollection.remap](#remap)

BY **REORGANIZING** THEM [featureCollection.sort](#sort) [featureCollection.flatten](#flatten)

[featureCollection.makeArray](#makeArray)

**TRANSFORMING** FEATURE COLLECTIONS [featureCollection.classify](#classify)

**REPRODUCING** FEATURE COLLECTIONS

AS **GEOMETRIES** [featureCollection.geometry](#geometry)

AS **IMAGES** [featureCollection.reduceToImage](#reduceToImage) [image.paint(featureCollection)](#paint)

AS GOOGLEMAP **OVERLAYS** [featureCollection.getMap](#getMap)

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

PROCESSING **FEATURE COLLECTION** [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

**QUERYING** FEATURE COLLECTIONS

FOR **FEATURES** [featureCollection.first](#first) [featureCollection.toList](#toList)

FOR **VALUES** [featureCollection.aggregate\_first](#ag_first) [featureCollection.aggregate\_array](#ag_array)

**SUMMARIZING** FEATURE COLLECTIONS

BY **PROPERTY VALUE** [featureCollection.aggregate\_stats](#ag_stats) [featureCollection.aggregate\_histogram](#ag_histo)

[featureCollection.aggregate\_count](#ag_count_) [featureCollection.aggregate\_count\_distinct](#ag_count_)

[featureCollection.aggregate\_max](#ag_count_)  [featureCollection.aggregate\_min](#ag_count_) [featureCollection.aggregate\_sum](#ag_count_) [featureCollection.aggregate\_mean](#ag_count_) [featureCollection.aggregate\_product](#ag_count_)

[featureCollection.aggregate\_sample\_var](#ag_count_) [featureCollection.aggregate\_total\_var](#ag_count_) [featureCollection.aggregate\_sample\_sd](#ag_count_) [featureCollection.aggregate\_total\_sd](ag_count_#ag_count_)

[featureCollection.reduceColumns](#reduceColumns)

**COMPARING** FEATURE COLLECTIONS [ee.Algorithms.IsEqual(featureCollection)](#IsEqual)

**PARALLEL PROCESSING** FEATURE COLLECTIONS [featureCollection.map](#map)

**DESCRIBING** FEATURE COLLECTIONS [featureCollection.errorMatrix](#errorMatrix)

**DOCUMENTING** FEATURE COLLECTIONS [featureCollection.getInfo](#Describe_getInfo) [ee.Algorithms.Describe(featureCollection)](#Describe_getInfo)

[featureCollection.toString](#toString_serialize) [featureCollection.serialize](#toString_serialize)

**PRESENTING** FEATURE COLLECTIONS

IN **PRINT** [print(featureCollection)](#print_console) [console.log(featureCollection)](#print_console)

[alert(featureCollection)](#alert_confirm) [confirm(featureCollection)](#alert_confirm)

IN **MAPS** [Map.addLayer(featureCollection)](#addLayer) [featureCollection.draw](#draw)

IN **CHARTS** [Chart.feature.histogram](#Chart_histo) [Chart.feature.byFeature](#Chart_byFeature)

[Chart.feature.byProperty](#Chart_byProperty) [Chart.feature.groups](#Chart_groups)

**EXPORTING** FEATURE COLLECTIONS [featureCollection.getDownloadURL](#getDownloadUrl) [ee.data.makeTableDownloadUrl](#TableDownload)

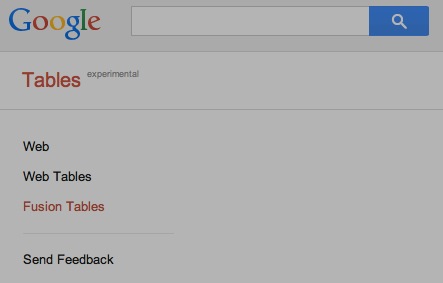
[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**UPLOADING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) ACCESSING PUBLIC SOURCES OF FEATURE DATA

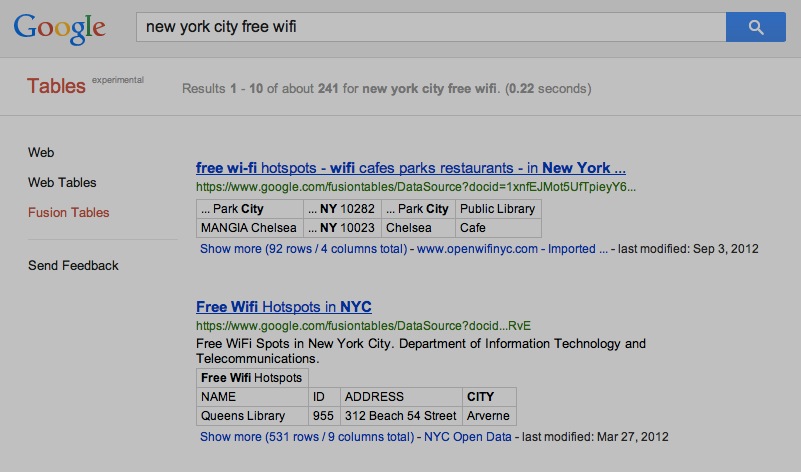
Both public and private data on (point, line, or polygon) features can be incorporated into Earth Engine by using another Google application called *Fusion Tables*,

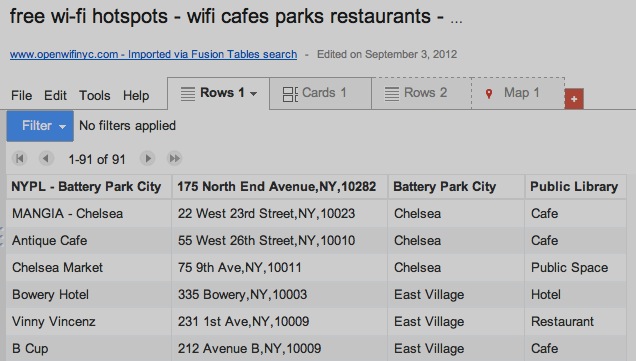
one that provides for the creation, storage, sharing, and visualization of tabular data online. Here, each set of data stored is (also) referred to as a “fusion table.”

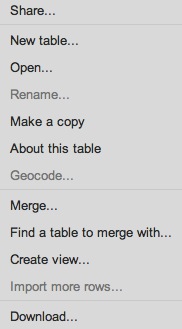
To open a feature collection that has been been stored and shared with the public as a fusion table requires the assetID under which that fusion table has been stored. To browse for such fusion table and to access their assetIDs, start by using Google’s *Chrome* web browser to navigate to <https://research.google.com/tables>. Then select **Fusion Tables** and use this field to specify an identifying characteristic of the data you’d like to access. This will yield a listing of available fusion tables, many

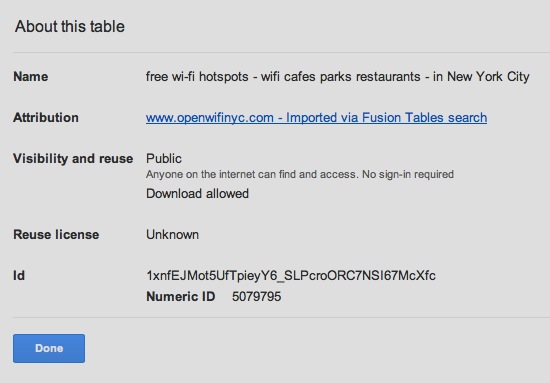
of which will have the locational fields (like street addresses or lon/lat coordinates) necessary to

generate cartographic displays. In fact, it will often be possible to automatically generate those displays within Fusion Table itself by simply clicking here in order to generate output like this.







****

For more information

on *Fusion Tables*,

check out

<https://www.google.com/fusiontables> and

<https://support.google.com/fusiontables>.

To get a fusion table’s assetID, use **File > About this table**

to open a listing a panel from which you can copy this

(by using **Edit > Copy**) and paste it into an Earth Engine script like this.

var hotSpotsInNYC = ee.FeatureCollection( 'ft:1xnfEJMot5UfTpieyY6\_SLPcroORC7NSI67McXfc' );

Map.addLayer( hotSpotsInNYC, { color:'FF0000' }, 'Free WiFi in New York City' );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**UPLOADING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) CREATING FUSION TABLES FROM PRIVATE SHAPEFILES WITH SHPSCAPE

To access your own point, line, or polygon data as an Earth Engine feature collection.

those data must first be stored online as a fusion table. To do so, start by navigating to

<https://support.google.com/fusiontables/answer/2571232?hl=en> with Google *Chrome*

and an authorized Earth Engine account. Here, you can click on **Get the Drive App**

and follow the instructions necessary to do just that.

Once installed, *Fusion Tables* (the application) will enable you to store *fusion tables*

(the data type) of up to several hundered thousand records at no cost.

Point, line, and polygon data of several types can be reproduced as fusion tables and

thereby used to generate feature collections accessible to Earth Engine. The include

- ArcGIS files (in shapefile format);

- Google Earth files (in .KML or .KMZ format); o

- delimited text files (in .TXT, .CSV, or TSV format) containing longitude and latitude

coordinates, street address, place names, or other geo-locational identifiers that

are sufficiently common and conspicuos to be (automatically) detected as such; or

- Google Spreadsheets.

To upload a (zipped) shapefile to *Fusion Tables* is made very easy by navigating to <http://shpescape.com/>, clicking on **shp 2 fusion tables**, following several simple instructions, and being patient if the shapefile is large. The result will be an asset ID like this.



If you click on that asset ID as it presented by <http://shpescape.com/>,

a link to this web address will be invoked, and the fusion table will

be opened in Fusion Tables.

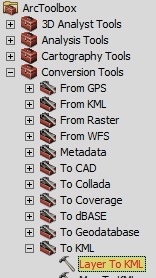
The procedure required to generate a feature collection

from a fusion table is described on page 5.

http://www.google.com/fusiontables/DataSource?docid=1ICSA4CbbuhwLOmwuAgOJD6oUCgQy0veryqFuD1QH&pli=1#map:id=3

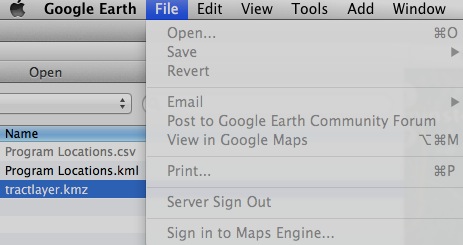
[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**UPLOADING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) CREATING FUSION TABLES FROM PRIVATE FILES WITH ARCGIS, GOOGLE EARTH, AND GOOGLE DRIVE

An alternative route from shapefile to fusion table starts with **ArcGIS > ArcToolbox > Conversion Tools > To KML > Layer to KML**.

This will generate to create a .KMZ file that can then be opened in Google Earth by using **Google Earth > File > Open**.

A .KMZ file is the zipped version of a .KML file.

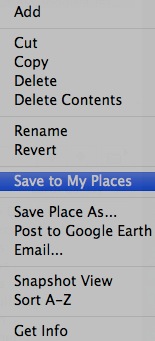


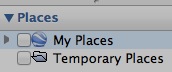
To generate the .KML file associated with the .KML file just opened in Google Earth,

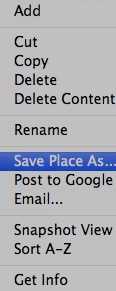
**Open …**

you must first right–click on the name of its folder and invoke **Save to My Places**.

in order to move that folder from ***Temporary Places***

to ***My Places*** in the Google Earth sidebar





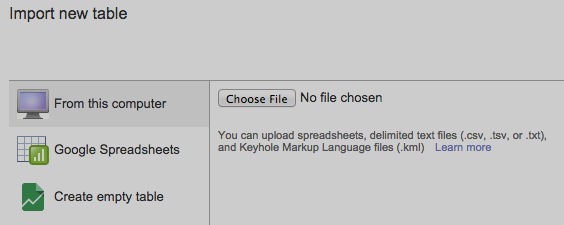
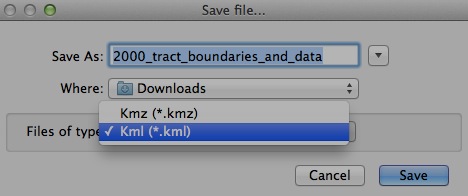
Once it is listed under ***My Places***, right-click

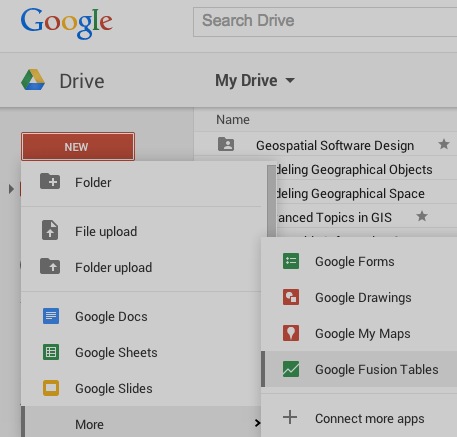
on the .KMZ folder, invoke the **Save Place As**

operation, and use this **Save file** dialog box

to create a copy in

.KML format.

To generate a fusion table from this .KML file, navigate with *Chrome* to <http://drive.google.com> and click on **New** then

(**More** then) **Google Fusion Tables**. This will open a dialog box in which you can

specify the .KML

file to be uploaded

and click on **Next**

to do so.

The procedure required to generate a feature collection from a fusion table is described on page 5.

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**UPLOADING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) CREATING FUSION TABLES FROM PRIVATE FILES OF DELIMITED TEXT

A procedure similar to that which generates a fusion table from a private .KML file can also be used

to generate a fusion table from a private text file of comma-separated or tab separated coordinates,

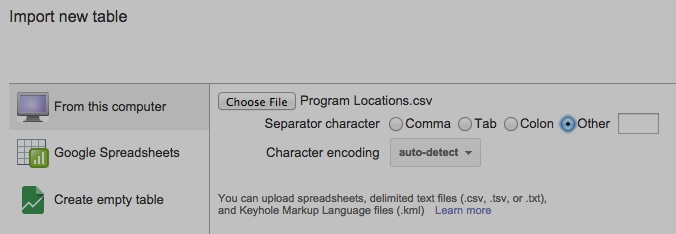
strret addresses, or place names. Navigate (with *Chrome*) to <http://drive.google.com>, then click on

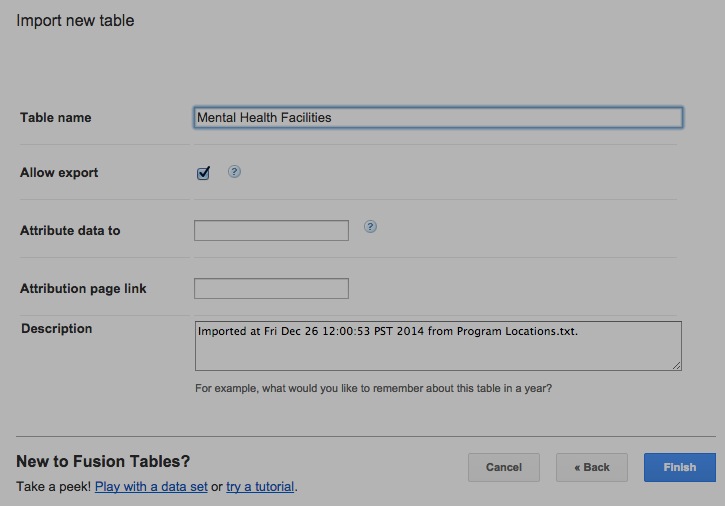
**New**, **More**, and **Google Fusion Tables** in order to open an **Import new table** dialog box, and use

this **Choose File** button to select a file to upload. Finally, indicate that file’s

**Separator character** and **Character Encoding** format before

clicking **Next** (twice) …





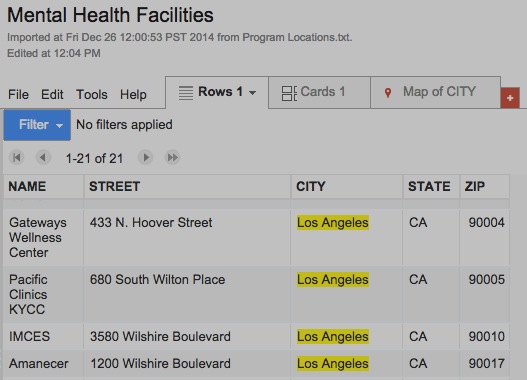


.

… then (optionally) providing additional information about the newly created

fusion table before clicking on a **Finish**

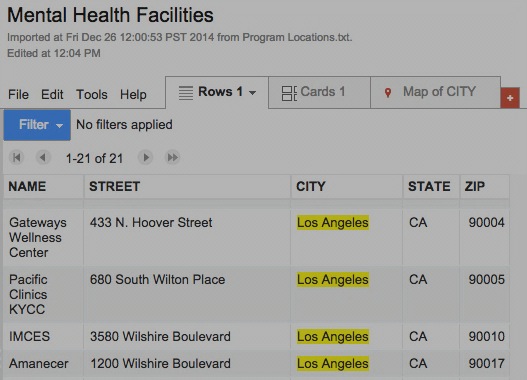
button and examining the result.

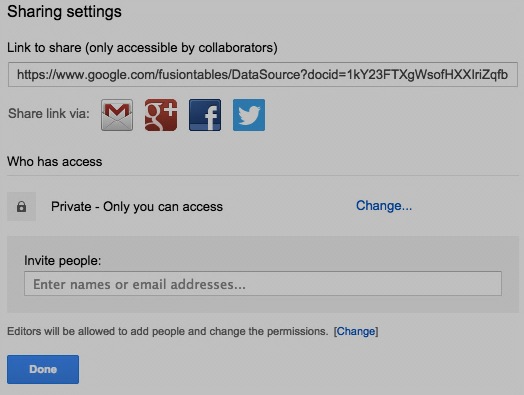
****

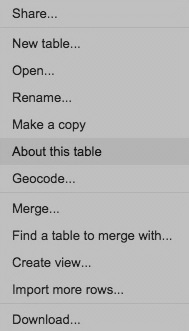
[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

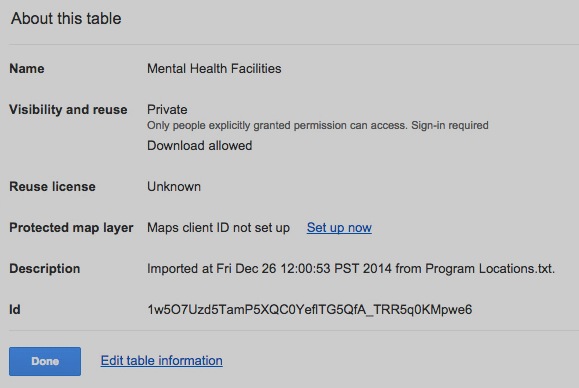
**UPLOADING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) CREATING FEATURE COLLECTIONS FROM FUSION TABLES

Once a fusion table has been generated, it can that can be processed by *Fusion Tables* itself in the variety of interesting ways. To use a fusion table to generate an Earth Engine feature collection, however, is simply a matter of clicking on this **Share** button, then here, here, and here in order to make to make it available, then …

********



****

****

… clicking here to complete this procedure. While this

link can then be used to access the fusion table in

*Fusion Tables*, what is needed in order to access it

as a feature collection in Earth Engine is its asset ID.

A click on **File > About this table** will open a panel like this

from which that asset ID can be copied (by right-clicking on it and using **Copy** rather than CTRL-V)

and pasted into an Earth Engine script as illustrated here.

var Facilities = ee.FeatureCollection( 'ft:1ICSA4CbbuhwLOmwuAgOJD6oUCgQy0veryqFuD1QH' );

Map.addLayer( Facilities, { color: 'ff0000' }, 'Health Facilities' );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**ACCESSING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

ee.FeatureCollection creates a new feature collection from a specified fusion fable.

newFeatureCollection = ee.FeatureCollection ( fusionTableID*,  geometryFieldName* )

The asset ID of the specified fusion table, given as a string beginningwith “ft:”

The name of a fusion table column to be used to define feature geometries. Default: ‘geometry’

The new feature collection

var polygonFEATURES = ee.FeatureCollection( 'ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6' ); // CT Census Tracts

var pointFEATURES = ee.FeatureCollection( 'ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6', 'geometry\_pos' );

Map.centerObject( polygonFEATURES, 11 );

Map.addLayer( polygonFEATURES, { color: '220099' }, 'The Geometry Column' );

Map.addLayer( pointFEATURES, { color: 'ffff00' }, 'The Geometry\_pos Column' );

Map.addLayer( pointFEATURES, { color: 'ff00ff' }, 'The Geometry\_pos Column' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**ACCESSING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

ee.Collection.loadTable creates a new feature collection from a specified fusion fable.

newFeatureCollection = ee.Collection.loadTable ( fusionTableID*,  geometryFieldName, version* )

The asset ID of the specified fusion table, given as a string beginning with “ft:”

The fusion table version number, given as a number. Default: -1 (the current version)

The new feature collection

The name of a fusion table column to be used to define feature geometries. Default: ‘geometry’

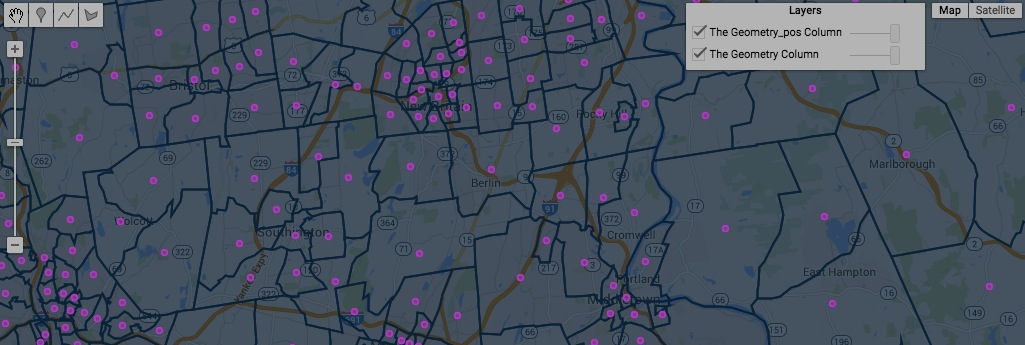
var polygonFEATURES = ee.Collection.loadTable( 'ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6' ); // CT Census Tracts

var pointFEATURES = ee.Collection.loadTable( 'ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6', 'geometry\_pos' );

Map.centerObject( polygonFEATURES, 11 );

Map.addLayer( polygonFEATURES, { color: '113355' }, 'The Geometry Column' );

Map.addLayer( pointFEATURES, { color: 'ff00ff' }, 'The Geometry\_pos Column' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**CREATING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **ONSCREEN DRAWING**

DESCRIBE THE ONSCREEN DRAWING TOOLS, FOCUSING ONLY ON FEATURE COLLECTIONS

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**CREATING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

ee.FeatureCollection creates a new feature collection from a specified geometry or list of geometries.

newFeatureCollection = ee.FeatureCollection ( oldGeometry )

The new feature

The specified geometry or list of geometries

var ColoGEOMETRY = ee.Geometry.Polygon([[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

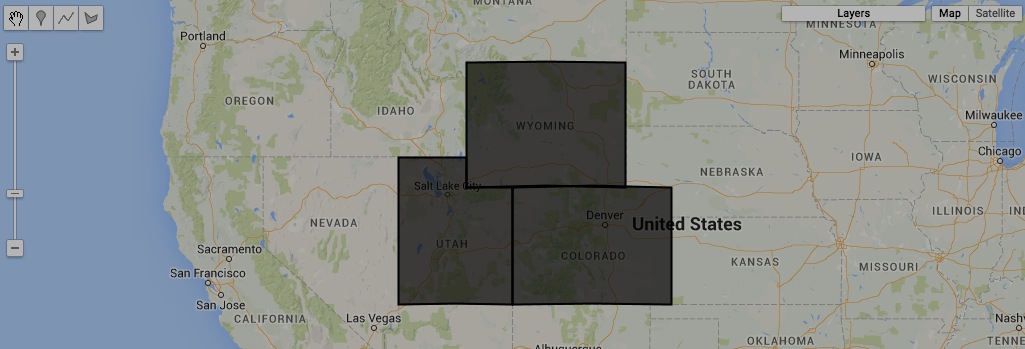
var WyomGEOMETRY = ee.Geometry.Polygon([[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var TheFEATURES = ee.FeatureCollection( [ColoGEOMETRY, WyomGEOMETRY, UtahGEOMETRY] );

Map.setCenter( -109.05, 41, 5 );

Map.addLayer( TheFEATURES );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**CREATING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

ee.FeatureCollection.randomPoints creates new feature collection containing a specified number of point features

that are uniformly but randomly distributed throughout a specified region.

newFeatureCollection = ee.FeatureCollection.randomPoints ( region*, points, seed, errorMargin* )

The specified feature collection

The specified region, given as a geometry

A seed for the random

point generator, given

as an integer.

Default: 1

An error margin

indicating the spatial

precision to be

maintained

The new

feature

collection

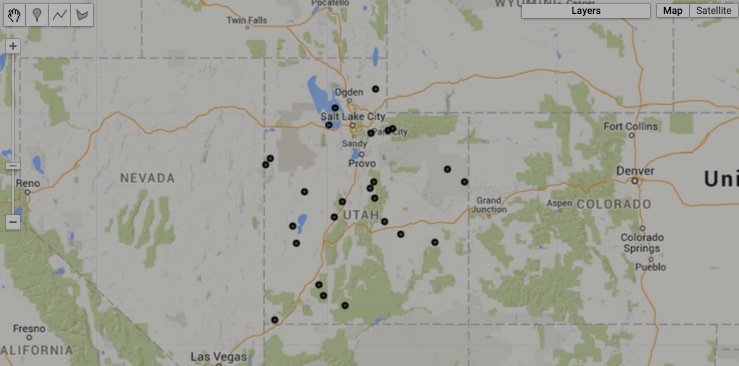
A specified number of points, given as an integer. Default: 1000

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var NewFEATURES = ee.FeatureCollection.randomPoints( UtahGEOMETRY, 25, 12 );

Map.setCenter( -111.445, 39.251, 6 );

Map.addLayer( NewFEATURES );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **LIMITING** FEATURES

featureCollection.limit creates new feature collection that contains only the first of a specified number of elements from a specified feature collection

after (optionally) sorting those elements according to the ascending or descending order of their values for a specified property.

newFeatureCollection = oldFeatureCollection.limit ( howMany*, sortProperty, ascendingOrder?* )

The specified number of elements,

given as an integer

The specified property,

given as a string

The specified feature

collection

The specified order, given as a Boolean set to

True for ascending or False for descending

The new feature

collection

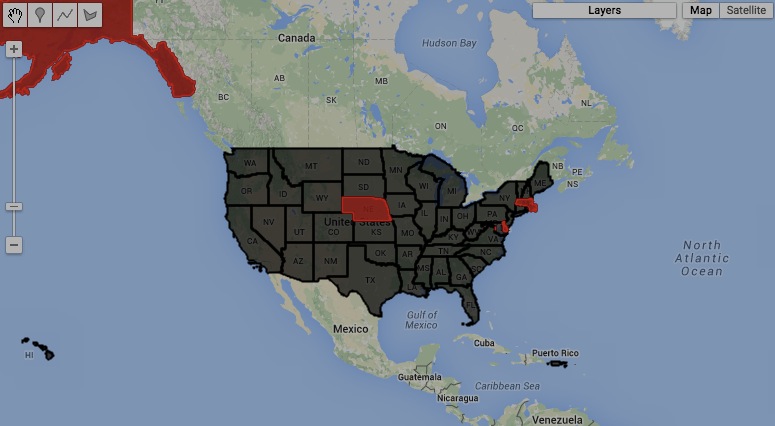
var AllFEATURES = ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNaR7Xk8'); // U.S. States

var SomeFEATURES = AllFEATURES.limit(5);

Map.setCenter( -95.9766, 39.3683, 3 );

Map.addLayer ( AllFEATURES );

Map.addLayer ( SomeFEATURES, {color:'FF0000'} );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **FILTERING** FEATURES

featureCollection.filterMetadata creates a new feature collection that contains only those features from a specified feature collection

whose values for a specified property bear a specified relationship to a specified value.

newFeatureCollection = oldFeatureCollection.filterMetadata ( property, relationship, value )

The specified value,

given as a string or a number

The specified feature collection

The new feature collection

The name of the specified property, given as a string

The specified relationship, given as **"equals" "less\_than" "greater\_than" "starts\_with" "ends\_with" "contains"**

one of the following strings. **"not\_equals" "not\_less\_than" "not\_greater\_than" "not\_starts\_with" "not\_ends\_with" "not\_contains"**

var AllFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ); // U.S. Cities

var SomeFEATURES = AllFEATURES.filterMetadata('city\_name', 'contains', 'ton' );

Map.setCenter( -95.9766, 39.3683, 3 );

Map.addLayer ( AllFEATURES );

print(SomeFEATURES.getInfo());

Map.addLayer ( SomeFEATURES, {color:'FF0000'} );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **FILTERING** FEATURES

featureCollection.filterDate creates a new feature collection that contains only those features from a specified feature collection

that are associated with the time period between specified starting and stopping dates.

newFeatureCollection = oldFeatureCollection.filterDate ( startingDate, stoppingDate )

The specified starting and stopping dates, given either as a string in 'month#/day#/year#'

(or 'month#-day#-year#') format or the number of milliseconds since January 1, 1970

The specified feature collection

The new feature collection

var need to find a dated feature collection to develop this example

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **FILTERING** FEATURES

featureCollection.filterBounds creates a new feature collection that contains only those features from a specified feature collection

that intersect a specified geometry, feature, or feature collection.

newFeatureCollection = oldFeatureCollection.filterBounds( oldGeometry )

The specified feature collection

The specified geometry

The new feature collection

var AllFEATURES = ee.FeatureCollection ('ft:1S4EB6319wWW2sWQDPhDvmSBIVrD3iEmCLYB7nMM'); // U.S. Counties

var SelectionFEATURES = ee.FeatureCollection ('ft:1dXsoWNDPeUEXzlQV52jaI4zeckuyIlHiUwox3K4');

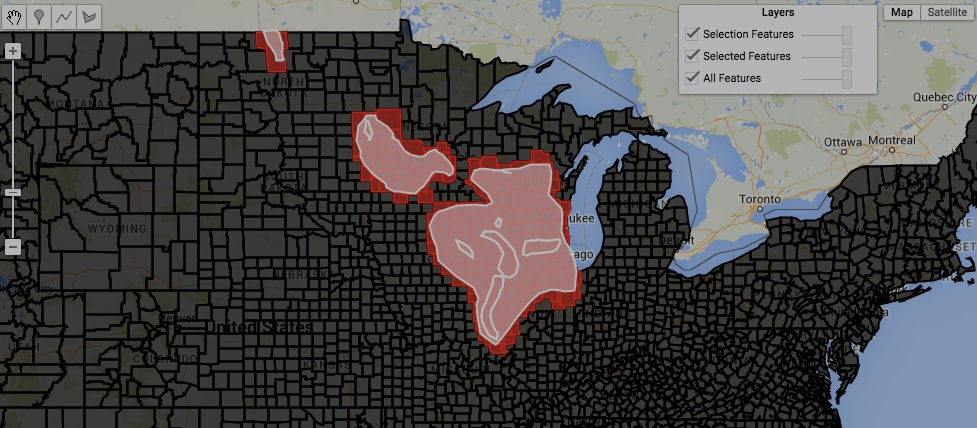
var SelectedFEATURES = AllFEATURES.filterBounds( SelectionFEATURES );

Map.centerObject ( SelectionFEATURES, 6 );

Map.addLayer ( AllFEATURES, null, 'All Features' );

Map.addLayer ( SelectedFEATURES, {color:'FF0000'}, 'Selected Features' );

Map.addLayer ( SelectionFEATURES, {color:'FFFFFF'}, 'Selection Features' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **FILTERING** FEATURES

featureCollection.filter creates a new feature collection that contains only those features from a specified feature collection

that satisfy the conditions of a specified filter.

newFeatureCollection = oldFeatureCollection.filter ( oldFilter )

The specified filter

The specified feature collection

The new feature collection

var AllFEATURES = ee.FeatureCollection ('ft:1S4EB6319wWW2sWQDPhDvmSBIVrD3iEmCLYB7nMM'); // U.S. Counties

var SelectionFEATURES = ee.FeatureCollection ('ft:1dXsoWNDPeUEXzlQV52jaI4zeckuyIlHiUwox3K4');

var TheFILTER = ee.Filter.bounds( SelectionFEATURES );

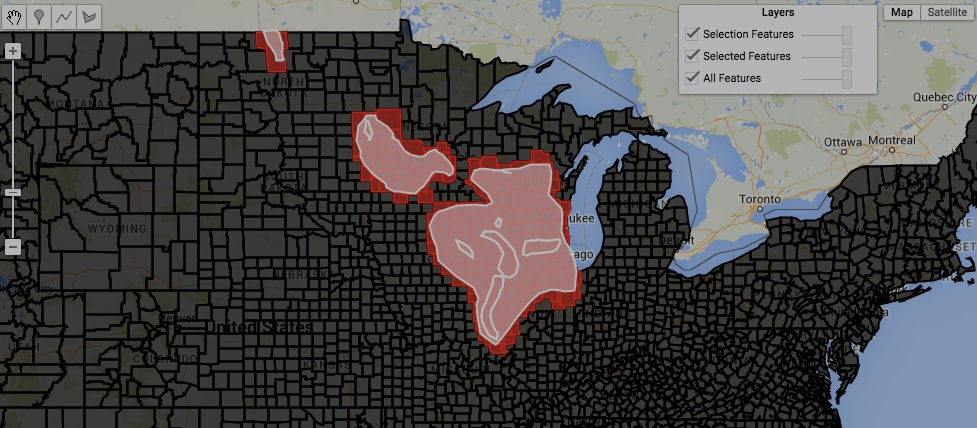
var SelectedFEATURES = AllFEATURES.filter( TheFILTER );

Map.centerObject ( SelectedFEATURES, 6 );

Map.addLayer ( AllFEATURES, 'All Features' );

Map.addLayer ( SelectedFEATURES, {color:'FF0000'}, 'Selected Features' );

Map.addLayer ( SelectionFEATURES, {color:'FFFFFF'}, 'Selection Features' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **SELECTING** FEATURES

featureCollection.select creates a new feature collection by replicating a specified feature collection but only with a specified set of properties.

newFeatureCollection = oldFeatureCollection.select ( propertyList*, propertyNameList* )

New names for the selected properties,

given as a list corresponding to **propertyList**

The specified properties, given

as a list of property names

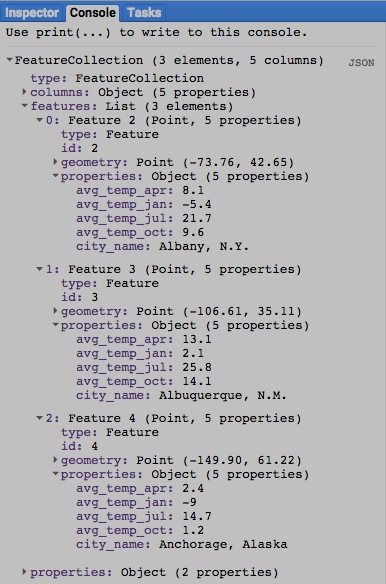
The specified feature collection

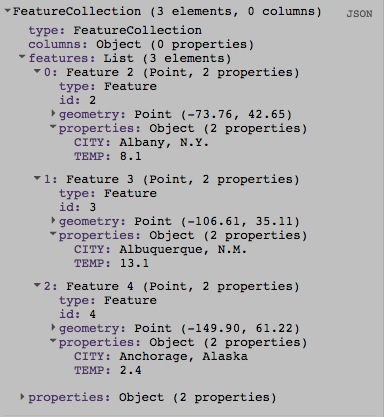
The new feature collection

var OldFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ).limit(3); // U.S. Cities

var NewFEATURES = OldFEATURES.select( ['city\_name','avg\_temp\_apr'], ['CITY','TEMP'] );

print( OldFEATURES, NewFEATURES );





[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **SELECTING** FEATURES

featureCollection.distinct creates a new feature collection by replicating a specified feature collection after removing any feature

whose value(s) for a specified set of one or more properties match those of a feature already considered.

newFeatureCollection = oldFeatureCollection.distinct ( properties )

The specified properties, given as a selectorSet

The specified feature collection

The new feature collection

var OldFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ); // U.S. Cities

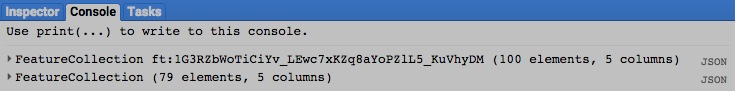
var TheSELECTORSET = ee.SelectorSet( ['avg\_temp\_apr'] );

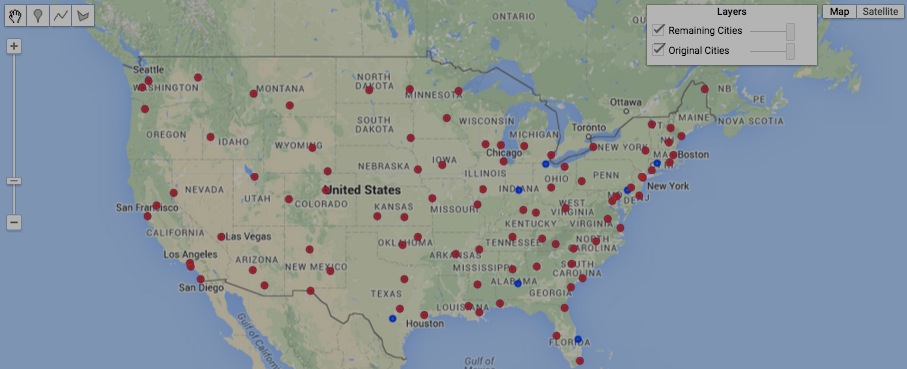
var NewFEATURES = OldFEATURES.distinct( TheSELECTORSET );

print( OldFEATURES.getInfo( ), NewFEATURES.getInfo( ) );

Map.addLayer( OldFEATURES, {color:'0000ff'}, 'Original Cities' );

Map.addLayer( NewFEATURES, {color:'ff0000'}, 'Remaining Cities' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **COMBINING** FEATURES

featureCollection.union creates a new feature collection by combining all of the features of a specified feature collection into a single feature

whose ID is ‘union\_result.’ This new feature retains none of the properties of the original features.

newFeatureCollection = oldFeatureCollection.union ( *errorMargin* )

The new feature collection

The specified feature collection

An error margin indicating the spatial precision to be maintained

var OldFEATURES = ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNaR7Xk8'); // U.S. States

var OldFEATURES = OldFEATURES.filterMetadata( 'Name', 'contains', 'New');

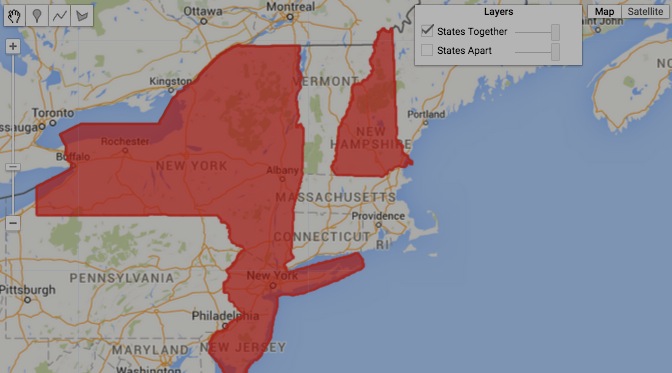
var NewFEATURES = OldFEATURES.union( );

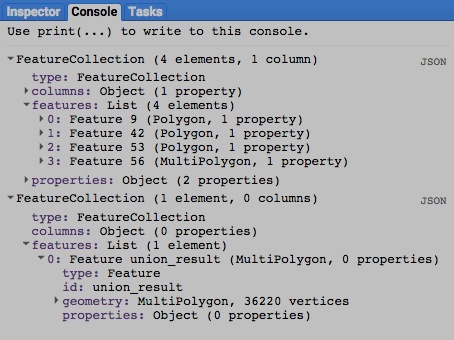
print( OldFEATURES, NewFEATURES );

Map.centerObject( NewFEATURES, 7 );

Map.addLayer( OldFEATURES, {color:'0000ff'}, 'States Apart' );

Map.addLayer( NewFEATURES, {color:'ff0000', opacity:0.5}, 'States Together' );





[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **COMBINING** FEATURES

featureCollection.merge creates new feature collection that contains all of the features from two specified feature collections.

newFeatureCollection = 1stOldFeatureCollection.merge ( 2ndOldFeatureCollection )

The second of the two specified feature collections

The first of the two specified feature collections

The new feature collection

var AllFEATURES = ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNaR7Xk8'); // U.S. States

var NorthFEATURES = AllFEATURES.filterMetadata('Name','contains','North');

var SouthFEATURES = AllFEATURES.filterMetadata('Name','contains','South');

var MergedFEATURES = NorthFEATURES.merge( SouthFEATURES );

print( NorthFEATURES, SouthFEATURES, MergedFEATURES );

Map.centerObject( MergedFEATURES, 4 );

Map.addLayer ( NorthFEATURES, {color:'ff0000'}, 'North States' );

Map.addLayer ( SouthFEATURES, {color:'0000ff'}, 'South States' );

Map.addLayer ( MergedFEATURES, {color:'00ff00'}, 'North and South States' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **JOINING** COLLECTIONS

ee.Join.apply creates a new (image or feature) collection by using a specified join to augment a specified “primary” (image or feature) collection

with whatever elements of a specified “secondary” (image or feature) collection satisfy a specified condition.

newFeatureCollection = oldJoin.apply( primaryCollection, secondaryCollection, condition )

The specified condition,

given as a filter

The specified

secondary collection

The specified

primary collection

The specified join

The new feature collection

var CityFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ); // U.S. Cities

var CityFEATURES = CityFEATURES.filterMetadata('city\_name','contains','Pa.'); // Pennsylvania Cities

var CountyFEATURES = ee.FeatureCollection('ft:1S4EB6319wWW2sWQDPhDvmSBIVrD3iEmCLYB7nMM'); // U.S. Counties

var CountyFEATURES = CountyFEATURES.filterMetadata('StateName','equals','Pennsylvania'); // Pennsylvania Counties

var TheFILTER = ee.Filter.withinDistance( 180000, '.geo', null, '.geo' );

var TheJOIN = ee.Join.simple();

var JoinedFEATURES = TheJOIN.apply( CountyFEATURES, CityFEATURES, TheFILTER );

print( 'Counties', CountyFEATURES );

print( 'Cities', CityFEATURES );

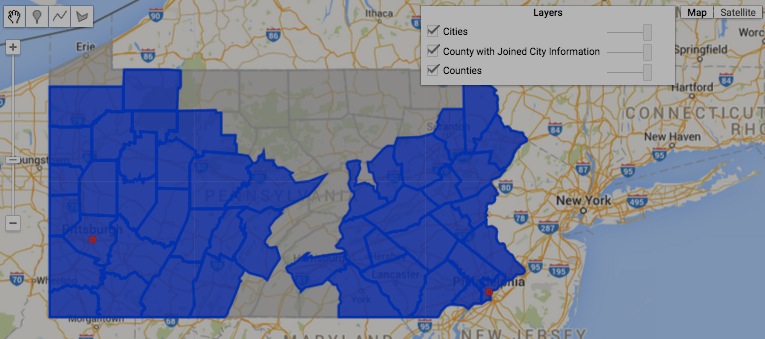
print( 'Joined Counties', JoinedFEATURES );

Map.centerObject( CountyFEATURES, 7 );

Map.addLayer( CountyFEATURES, {color:'aaaaaa'}, 'Counties' );

Map.addLayer( JoinedFEATURES, {color:'0000ff'}, 'County with Joined City Information' );

Map.addLayer( CityFEATURES, {color:'ff0000'}, 'Cities' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **RESETTING** VALUES

featureCollection.set creates new feature collection by replicating a specified feature collection after setting or resetting specified properties to specified values.

newFeatureCollection = oldFeatureCollection.set ( pairsOfPropertiesAndValues )

The specified properties and new values, given as a comma-separated sequence

(or a dictionary) of property name strings, each immediately followed by its new value.

The specified feature

collection

The new feature

collection

var ColoGEOMETRY = ee.Geometry.Polygon([[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

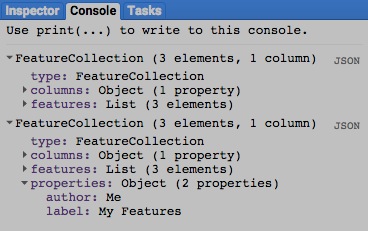
var WyomGEOMETRY = ee.Geometry.Polygon([[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var OldFEATURES = ee.FeatureCollection( [ColoGEOMETRY, WyomGEOMETRY, UtahGEOMETRY] );

var NewFEATURES = OldFEATURES.set( 'author','Me', 'label','My Features' );

print( OldFEATURES.getInfo( ), NewFEATURES.getInfo( ) );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **RESETTING** VALUES

featureCollection.setMulti creates new feature collection by replicating a specified feature collection

after setting or resetting specified properties to specified values.

newFeatureCollection = oldFeatureCollection.set ( dictionaryOfPropertiesAndValues )

The specified properties and new values, given as a

dictionary of property name strings and new values

The specified feature collection

The new feature collection

var ColoGEOMETRY = ee.Geometry.Polygon([[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

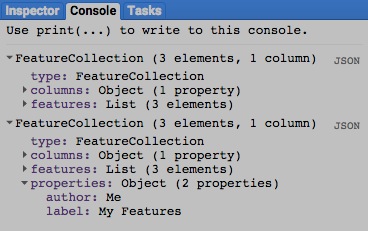
var WyomGEOMETRY = ee.Geometry.Polygon([[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var OldFEATURES = ee.FeatureCollection( [ColoGEOMETRY, WyomGEOMETRY, UtahGEOMETRY] );

var NewFEATURES = OldFEATURES.setMulti( {'author':'Me','label':'My Features'} );

print( OldFEATURES.getInfo( ), NewFEATURES.getInfo( ) );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **RESETTING** VALUES

featureCollection.remap creates new feature collection by replacing the values of a specified property for all features in a specified feature collection

with a specified set of new values, then deleting those features for which no new value has been specified.

newFeatureCollection = oldFeatureCollection.remap ( oldValues, newValues, oldProperty )

The specified set of new values,

given as a list of integers such

that each replaces the value at

the same position in **oldValues**.

The specified set

of values to be

replaced, given as

a list of integers

The name of the specified

property, given as a string

The specified feature collection

The new feature collection

var CountyFEATURES = ee.FeatureCollection('ft:1S4EB6319wWW2sWQDPhDvmSBIVrD3iEmCLYB7nMM'); // U.S. Counties

var OldFEATURES = CountyFEATURES.filterMetadata('StateName','equals','Oregon'); // Oregon Counties

var OldNUMBERS = [ 1, 3, 5, 7, 9,11,13,15,17,19,21,23,25,27,29,31,33,35,

37,39,41,43,45,47,49,51,53,55,57,59,61,63,65,67,69,71];

var NewNUMBERS = [ 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,

2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,];

var NewFEATURES = OldFEATURES.remap( OldNUMBERS, NewNUMBERS, 'CntyFips' );

var OldLIST = OldFEATURES.aggregate\_array( 'CntyFips' );

var NewLIST = NewFEATURES.aggregate\_array( 'CntyFips' );

print( 'Old County ID Numbers', OldLIST, OldFEATURES );

print( 'New County ID Numbers', NewLIST, NewFEATURES );

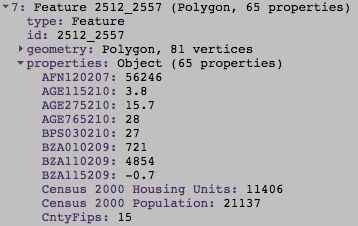
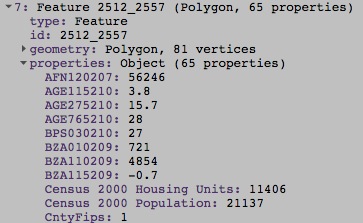
var OldIMAGE = OldFEATURES.reduceToImage( ['CntyFips'], ee.Reducer.first( ) );

var NewIMAGE = NewFEATURES.reduceToImage( ['CntyFips'], ee.Reducer.first( ) );

Map.centerObject(NewFEATURES, 6 );

Map.addLayer( OldIMAGE, {min:1, max:71, palette:'000000,ffffff', opacity:0.7}, 'Old ID Numbers' );

Map.addLayer( NewIMAGE, {min:1, max: 2, palette:'ff0000,00ff00', opacity:0.5}, 'New ID Numbers' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **REORGANIZING** THEM

featureCollection.sort creates a new feature collection by sorting the elements of a specified feature collection according to a specified property.

newFeatureCollection = oldFeatureCollection.sort ( property, *ascendingOrder*? )

The specified feature collection

The specified property

A Boolean set to True (only) if features are to be

sorted in ascending order. Default: True

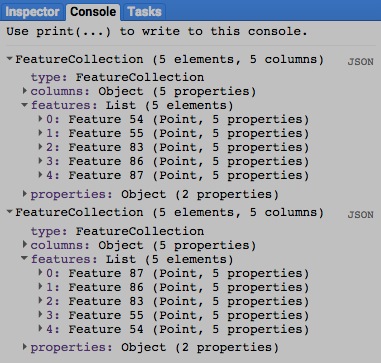
The new feature collection

var AllFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ); // U.S. Cities

var SomeFEATURES = AllFEATURES.filterMetadata( 'city\_name', 'contains', 'Cal' );

var SortedFEATURES = SomeFEATURES.sort( 'city\_name', false );

print( SomeFEATURES.getInfo(), SortedFEATURES.getInfo() );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **REORGANIZING** THEM

featureCollection.flatten creates a new one-dimensional feature collection that replicates the content of

a specified collection of feature collections in the order of its axes.

newFeatureCollection = oldFeatureCollection.flatten ( )

The new feature collection

The specified feature collection

var OldFEATURES = HOW TO CREATE A COLLECTION OF FEATURE COLLECTIONS

var NewFEATURES = OldFEATURES.flatten( );

print( OldFEATURES, NewFEATURES );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EDITING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **REORGANIZING** THEM

featureCollection.makeArray creates a new feature collection by replicating a specified feature collection after adding a new property that contains

a one-dimensional array of the values associated with a specified set of each feature’s numerical properties.

newFeatureCollection = oldFeatureCollection.select ( oldPropertyList*, newPropertyName* )

A names for the new property.

Default: 'array'

The specified properties, given

as a list of property names

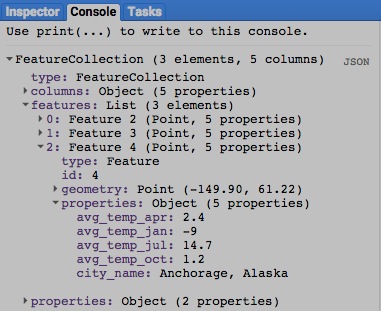
The specified feature collection

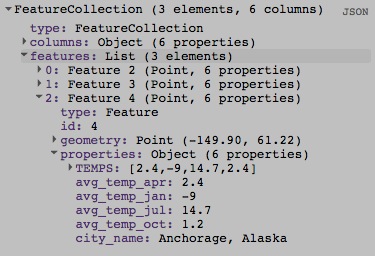
The new feature collection

var OldFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ).limit(3); // U.S. Cities

var NewFEATURES = OldFEATURES.makeArray( ['avg\_temp\_apr', 'avg\_temp\_jan', 'avg\_temp\_jul', 'avg\_temp\_apr'], 'TEMPS' );

print( OldFEATURES, NewFEATURES );





[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**TRANSFORMING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

featureCollection.classify creates a new feature collection by replicating a specified feature collection and adding to it a new property of a specified name

whose values are generated by classifying those of the specified feature collection according to a specified classifier.

newFeatureCollection = oldFeatureCollection.classify( classifier, newClassificationBandName )

The specified name of the

new property, given as a string

The specified classifier

The specified feature collection, whose properties

must be consistent with those called for by **classifier**

The new feature collection

var

ADD EXAMPLE

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**REPRODUCING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) AS **GEOMETRIES**

featureCollection.geometry creates a new geometry by merging those of all features in a specified feature collection.

newGeometry = oldFeatureCollection.geometry ( *errorMargin* )

An error margin indicating the spatial precision to be maintained

The new geometry

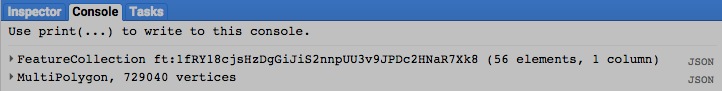
The specified feature collection

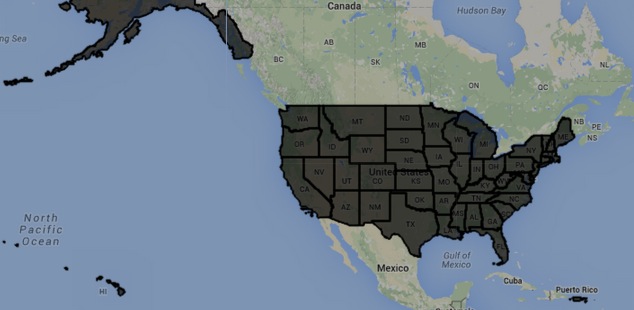
var TheFEATURES = ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNaR7Xk8'); // U.S. States

var TheGEOMETRY = TheFEATURES.geometry( );

var OneFEATURE = ee.Feature( TheGEOMETRY );

Map.addLayer( OneFEATURE );





[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**REPRODUCING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) AS **IMAGES**

featureCollection.reduceToImage creates a new image in which each pixel’s value is computed by applying a specified reducer to the values

of a selected property for whatever features from a specified feature collection intersect that pixel.

newImage = oldFeatureCollection.reduceToImage ( property, reducer )

The specified reducer

The name of the specified

property, given as a listed string

The specified feature collection

The new image

var CountyFEATURES = ee.FeatureCollection('ft:1S4EB6319wWW2sWQDPhDvmSBIVrD3iEmCLYB7nMM'); // U.S. Counties

var CountyFEATURES = CountyFEATURES.filterMetadata('StateName','equals','Pennsylvania'); // Pennsylvania Counties

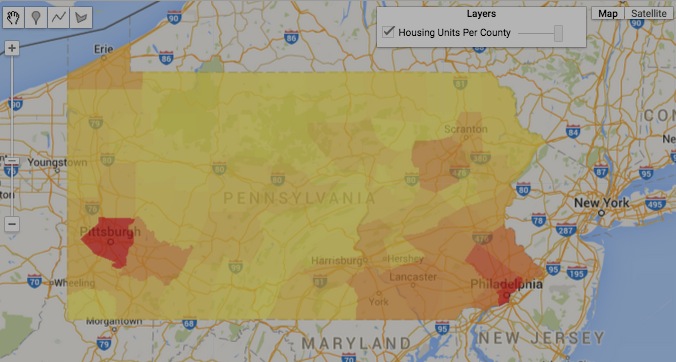
var CountyIMAGE = CountyFEATURES.reduceToImage( ['Census 2000 Housing Units'], ee.Reducer.first( ) );

print( CountyFEATURES, CountyIMAGE );

Map.centerObject( CountyFEATURES, 7 );

Map.addLayer( CountyIMAGE, {max:450000, palette:'ffff55,ff0000', opacity:0.7}, 'Housing Units Per County' );

How to make this work when more than one property is specified?



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**REPRODUCING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) AS **IMAGES**

image.paint creates a new image by assigning a specified value to all pixels of a specified image that coincide with the geometry of a specified feature collection.

newImage = oldImage.paint( *featureCollection, value, width*  )

The assumed width (in pixels) of geometry points, lines, or polygonal boundaries, given either as a number or a string naming a numerical property. Polygonal interiors are ignored unless *width* is omitted.

The specified feature collection

The specified image

The new image

The specified value, given either as a number or a string naming a numerical property

var ColoGEOMETRY = ee.Geometry.Polygon([[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05, 42.0]]);

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {'name':'Colorado', X:1} );

var UtahFEATURE = ee.Feature( UtahGEOMETRY, {'name':'Utah', X:2} );

var TheFEATURES = new ee.FeatureCollection( [ColoFEATURE, UtahFEATURE] );

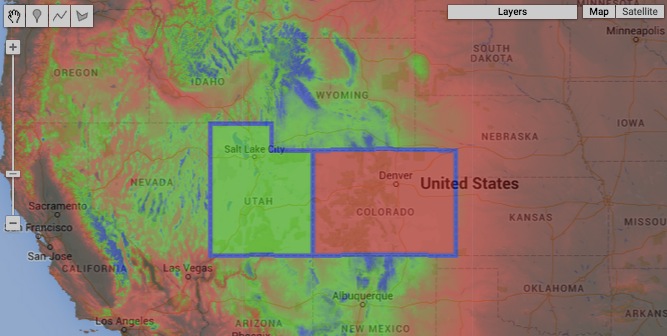
var FirstIMAGE = ee.Image( 'CGIAR/SRTM90\_V4' ).unitScale(0,999);

var SecondIMAGE = FirstIMAGE.paint ( TheFEATURES, 'X' );

var FinalIMAGE = SecondIMAGE.paint( TheFEATURES, 3, 5 );

Map.setCenter( -107, 41, 5 );

Map.addLayer(FinalIMAGE, { 'palette':'000000,FF0000,00FF00,0000FF', 'max':3, 'opacity':0.5 } );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**REPRODUCING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) AS **GOOGLE MAP OVERLAYS**

featureCollection.getMap creates a new object containing a specified feature collection, Map ID, and token from which to create a Google Maps overlay.

newObject = oldFeatureCollection.getMap ( color)

**000000 505050**

**FF0000 800000**

**FFAA00 DDCCAA**

**FFFF00 808000**

**00FF00 008000**

**00FFFF 008080**

**0000FF 000080**

**FF00FF 800080**

**FFFFFF 808080**

The specified feature collection

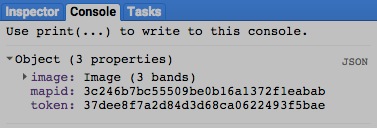
The new object

The specified color, given as the name of a color property or a string of three two-character codes that indicate the amount of redness, greenness, and blueness (in that order) comprising the color. The characters in each two-character code are digits that represent quantities ranging from 0 to 15 as follows: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Thus, A (or a) represents ten, B (or b) represents eleven, and so on while A0 represents 160, while A represents 161, and FF represents 255. To the right are some examples; each six-character code appears in the color it represents as described [here](http://www.w3schools.com/tags/ref_colormixer.asp). Default: ‘000000’.

var TheFEATURES = ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNaR7Xk8'); // U.S. States

var ThePACKAGE = TheFEATURES.getMap({ bands:'B4,B3,B2', min:0, max:0.2, opacity:0.7, gamma:0.5 });

print( ThePACKAGE );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**QUERYING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) FOR **FEATURES**

featureCollection.first creates a new feature collection element\* by reproducing the initial feature a specified feature collection.

newElement = oldFeatureCollection.first ( )

\* A feature collection element is subtly but significantly different from a feature. To generate the latter from the former, use **ee.Feature** as illustrated below.

The specified feature collection

The new element

var UnsortedFEATURES = ee.FeatureCollection( 'ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6' ); // CT Census Tracts

var SortedFEATURES = UnsortedFEATURES.sort('AREA', false);

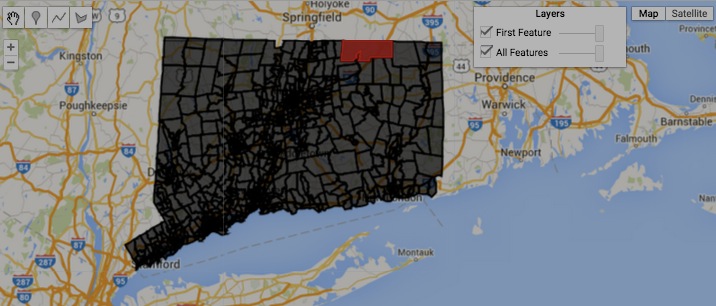
var FirstELEMENT = SortedFEATURES.first( );

var FirstFEATURE = ee.Feature( FirstELEMENT );

Map.centerObject( UnsortedFEATURES, 8 );

Map.addLayer( UnsortedFEATURES, null, 'All Features' );

Map.addLayer( FirstFEATURE, {color:"ee0000"}, 'First Feature' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**QUERYING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) FOR **FEATURES**

featureCollection.toList creates a new list of a specified number of consecutive features from a specified feature collection.

newList = oldFeatureCollection.toList ( howManyFeaturesToList, *howManyFeaturesToSkip* )

The index of the first feature to be listed. Default: 0

The number of consecutive features to be listed

The new list

The specified feature collection

var FeatureCOLLECTION = ee.FeatureCollection( 'ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6' ); // CT Census Tracts

var FeatureLIST = FeatureCOLLECTION.toList( 5,0 );

var InitialFEATURE = ee.Feature( FeatureLIST.get(0) );

var NextFEATURE = ee.Feature( FeatureLIST.get(1) );

var FinalFEATURE = ee.Feature( FeatureLIST.get(2) );

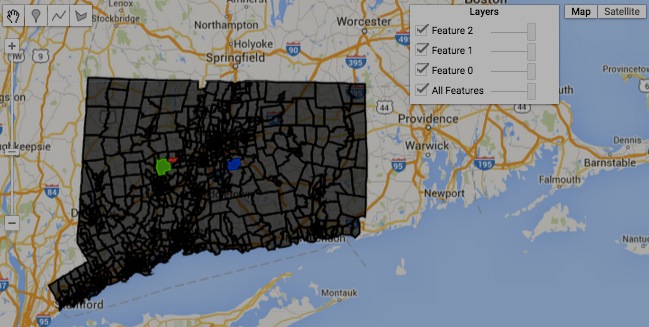
Map.centerObject( NextFEATURE, 9 );

Map.addLayer( FeatureCOLLECTION, null, 'All Features' );

Map.addLayer( InitialFEATURE, {color:"ee0000"}, 'Feature 0' );

Map.addLayer( NextFEATURE, {color:"00ee00"}, 'Feature 1' );

Map.addLayer( FinalFEATURE, {color:"0000ee"}, 'Feature 2' );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**QUERYING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) FOR **VALUES**

featureCollection.aggregate\_first creates a new object set to the value of a specified property of the first feature in a specified feature collection.

newObject = oldFeatureCollection.aggregate\_first( theProperty )

The new object

The specified feature collection

The name of the specified property, given as a string

var ColoGEOMETRY = ee.Geometry.Polygon( [[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var WyomGEOMETRY = ee.Geometry.Polygon( [[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

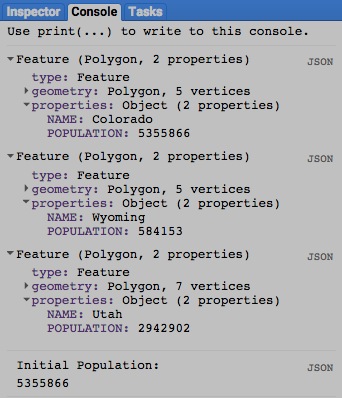
var UtahGEOMETRY = ee.Geometry.Polygon( [[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {NAME:'Colorado', POPULATION:5355866} );

var WyomFEATURE = ee.Feature( WyomGEOMETRY, {NAME:'Wyoming', POPULATION: 584153} );

var UtahFEATURE = ee.Feature( UtahGEOMETRY, {NAME:'Utah', POPULATION:2942902} );

var TheFEATURES = ee.FeatureCollection( [ColoFEATURE, WyomFEATURE, UtahFEATURE] );

var TheNUMBER = TheFEATURES.aggregate\_first( 'POPULATION' );

print( ColoFEATURE, WyomFEATURE, UtahFEATURE );

print( 'Initial Population:', TheNUMBER );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**QUERYING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) FOR **VALUES**

featureCollection.aggregate\_array creates a new list presenting the value of a specified property for each feature in a specified feature collection.

newList = oldFeatureCollection.aggregate\_array( theProperty )

The specified feature collection

The new list

The name of the specified property, given as a string

var ColoGEOMETRY = ee.Geometry.Polygon( [[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var WyomGEOMETRY = ee.Geometry.Polygon( [[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon( [[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

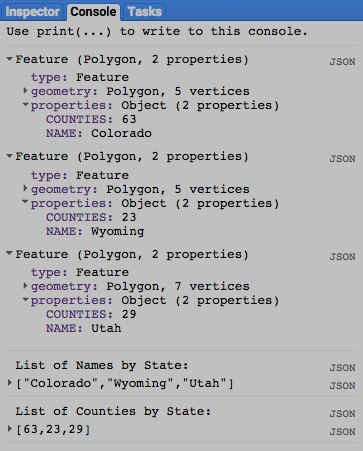
var ColoFEATURE = ee.Feature( ColoGEOMETRY, {NAME:'Colorado', COUNTIES:63} );

var WyomFEATURE = ee.Feature( WyomGEOMETRY, {NAME:'Wyoming', COUNTIES:23} );

var UtahFEATURE = ee.Feature( UtahGEOMETRY, {NAME:'Utah', COUNTIES:29} );

var TheFEATURES = ee.FeatureCollection( [ColoFEATURE, WyomFEATURE, UtahFEATURE] );

var NameLIST = TheFEATURES.aggregate\_array( 'NAME' );

var CountiesLIST = TheFEATURES.aggregate\_array( 'COUNTIES' );

print( ColoFEATURE, WyomFEATURE, UtahFEATURE );

print( 'List of Names by State:', NameLIST );

print( 'List of Counties by State:', CountiesLIST );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**SUMMARIZING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **PROPERTY VALUE**

featureCollection.aggregate\_stats creates a new dictionary indicating the maximum, mean, minimum, sample standard deviation,

sample variance, sum, sum squared, total count, total standard deviation total variance, valid count,

weight sum, and weighted sum of values associated with a specified numerical property of the features

in a specified feature collection.

newDictionary = oldFeatureCollection.aggregate\_stats( theProperty )

The new dictionary

The specified feature collection

The name of the specified property, given as a string

var ColoGEOMETRY = ee.Geometry.Polygon( [[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var WyomGEOMETRY = ee.Geometry.Polygon( [[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon( [[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {NAME:'Colorado', POPULATION:5355866} );

var WyomFEATURE = ee.Feature( WyomGEOMETRY, {NAME:'Wyoming', POPULATION: 584153} );

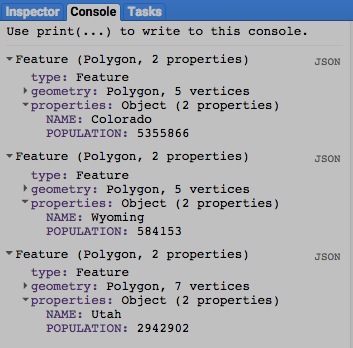
var UtahFEATURE = ee.Feature( UtahGEOMETRY, {NAME:'Utah', POPULATION:2942902} );

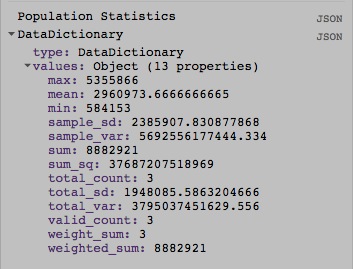
var TheFEATURES = ee.FeatureCollection( [ColoFEATURE, WyomFEATURE, UtahFEATURE] );

var TheDICTIONARY = TheFEATURES.aggregate\_stats( 'POPULATION' );

print( ColoFEATURE, WyomFEATURE, UtahFEATURE );

print( 'Population Statistics', TheDICTIONARY );





[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**SUMMARIZING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **PROPERTY VALUE**

featureCollection.aggregate\_histogram creates a new histogram indicating how many of the features in a specified feature collection

are associated with each particular value of a specified property of those features.

newHistogram = oldFeatureCollection.aggregate\_histogram( theProperty )

The specified feature collection

The name of the specified property, given as a string

The new histogram

var ColoGEOMETRY = ee.Geometry.Polygon( [[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var WyomGEOMETRY = ee.Geometry.Polygon( [[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon( [[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {NAME:'Colorado', CORNERS:4} );

var WyomFEATURE = ee.Feature( WyomGEOMETRY, {NAME:'Wyoming', CORNERS:4} );

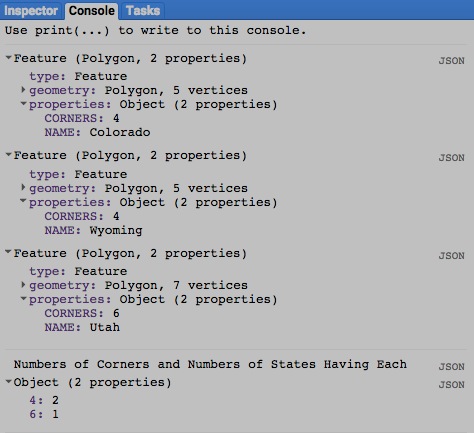
var UtahFEATURE = ee.Feature( UtahGEOMETRY, {NAME:'Utah', CORNERS:6} );

var TheFEATURES = ee.FeatureCollection( [ColoFEATURE, WyomFEATURE, UtahFEATURE] );

var TheHISTOGRAM = TheFEATURES.aggregate\_histogram( 'CORNERS' );

print( ColoFEATURE, WyomFEATURE, UtahFEATURE );

print( 'Numbers of Corners and Numbers of States Having Each', TheHISTOGRAM );

Map.centerObject( TheFEATURES, 5 );

Map.addLayer( TheFEATURES );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**SUMMARIZING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **PROPERTY VALUE**

featureCollection.aggregate\_count, … \_count\_distinct, … \_max, … \_min, … \_mean,

,… \_sample\_sd, and … \_total\_sd

… \_sum, … \_product, … \_sample\_var, … \_total\_var

all create a new number by applying a specified

statistical function to the values associated with

a specified numerical property for each

of the features in a specified feature collection

newNumber = oldFeatureCollection.aggregate\_theStatistic( theProperty )

The new number

The name of the specified property, given as a string

The specified statistic, given as one of the following:

**count**, **count\_distinct**, **max**, **min**, **sum**, **product**, **sample\_var**, **total\_var**, **sample\_sd**, or **total\_sd**

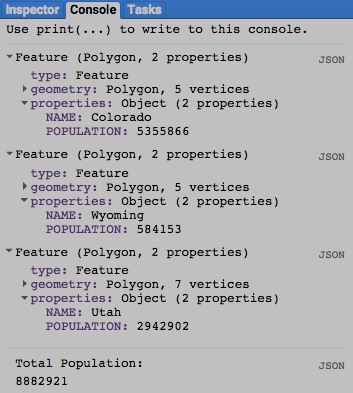
The specified feature collection

var ColoGEOMETRY = ee.Geometry.Polygon( [[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var WyomGEOMETRY = ee.Geometry.Polygon( [[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon( [[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {NAME:'Colorado', POPULATION:5355866} );

var WyomFEATURE = ee.Feature( WyomGEOMETRY, {NAME:'Wyoming', POPULATION: 584153} );

var UtahFEATURE = ee.Feature( UtahGEOMETRY, {NAME:'Utah', POPULATION:2942902} );

var TheFEATURES = ee.FeatureCollection( [ColoFEATURE, WyomFEATURE, UtahFEATURE] );

var TheNUMBER = TheFEATURES.aggregate\_sum( 'POPULATION' );

print( ColoFEATURE, WyomFEATURE, UtahFEATURE );

print( 'Total Population:', TheNUMBER );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**SUMMARIZING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) BY **PROPERTY VALUE**

featureCollection.reduceColumns creates a new dictionary in which each item’s value is computed by applying a specified reducer to the values

of a selected property for all features in a specified feature collection.

newDictionary = oldFeatureCollection.reduceColumns ( reducer, baseProperty*, weightProperty* )

The name of the specified property,

given as a listed string

The new dictionary

The specified reducer

The specified feature collection

The name of a weighting property, given as a listed string. Each feature’s **baseProperty** will be multiplied by its **weightProperty** value before being processed.

This affects only the following types of reducer: sum, mean, frequenctHistogram, and linearFit.

var ColoGEOMETRY = ee.Geometry.Polygon( [[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var WyomGEOMETRY = ee.Geometry.Polygon( [[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon( [[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {NAME:'Colorado', POPULATION:5355866, FEMALE\_SHARE:0.498} );

var WyomFEATURE = ee.Feature( WyomGEOMETRY, {NAME:'Wyoming', POPULATION: 584153, FEMALE\_SHARE:0.490} );

var UtahFEATURE = ee.Feature( UtahGEOMETRY, {NAME:'Utah', POPULATION:2942902, FEMALE\_SHARE:0.499} );

var TheFEATURES = ee.FeatureCollection( [ColoFEATURE, WyomFEATURE, UtahFEATURE] );

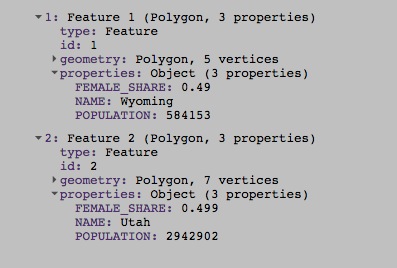
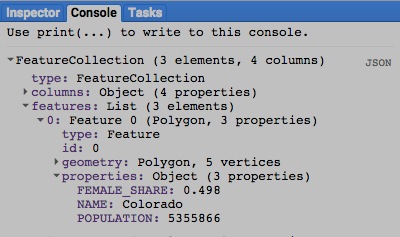
var UnweightedDICTIONARY = TheFEATURES.reduceColumns( ee.Reducer.sum() , ['POPULATION'], null );

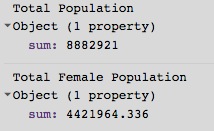
var WeightedDICTIONARY = TheFEATURES.reduceColumns( ee.Reducer.sum() , ['POPULATION'], ['FEMALE\_SHARE'] );

print( TheFEATURES );

print( 'Total Population', UnweightedDICTIONARY );

print( 'Total Female Population', WeightedDICTIONARY );





[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**COMPARING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

ee.Algorithms.IsEqual creates a new Boolean set to True (only) if the first of two specified feature collections.

is identical to the second in both structure and content.

newBoolean = ee.Algorithms.IsEqual ( 1stFeatureCollection, 2ndFeatureCollection )

The first specified feature collection

The second specified feature collection

The new Boolean

var ColoGEOMETRY = ee.Geometry.Polygon( [[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var WyomGEOMETRY = ee.Geometry.Polygon( [[-111.05,41],[-111.05,45],[-104.10,45],[-104.10,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon( [[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05,42]] );

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {NAME:'Colorado', POPULATION:5355866, FEMALE\_SHARE:0.498} );

var WyomFEATURE = ee.Feature( WyomGEOMETRY, {NAME:'Wyoming', POPULATION: 584153, FEMALE\_SHARE:0.490} );

var UtahFEATURE = ee.Feature( UtahGEOMETRY, {NAME:'Utah', POPULATION:2942902, FEMALE\_SHARE:0.499} );

var TheseFEATURES = ee.FeatureCollection( [ColoFEATURE, WyomFEATURE, UtahFEATURE] );

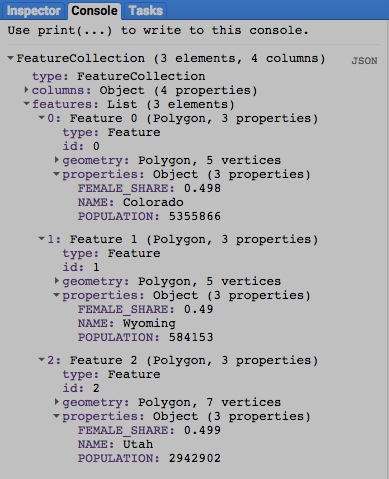
var ThoseFEATURES = ee.FeatureCollection( [WyomFEATURE, ColoFEATURE, UtahFEATURE] );

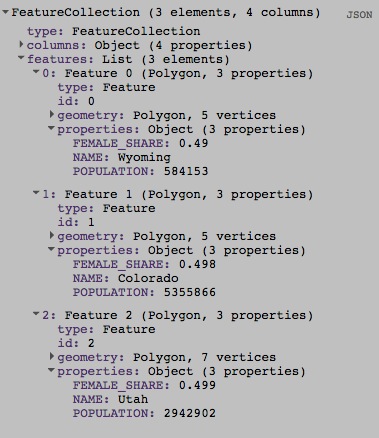
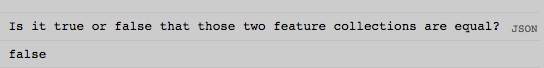
var TrueOrFalse = ee.Algorithms.IsEqual( TheseFEATURES, ThoseFEATURES );

print( TheseFEATURES, ThoseFEATURES );

print( 'Is it true or false that those two feature collections are equal?' );

print( TrueOrFalse );





[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PARALLEL PROCESSING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

featureCollection.map creates a new feature collection by applying a specified algorithm to each of the features in a specified feature collection.

(Because this algorithm is called only once, it should not include imperative functions like **print** or rely on external states.)

newFeatureCollection = oldFeatureCollection.map ( algorithm)

The specified feature collection

The new feature collection

The specified algorithm

function CentroidBufferingALGORITHM( AnyGivenPolygonalFEATURE )

{ var TheCentroidFEATURE = AnyGivenPolygonalFEATURE.centroid( );

return TheCentroidFEATURE.buffer( 10000 );

}

var OldFEATURES = ee.FeatureCollection('ft:1S4EB6319wWW2sWQDPhDvmSBIVrD3iEmCLYB7nMM'); // U.S. Counties

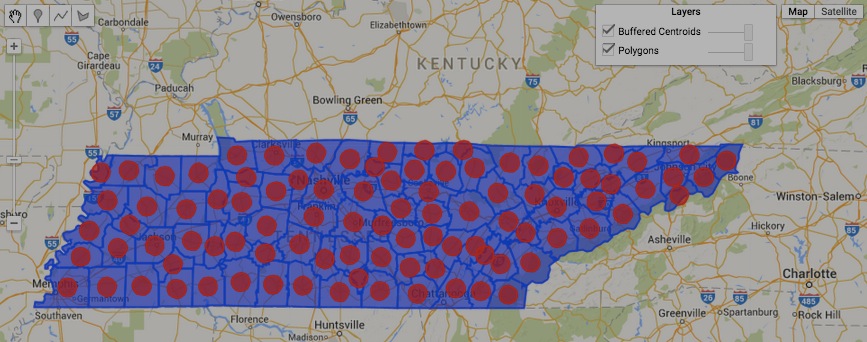
var OldFEATURES = OldFEATURES.filterMetadata('StateName','equals','Tennessee');

var NewFEATURES = OldFEATURES.map( CentroidBufferingALGORITHM );

Map.centerObject( NewFEATURES, 7 );

Map.addLayer( OldFEATURES, {color:'3333ff'}, 'Polygons' );

Map.addLayer( NewFEATURES, {color:'ff0000'}, 'Buffered Centroids');



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PARALLEL PROCESSING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

// Example of an algorithm within an algorithm

function CentroidBufferingALGORITHM( AnyGivenPolygonalFEATURE )

{ var TheCentroidFEATURE = AnyGivenPolygonalFEATURE.centroid( );

return TheCentroidFEATURE.buffer( 10000 );

}

function TheALGORITHM( )

{var NewFEATURES = OldFEATURES.map( CentroidBufferingALGORITHM );

return NewFEATURES;

}

var TheFEATURES = ee.FeatureCollection('ft:1S4EB6319wWW2sWQDPhDvmSBIVrD3iEmCLYB7nMM');

var OldFEATURES = TheFEATURES.filterMetadata('StateName','equals','Tennessee');

var FinalFEATURES = TheALGORITHM();

Map.centerObject( FinalFEATURES, 7 );

Map.addLayer( OldFEATURES, {color:'3333ff'}, 'Polygons' );

Map.addLayer( FinalFEATURES, {color:'ff0000'}, 'Buffered Centroids');

// Calculate the mean population of the Connecticut census tracts adjacent to each tract

function TheALGORITHM( TypicalFEATURE )

{ var TheGEOMETRY = TypicalFEATURE .geometry();

var NearbyFEATURES = OldTractFEATURES.filterBounds( TheGEOMETRY );

var NearbyPopNUMBER = NearbyFEATURES.aggregate\_mean( 'POP' );

var NewTypicalFEATURE = TypicalFEATURE.set( 'TotalNearbyPopulation', NearbyPopNUMBER );

return NewTypicalFEATURE;

}

var OriginalFEATURES = ee.FeatureCollection( 'ft:1xa2PvKTf7ynyAAEXEeHoltriaHFkyFJpvd74BLc6' );

var OldTractFEATURES = OriginalFEATURES.filterMetadata( 'POP','greater\_than',0 );

var NewTractFEATURES = OldTractFEATURES.map( TheALGORITHM );

print ( OldTractFEATURES, NewTractFEATURES );

Map.centerObject( OldTractFEATURES, 7 );

Map.addLayer( OldTractFEATURES, {color:"550000"} );

Map.addLayer( NewTractFEATURES, {color:"00aa00"} )

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**DESCRIBING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

featureCollection.errorMatrix creates a new confusion matrix from two specified properties of a specified feature collection:

- one representing known values in horizontal rows (axis 1), and

- the other representing vertical predicted values in vertical columns (axis 0).

newConfusionMatrix = oldFeatureCollection.errorMatrix( actual, predicted, *order* )

The index numbers to be assumed for

each of the confusion matrix’s axes,

given as a list. Default: [0,2,3, …]

The name of the property containing

known values, given as a string

The specified

feature collection

The new confusion matrix

The name of the property containing predicted values, given as a string

var AllIMAGES = ee.ImageCollection( 'LANDSAT/LT5\_L1T\_TOA' );

var DateSpecificIMAGES = AllIMAGES.filterDate( '2011-01-01', '2011-12-31' );

var LocationSpecificIMAGES = DateSpecificIMAGES.filterBounds( ee.Geometry.Point(-122.3942, 37.7295) );

var CloudSortedIMAGES = LocationSpecificIMAGES.sort('CLOUD\_COVER' );

var LandsatIMAGE = ee.Image(CloudSortedIMAGES.first() );

var AllBandCloudScoreIMAGE = ee.Algorithms.Landsat.simpleCloudScore( LandsatIMAGE );

var OneBandCloudScoreIMAGE = AllBandCloudScoreIMAGE.select('cloud');

var LandsatBandMaskIMAGE = LandsatIMAGE.mask();

var LandsatAnyMaskIMAGE = LandsatBandMaskIMAGE.reduce( 'min' );

var LandsatCloudMaskIMAGE = LandsatAnyMaskIMAGE.and( OneBandCloudScoreIMAGE.lte(50) );

var CloudfreeLandsatIMAGE = LandsatIMAGE.mask( LandsatCloudMaskIMAGE );

var ModisIMAGE = ee.Image( 'MODIS/051/MCD12Q1/2011\_01\_01' );

var ModisLandcoverIMAGE = ModisIMAGE.select( 'Land\_Cover\_Type\_1' );

var LandsatModisIMAGE = CloudfreeLandsatIMAGE.addBands( ModisLandcoverIMAGE );

var SampledFEATURES = LandsatModisIMAGE.sample( null,null,null,null,5000,0);

var TrainedFEATURES = SampledFEATURES.filter( ee.Filter.neq( 'B1', null ) );

var UntrainedCLASSIFIER = ee.Classifier.randomForest(10);

var TrainedCLASSIFIER = UntrainedCLASSIFIER.train( TrainedFEATURES, 'Land\_Cover\_Type\_1');

var ClassifiedIMAGE = CloudfreeLandsatIMAGE.classify( TrainedCLASSIFIER );

var ClassifierCONMATRIX = TrainedCLASSIFIER.confusionMatrix();

// Resample the input with a different random seed to get retrained features for validation

var LandsatModisIMAGE = CloudfreeLandsatIMAGE.addBands( ModisLandcoverIMAGE );

var SampledFEATURES = LandsatModisIMAGE.sample( null,null,null,null,5000,1 );

var RetrainedFEATURES = SampledFEATURES.filter( ee.Filter.neq( 'B1', null ) );

// Classify the retrained features

var ReclassifiedFEATURES = RetrainedFEATURES.classify( TrainedCLASSIFIER );

// Get a confusion matrix representing expected accuracy

var RetrainingCONMATRIX = ReclassifiedFEATURES.errorMatrix('Land\_Cover\_Type\_1', 'classification');

print('Classifier Confusion Matrix: ', ClassifierCONMATRIX);

print('Classifier Confusion Matrix Accuracy: ', ClassifierCONMATRIX.accuracy());

print('Retraining Confusion Matrix: ', RetrainingCONMATRIX );

print('Retraining Confusion Matrix Accuracy: ', RetrainingCONMATRIX.accuracy());

Map.centerObject( ClassifiedIMAGE, 10);

Map.addLayer( CloudfreeLandsatIMAGE, {bands: ['B3', 'B2', 'B1'], max: 0.4}, 'landsat' );

Map.addLayer( ClassifiedIMAGE, {min:0, max:17,

palette: ['aec3d4', // water

'152106', '225129', '369b47',

'30eb5b', '387242', // forest

'6a2325', 'c3aa69', 'b76031',

'd9903d', '91af40', // shrub, grass

'111149', // wetlands

'cdb33b', // croplands

'cc0013', // urban

'33280d', // crop mosaic

'd7cdcc', // snow and ice

'f7e084', // barren

'6f6f6f'] // tundra

}, 'classification' );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**DOCUMENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

ee.Algorithms.Describe and featureCollection.getInfo

each creates a JSON-compatible text object

representing a specified feature collection.

newObject = ee.Algorithms.Describe( oldFeatureCollection )

and oldFeatureCollection.getInfo( )

The specified feature collection

The new object

var ColoGEOMETRY = ee.Geometry.Polygon([[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05, 42.0]]);

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {'name':'Colorado', X:1} );

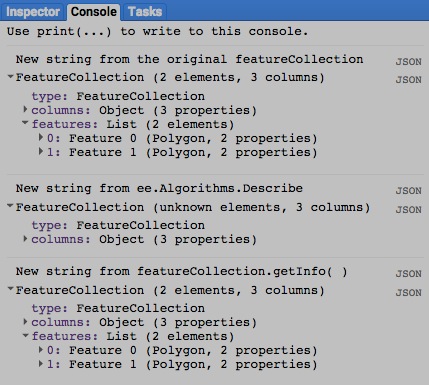
var UtahFEATURE = ee.Feature( UtahGEOMETRY, {'name':'Utah', X:2} );

var OldFEATURES = new ee.FeatureCollection( [ColoFEATURE, UtahFEATURE] );

print( 'New string from the original featureCollection', OldFEATURES );

print( 'New string from ee.Algorithms.Describe', ee.Algorithms.Describe( OldFEATURES ) );

print( 'New string from featureCollection.getInfo( )', OldFEATURES.getInfo( ) );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**DOCUMENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

featureCollection.toString and .serialize each creates a new string presenting information on a specified feature collection.

newString = oldfeatureCollection.toString ( )

and oldfeatureCollection.serialize( )

The specified feature collection

The new string

var ColoGEOMETRY = ee.Geometry.Polygon([[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05, 42.0]]);

var ColoFEATURE = ee.Feature( ColoGEOMETRY, {'name':'Colorado', X:1} );

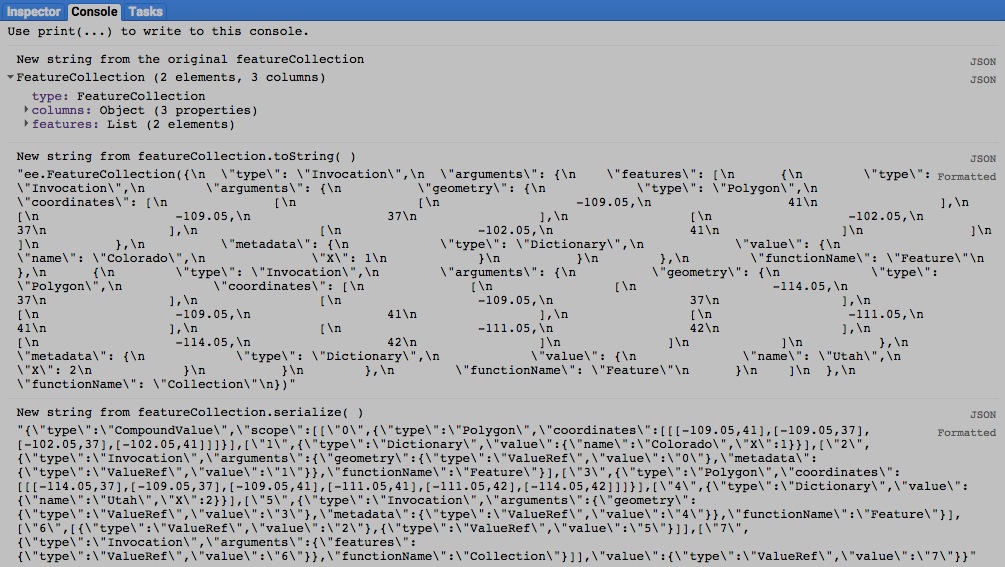
var UtahFEATURE = ee.Feature( UtahGEOMETRY, {'name':'Utah', X:2} );

var OldFEATURES = new ee.FeatureCollection( [ColoFEATURE, UtahFEATURE] );

print( 'New string from the original featureCollection', OldFEATURES );

print( 'New string from featureCollection.toString( )', OldFEATURES.toString( ) );

print( 'New string from featureCollection.serialize( )', OldFEATURES.serialize( ) );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PRESENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) IN **PRINT**

print ( featureCollection ) and console.log ( featureCollection ) present JSON-formatted text renditions of a specified

feature collection in the console.

print( oldFeatureCollection ) or console.log( oldFeatureCollection )

The specified feature collection

var ColoGEOMETRY = ee.Geometry.Polygon([[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05, 42.0]]);

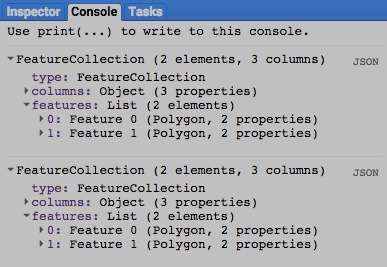
var ColoFEATURE = ee.Feature( ColoGEOMETRY, {'name':'Colorado', X:1} );

var UtahFEATURE = ee.Feature( UtahGEOMETRY, {'name':'Utah', X:2} );

var TheFEATURES = new ee.FeatureCollection( [ColoFEATURE, UtahFEATURE] );

print( TheFEATURES );

console.log( TheFEATURES );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PRESENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) IN **PRINT**

alert ( featureCollection ) and confirm( featureCollection ) presents the text representation of a specified string in a pop-up message box.

alert( oldFeatureCollection ) or confirm( oldFeatureCollection )

The specified feature collection

var ColoGEOMETRY = ee.Geometry.Polygon([[-109.05,41],[-109.05,37],[-102.05,37],[-102.05,41] ] );

var UtahGEOMETRY = ee.Geometry.Polygon([[-114.05,37],[-109.05,37],[-109.05,41],[-111.05,41],[-111.05,42],[-114.05, 42.0]]);

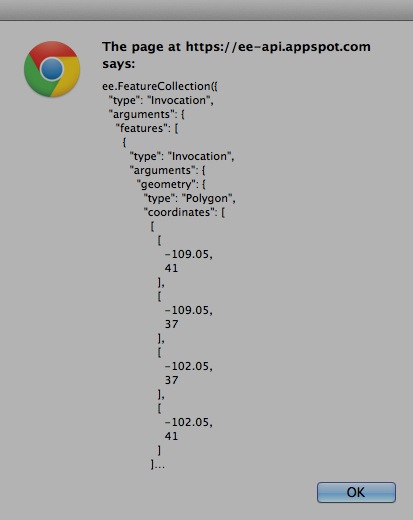
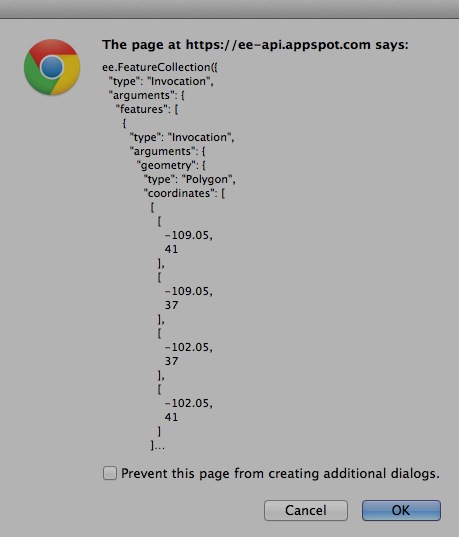
var ColoFEATURE = ee.Feature( ColoGEOMETRY, {'name':'Colorado', X:1} );

var UtahFEATURE = ee.Feature( UtahGEOMETRY, {'name':'Utah', X:2} );

var TheFEATURES = new ee.FeatureCollection( [ColoFEATURE, UtahFEATURE] );

alert( TheFEATURES );

confirm( TheFEATURES );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PRESENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) IN **MAPS**

Map.addLayer ( featureCollection ) adds a specified feature collection to the map as a layer.

Map.addLayer ( featureCollection*,  color, name, visibility* )

A Boolean set to TRUE (only) if the layer being displayed should initially be toggled on (visible) rather than off (invisible).Visibility can also be adjusted interactively by using the map’s layer listing.

The specified feature collection

The name for the layer to be displayed, given as a string. This name will appear on the map’s layer list listing.Default: "Layer N," where N is 1, 2, 3, and so on according to the order in which layers are added to the display.

**000000 505050**

**FF0000 800000**

**FFAA00 DDCCAA**

**FFFF00 808000**

**00FF00 008000**

**00FFFF 008080**

**0000FF 000080**

**FF00FF 800080**

**FFFFFF 808080**

The specified color, given as **{ color: X }** where X is a string of three two-character codes

that indicate the amount of redness, greenness, and blueness (in that order) comprising

the color. The characters in each two-character code are digits that represent quantities

ranging from 0 to 15 as follows: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Thus, A (or a) represents 10, B (or b) represents 11, and so on while A0 represents 160,

while A1 represents 161, and FF represents 255. To the right are some examples;

each six-character code appears in the color it represents as described [here](http://www.w3schools.com/tags/ref_colormixer.asp).

Default: ‘000000’.

var TheFEATURES = ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNaR7Xk8'); // U.S. States

Map.setCenter( -95.9766, 39.3683, 4 );

Map.addLayer ( TheFEATURES );

Map.addLayer ( TheFEATURES, {color:'996644'} );

Map.addLayer ( TheFEATURES, {color:'aa7755'}, 'Recolored' );

Map.addLayer ( TheFEATURES, {color:'bb8866'}, 'Hidden', false );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PRESENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) IN **MAPS**

featureCollection.draw creates a new image (for display purposes only) by reproducing a specified feature collection in with specified graphic settings.

newImage = featureCollection.draw( color*, pointSize, lineSize* )

The specified feature collection

The radius at which to display point markers in the specified

feature collection, given as an integer referring to pixel widths.

The new image

The width at which to display lines and polygonal borders in the specified feature collection, given as an integer referring to pixel widths.

**000000 505050**

**FF0000 800000**

**FFAA00 DDCCAA**

**FFFF00 808000**

**00FF00 008000**

**00FFFF 008080**

**0000FF 000080**

**FF00FF 800080**

**FFFFFF 808080**

**000000 505050**

**FF0000 800000**

**FFAA00 DDCCAA**

**FFFF00 808000**

**00FF00 008000**

**00FFFF 008080**

**0000FF 000080**

**FF00FF 800080**

**FFFFFF 808080**

The specified color, given as **{ color: X }** where X is a string of three two-character codes

that indicate the amount of redness, greenness, and blueness (in that order) comprising

the color. The characters in each two-character code are digits that represent quantities

ranging from 0 to 15 as follows: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Thus, A (or a) represents 10, B (or b) represents 11, and so on while A0 represents 160,

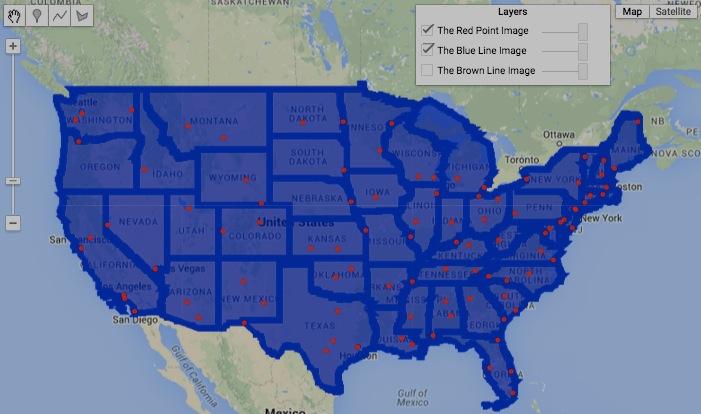
while A1 represents 161, and FF represents 255. To the right are some examples;

each six-character code appears in the color it represents as described [here](http://www.w3schools.com/tags/ref_colormixer.asp).

Default: ‘000000’.

var TheLINEFEATURES = ee.FeatureCollection('ft:1fRY18cjsHzDgGiJiS2nnpUU3v9JPDc2HNaR7Xk8'); // U.S. States

var ThePOINTFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ); // U.S. Cities

var TheBROWNLINEIMAGE = TheLINEFEATURES.draw( '663311' );

var TheBLUELINEIMAGE = TheLINEFEATURES.draw( '0000cc', 1, 7 );

var ThePOINTIMAGE = ThePOINTFEATURES.draw( 'ff0000', 2 );

Map.setCenter( -95.9766, 39.3683, 4 );

Map.addLayer ( TheBROWNLINEIMAGE, null, 'The Brown Line Image' );

Map.addLayer ( TheBLUELINEIMAGE, null, 'The Blue Line Image' );

Map.addLayer ( ThePOINTIMAGE, null, 'The Red Point Image' );

[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PRESENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) IN **CHARTS**

Chart.feature.histogram presents a chart

- whose horizontal (X) axis identifies the values of a specified property of a specified set of features;

- whose vertical (Y) axis identifies a number of features; and

- whose charted data indicate the number of features associated with each (grouping of) property value.

newChart = Chart.feature.histogram( oldFeatures, xProperty*, groupNumber, groupSize, maxSample* )

A minimum number of (numerical) features to use before constructing an initial histogram.

A minimum size of property group over which to count

(numerical) feature frequencies.

The specified property, given

as a property-name string

The new

chart

The specified set of features, given as a single

feature, a list of features, or a feature collection

A maximum number of property groups over which to count (numerical) feature frequencies.

var TheFEATURES = ee.FeatureCollection( 'ft:1R3so\_o1l-HtvZo4iFjMqpr1oNDxw9RexCDC6dkgU' );

var TheCHART = Chart.feature.histogram( TheFEATURES, 'Murder', 50 );

var TheCHART = TheCHART.setSeriesNames( ['Murder Rate'] );

var TheCHART = TheCHART.setOptions( { title: 'Numbers of Cities by Annual Murder Rate',

colors: ['deeppink'],

hAxis: {title:'Number of Murders'},

vAxis: {title:'Number of Cities'} } );

print( TheCHART );

print( TheFEATURES.getInfo( ) );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PRESENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) IN **CHARTS**

Chart.feature.byFeature presents a chart

- whose horizontal (X) axis identifies each of the features in a specified feature collection;

- whose vertical (Y) axis identifies the values associated with a specified property of those features; and

- whose charted data indicate the value of every feature for each of a specified series of properties.

newChart = Chart.feature.byFeature( oldFeatures*, xLabelProperty, ySeriesProperties* )

The specified property from

which to draw X-axis labels.

Default: “system:index”

The specified set of features, given

as a single feature, a list of features,

or a feature collection

The specified series of one or more properties whose values are

to be charted, given as a single property-name string or as a list

of such strings. Default: all properties other than **xLabelProperty**

The new chart

var TheFEATURES = ee.FeatureCollection('ft:1R3so\_o1l-HtvZo4iFjMqpr1oNDxw9RexCDC6dkgU'); // Selected Cities

var TheCHART = Chart.feature.byFeature( TheFEATURES, 'City', ['Robbery','Assault','Burglary'] );

var TheCHART = TheCHART.setOptions( { title: 'Number of Crimes per City by Type',

hAxis: { title: 'City' },

vAxis: { title: 'Number of Incidents' },

colors: ['lightgreen', 'red', 'darkblue'],

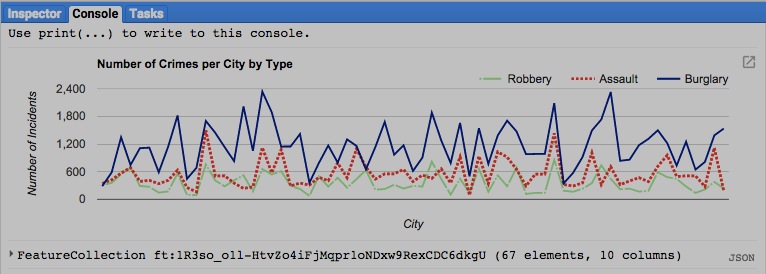
series: { 0: { lineWidth: 2, lineDashStyle: [2, 2, 20, 2, 20, 2]},

1: { lineWidth: 3, lineDashStyle: [3,2]},

2: { lineWidht: 4} } } );

print( TheCHART );

print( TheFEATURES.getInfo( ) );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PRESENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) IN **CHARTS**

Chart.feature.byProperty presents a chart

- whose horizontal (X) axis identifies each of the features in a specified feature collection;

- whose vertical (Y) axis identifies the values associated with a specified property of those features; and

- whose charted data indicate, for each of a specified series of properties, the property value of every feature.

newChart = Chart.feature.byProperty( oldFeatures*, xLabelProperties, ySeriesProperty* )

The specified set of features, given

as a single feature, a list of features,

or a feature collection

The specified property(ies) from which to draw X-

axis labels, given as a list of property-name strings or a property-label dictionary. Default: all properties

The property from which to draw

feature labels, given as a property-

name string. Default: “system:index”

The new chart

var TheFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ).limit(3); // Three Cities

var TheCHART = Chart.feature.byProperty( TheFEATURES, ['avg\_temp\_jan','avg\_temp\_jul'], 'city\_name' );

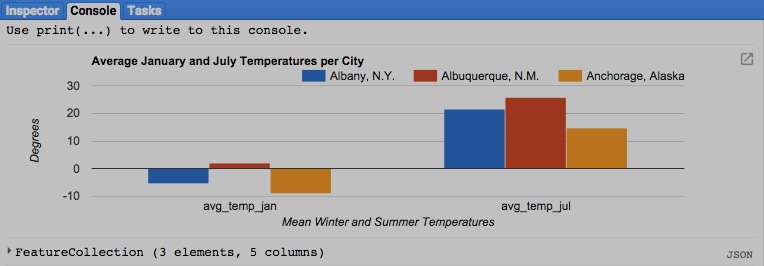
var TheCHART = TheCHART.setOptions( { title:'Average January and July Temperatures per City',

hAxis:{title:'Mean Winter and Summer Temperatures'},

vAxis:{title:'Degrees'} } );

print( TheCHART );

print( TheFEATURES.getInfo( ) );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**PRESENTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx) IN **CHARTS**

Chart.feature.groups presents a chart

- whose horizontal (X) axis identifies one of the properties of a specified list or collection of features;

- whose vertical (Y) axis identifies another of the properties of that specified feature collection; and

- whose charted data indicate the pair of property values for each of a specified series of feature groups.

newChart = Chart.feature.groups( oldFeatures*, xAxisProperty, yAxisProperty, groupProperty* )

The new

chart

The specified set of features, given

as a single feature, a list of features,

or a feature collection

The specified property from

which to draw X-axis labels,

given as a property-name string

The specified property from

which to draw Y-axis labels,

given as a property-name string

The specified property by which

to group series of features, given

as a property-name string

var TheFEATURES = ee.FeatureCollection( 'ft:1G3RZbWoTiCiYv\_LEwc7xKZq8aYoPZlL5\_KuVhyDM' ).limit(3); // Three Cities

var TheCHART = Chart.feature.groups(TheFEATURES, 'avg\_temp\_jul', 'avg\_temp\_jan', 'city\_name' );

var TheCHART = TheCHART.setOptions( { title:'Mean January and July Temperatures by City',

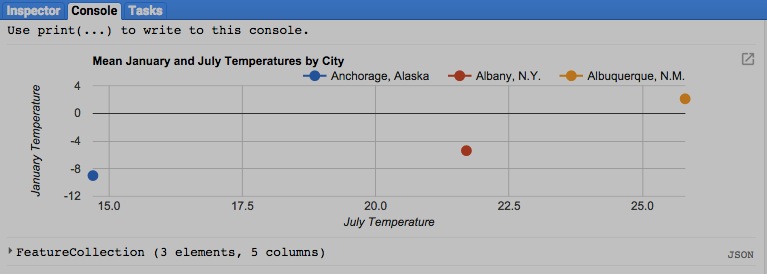
hAxis:{title:'July Temperature'},

vAxis:{title:'January Temperature'},

pointSize:10 } );

print( TheCHART );

print( TheFEATURES.getInfo( ) );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EXPORTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

featureCollection.getDownloadURL creates a link to a web address (URL) from which one can download a file (with a specified name)

containing a table of vertex coordinates (in a specified format) and specified attributes for the features

of a specified feature collection.

newLink = oldFeatureCollection.getDownloadURL ( *format, attributes, filename* )

The specified file name

The specified attributes,

given as a string of

attribute names

The specified format, given

as either "CSV" or "JSON."

Default: "CSV"

The specified feature collection

The new link

var ColoradoGeometry = ee.Geometry.Polygon( [[-109.05, 41], [-109.05, 37], [-102.05, 37], [-102.05, 41]] );

var ColoradoFeature = ee.Feature( ColoradoGeometry );

var ColoradoCollection = ee.Algorithms.Collection( [ColoradoFeature] );

var newString = ColoradoCollection.getDownloadURL();

print ( newString );



[GOOGLE EARTH ENGINE](EE01%20Earth%20Engine%20(EE).docx) [APPLICATION PROGRAMMING INTERFACE](EE05%20%20%20The%20EE%20API.docx) [CAPABILITIES](EE07%20%20%20%20%20%20API%20Capabilities.docx)

**EXPORTING** [FEATURE COLLECTION](#_top) [VARIABLES](EE13%20%20%20%20%20%20%20%20%20Variables.docx)

ee.data.makeTableDownloadUrl creates the link to a web address from which to download the file associated with a specified downloadID.

newLink = ee.data.makeTableDownloadUrl( downloadID)

The specified downloadID, as generated by **ee.data.getTableDownloadId**

The specified feature collection

The new link

ee.data.getTableDownloadId creates a new fownloadID (a dictionary containing the asset ID and token) for a file (of a specified name)

containing the vertex coordinates (in a specified format) and specified attributes for the features of a specified feature collection. This dictionary can the used as input to **ee.data.makeTableDownloadUrl**.

A much simpler alternative, however, is **featureCollection.getDownloadUrl.**

newDownloadID = ee.data.getTableDownloadId( *format, attributes, filename* )

The specified attributes,

given as a string of

attribute names

The specified

file name

The specified format, given

as either "CSV" or "JSON."

Default: "CSV"

The new downloadID

var TheDownloadID = ee.data.getTableDownloadId( );

var TheLINK = ee.data.makeTableDownloadUrl( TheDownloadID );

print( TheLINK );

still unclear about how to specify the feature collection to be downloaded