

INTRODUCTION TO GOOGLE EARTH ENGINE

/ From Pixels to Insights */*

What will you learn today?



You will acquire the basics of Google Earth Engine

Workshop Objectives:

- Get oriented with the JavaScript Code Editor
- Load an image collection and filter it to a relevant image
- Learn how to create a mosaic and composite image
- Work with Feature Collection
- Import, clip and export data
- Calculating Indices (e.g. NDVI)
- Computation on ImageCollection
- Reducers
- How to share your script

OUTLINE OF THIS WORKSHOP

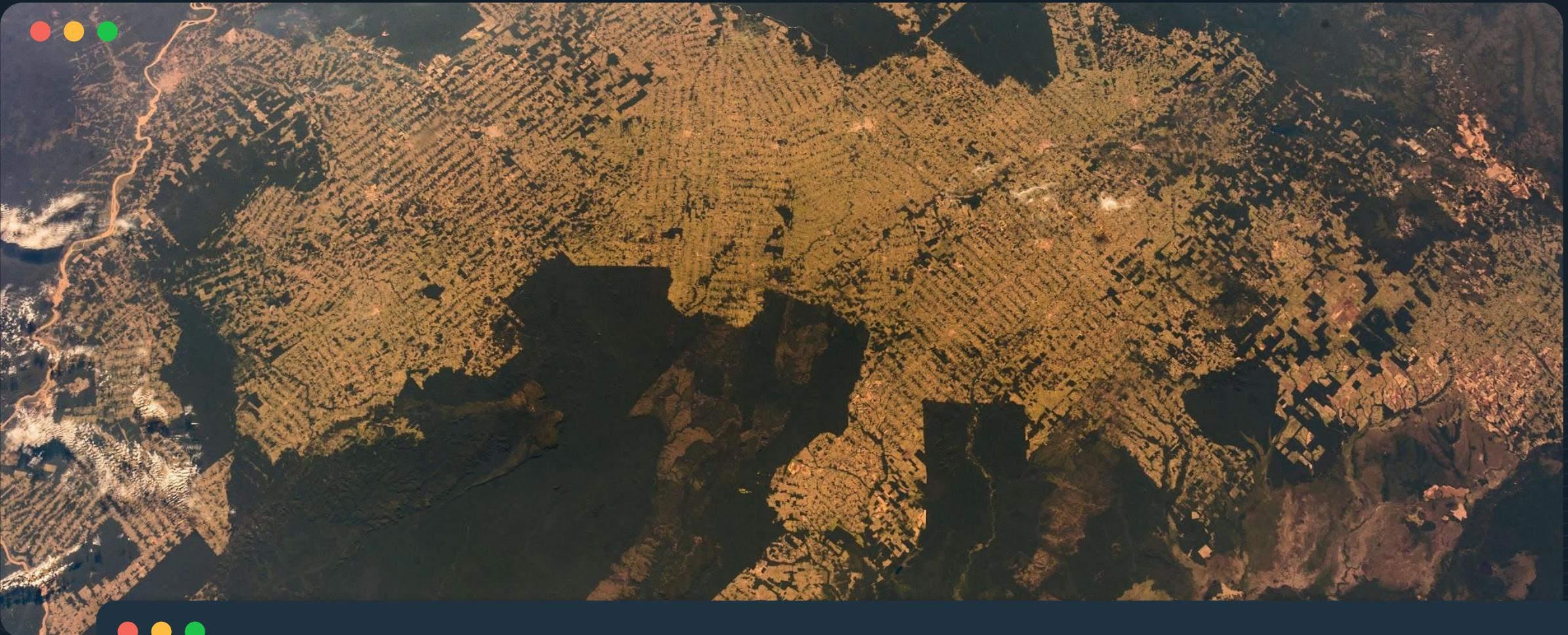
1 Intro to GEE
The "Why?"
And the "What?"

2 Components of
GEE
Intro to the Code Editor

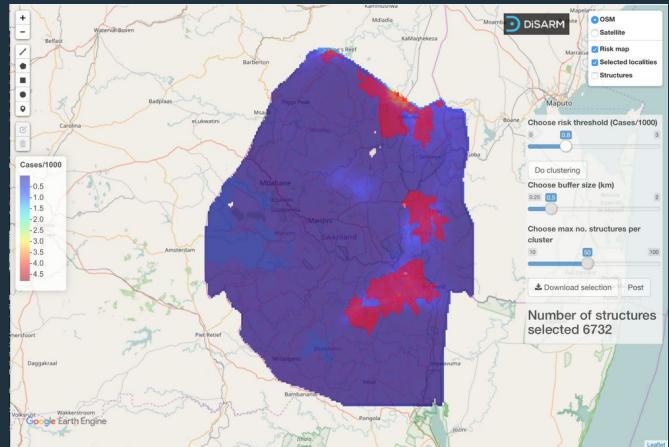
3 Code Along
Session
Hands-on using the Code Editor

4 Moving Forward
Available Resources

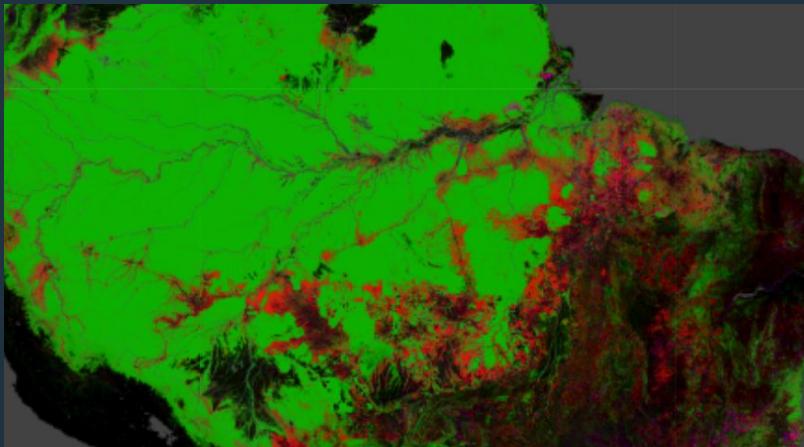




INTRODUCTION TO GEE



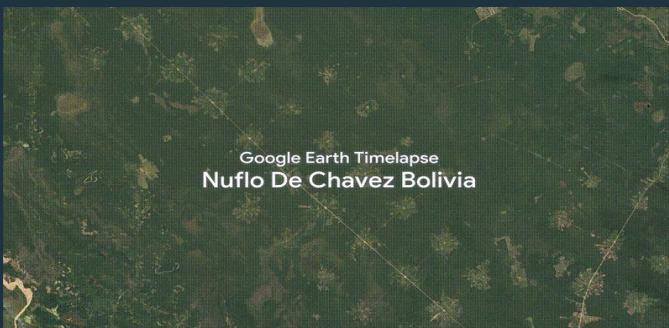
Infectious Disease



Deforestation



Water management



Land Cover Change



Biodiversity



Climate Change

In order to make progress on global problems, scientists and decision-makers need to be able to ask questions of increasingly large and complex geospatial datasets.

There are A LOT of data to manage

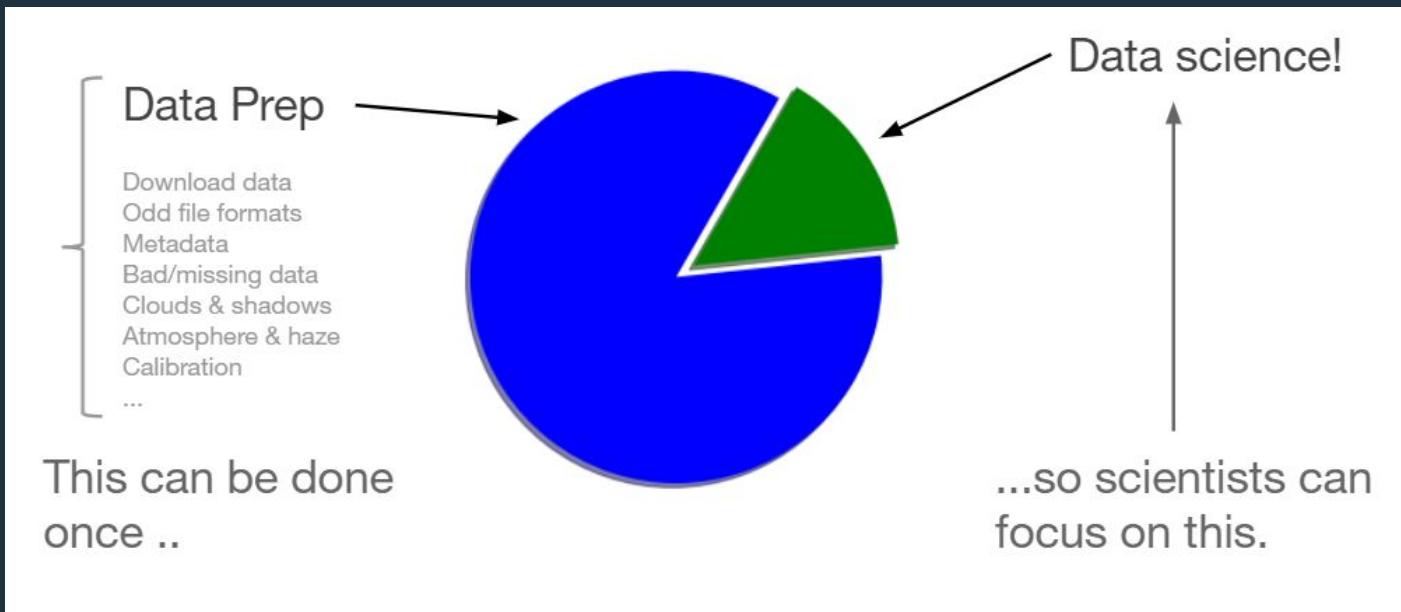


Using these datasets requires lots of computational power





The Classic Remote Sensing Workflow



Difference between Google Earth and Google Earth Engine



Google Earth

3D Viewer for
high-resolution imagery.
Targeted for everyone!

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



Google Earth Engine

Cloud-based platform for
remote sensing. Targeted for
scientists and researchers

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



What is Google Earth Engine?

Earth Engine is Google's platform for doing large-scale analysis of Earth data, including satellite imagery, census data, etc. It's composed of two big components: a collection of scientific data, and the computational power to effectively work with data at that scale.

The [Earth Engine API](#) is the interface through which users express the operations they would like Earth Engine to perform.

What is Google Earth Engine?



Data

An exhaustive catalog of remote sensing datasets, including multispectral, radar, aerial, climate, land cover, and vector.



Computation

Colocated data storage + computation



API

Simple, yet powerful JavaScript and Python API



Browser-based IDE

No software to download or keep up to date. All you need is a modest internet connection.



The Earth Engine Data Catalog



Landsat & Sentinel
10-30m, weekly



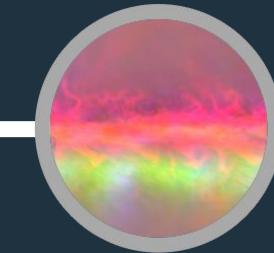
MODIS
250m daily



Vector Data
WDPA, TIGER, WHC



**Terrain &
Land Cover**



Weather & Climate
NOAA NCEP, OMI, ...

900+ public datasets

100+ datasets added yearly

70+ petabytes of data

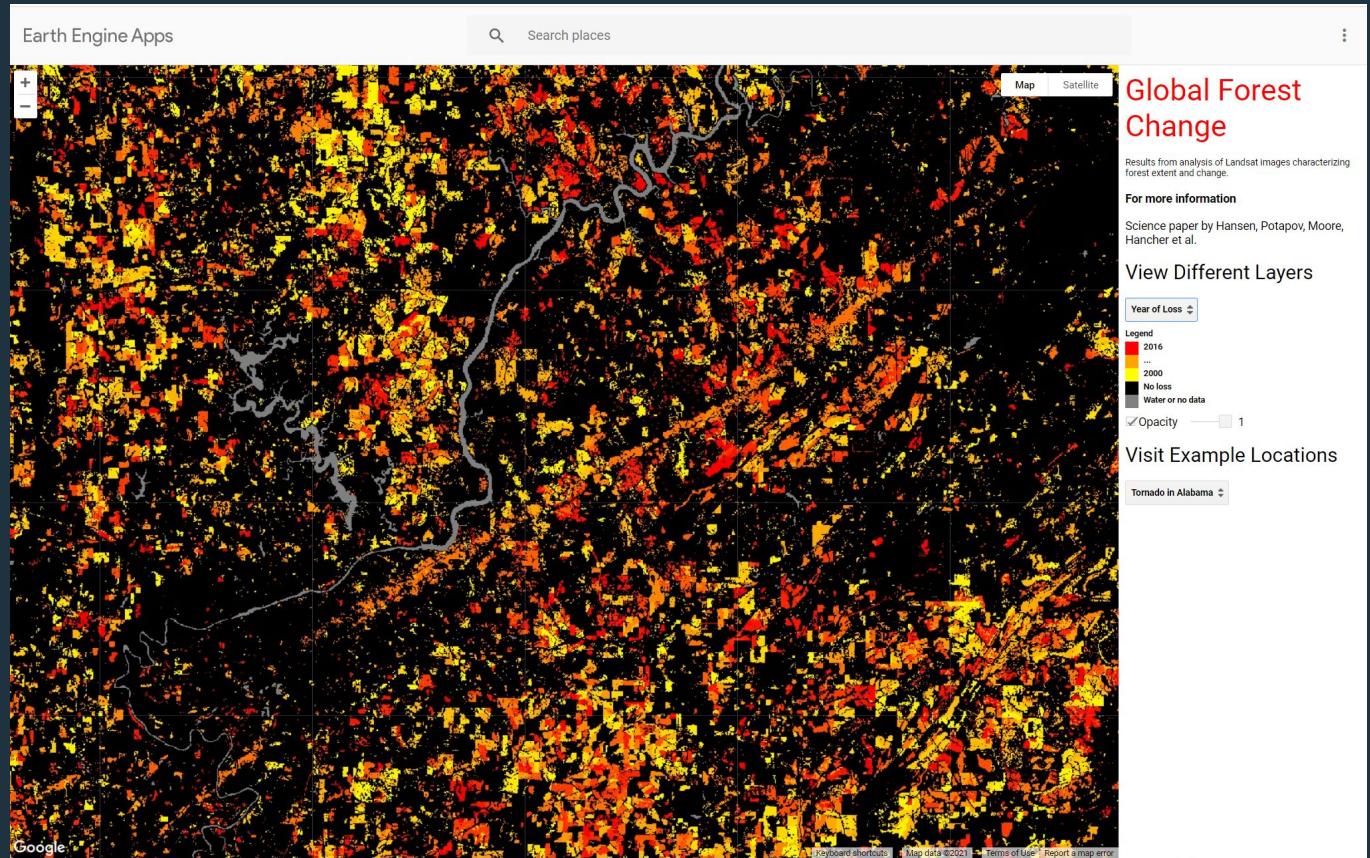
~2 PB of new data every month

1984





Earth Engine Apps



<https://google.earthengine.app/view/forest-change>

OUTLINE OF THIS WORKSHOP



1 Intro to GEE

The "Why?"
And the "What?"



2 Concepts

What is spatial data?



3 Code Along Session

Hands-on using the Code Editor



4 Moving Forward

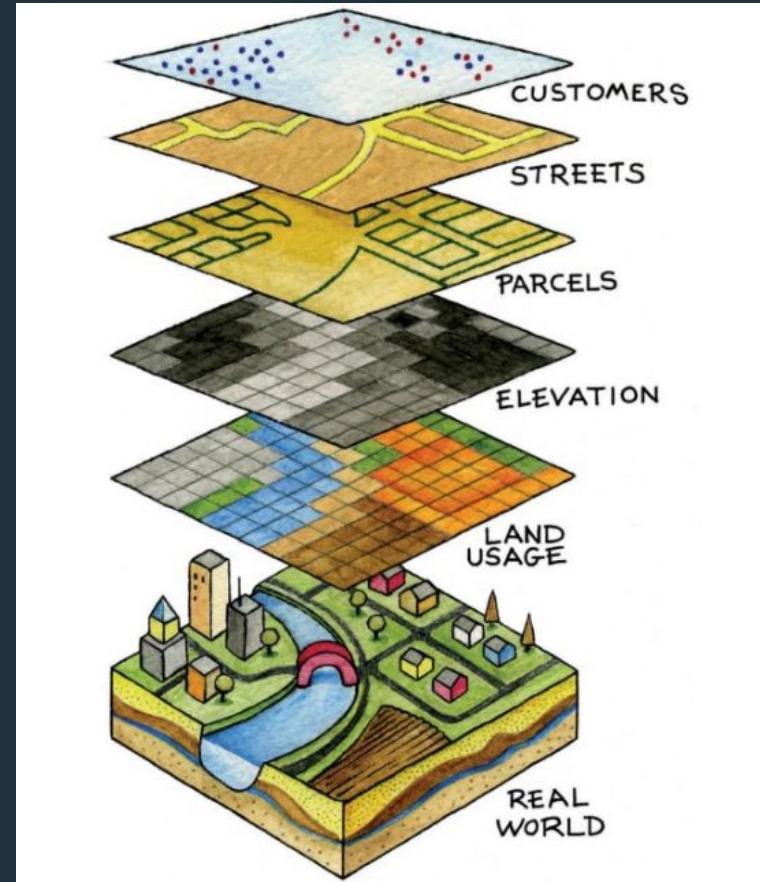
Available Resources

What is Spatial Data?

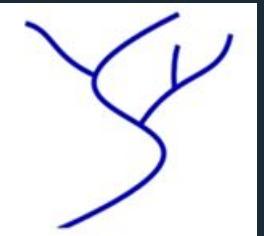


Information about the locations and shapes of geographic features and the relationships between them, usually stored as coordinates and topology

Any data that can be mapped

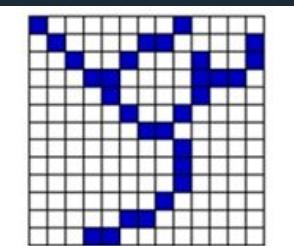


Spatial Data Formats



Vector

- Features with discrete shapes and boundaries (e.g. street, land parcel etc.)
- Can usually be opened in a web map or GIS software with no additional processing
- Represent information using points, line, and polygons
- .shp, .gpx, .kml, .kmz, .geojson, .osm, .bz2



Raster/Image

- Continuous surfaces with fuzzy boundaries or with qualities that change gradually over space (e.g. land cover, soil etc.)
- Pixelated data that can be added to a map, but cannot always be edited
- May need to be georeferenced
- Geotiff, .jpg (and other image formats)

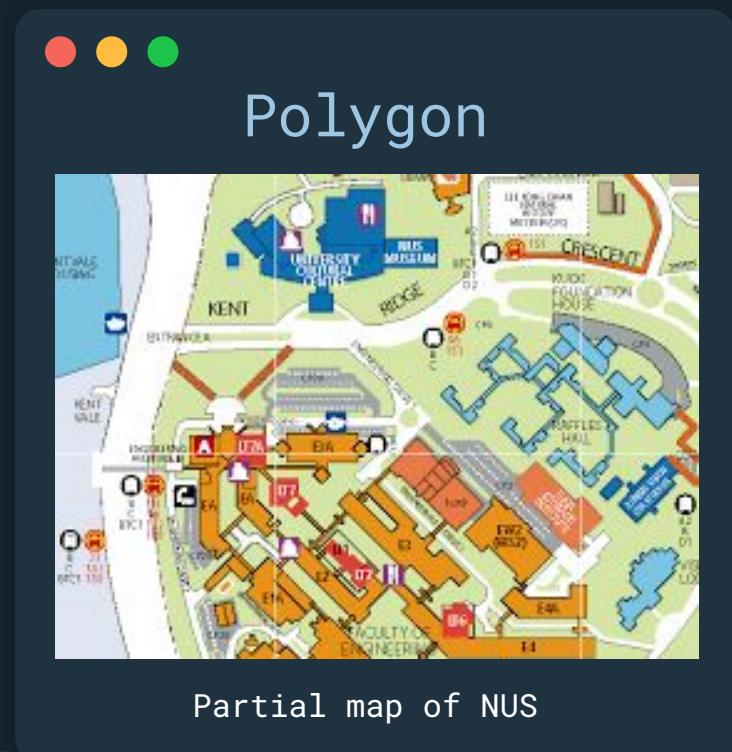


Address	City	State	ZIP Code
1134 Massachusetts Ave	Cambridge	MA	02138
290 Main St	Cambridge	MA	02142
47 Mount Auburn St	Cambridge	MA	02138
428 Massachusetts Ave	Cambridge	MA	02139
314 3rd St	Cambridge	MA	02142
675 W Kendall St	Cambridge	MA	02142
746 Massachusetts Ave	Cambridge	MA	02139
247 Cambridge St	Cambridge	MA	02141
276 Broadway	Cambridge	MA	02139
2370 Massachusetts Ave	Cambridge	MA	02140
1687 Massachusetts Ave	Cambridge	MA	02138
1722 Massachusetts Ave	Cambridge	MA	02138

Tabular

- Will need to be 'georeferenced' or 'geocoded' to translate coordinates or address into vector (points, lines, etc.)
- Can be joined to vector data if there is a common ID
- .csv, .txt, .xls, .xlsx, .tab, .sql, .ods

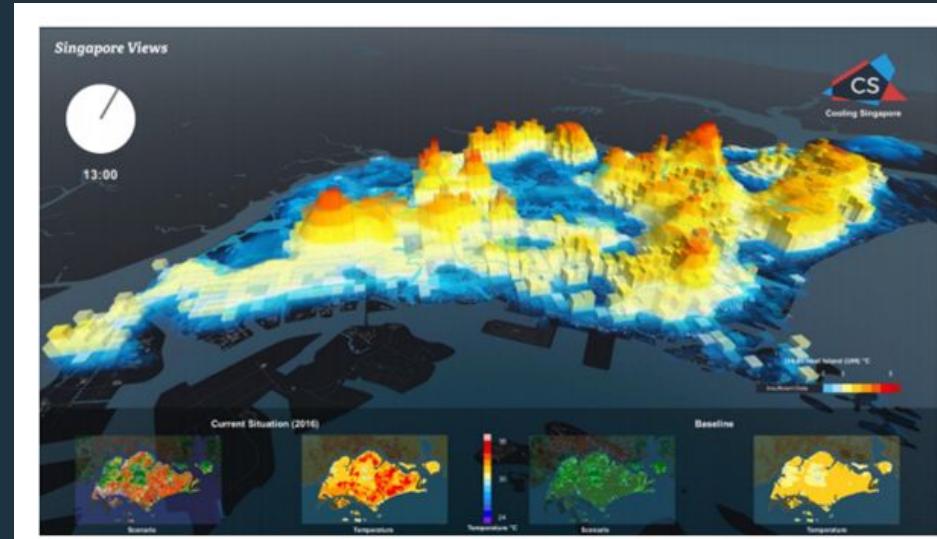
VECTOR: Points, Lines and Polygons



RASTER

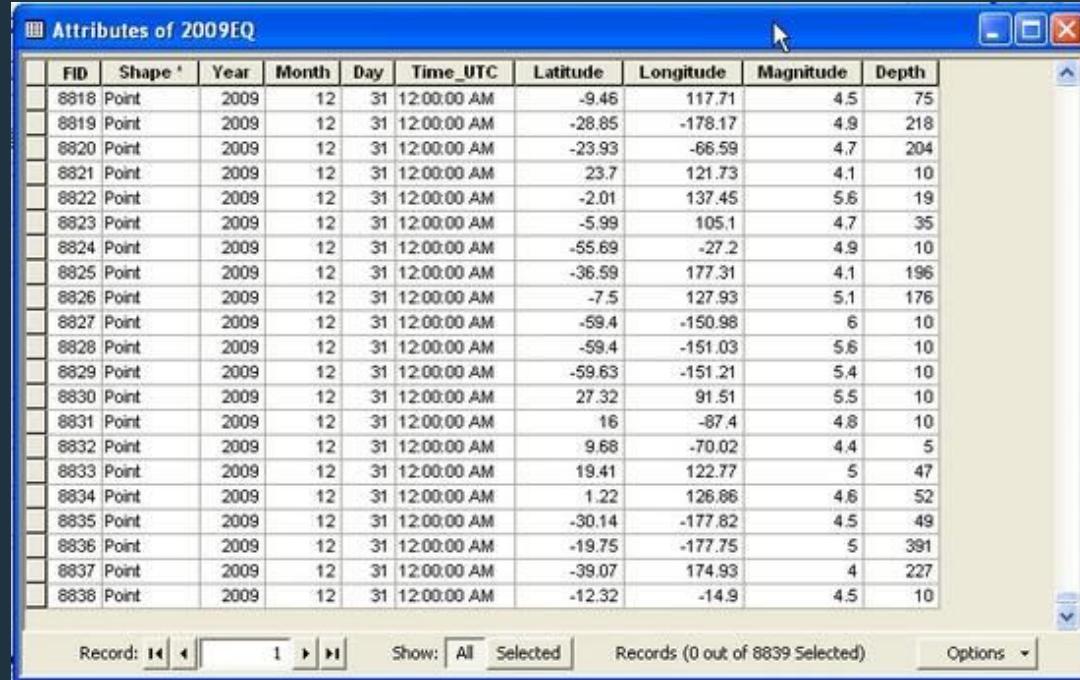


Google map of Singapore

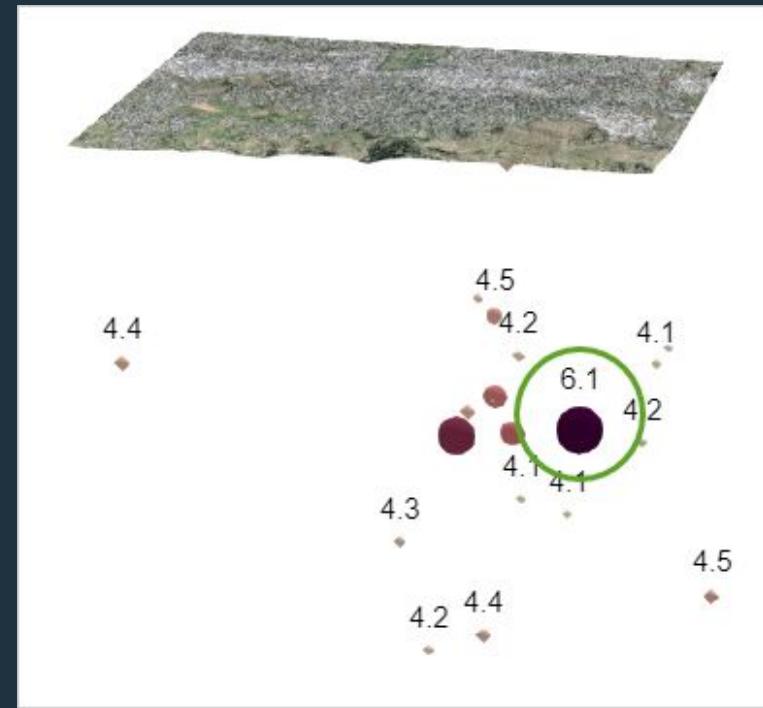


Urban Heat Island (UHI) Readings
produced by Cooling Singapore,
visualised by Singapore Views

Tabular Data

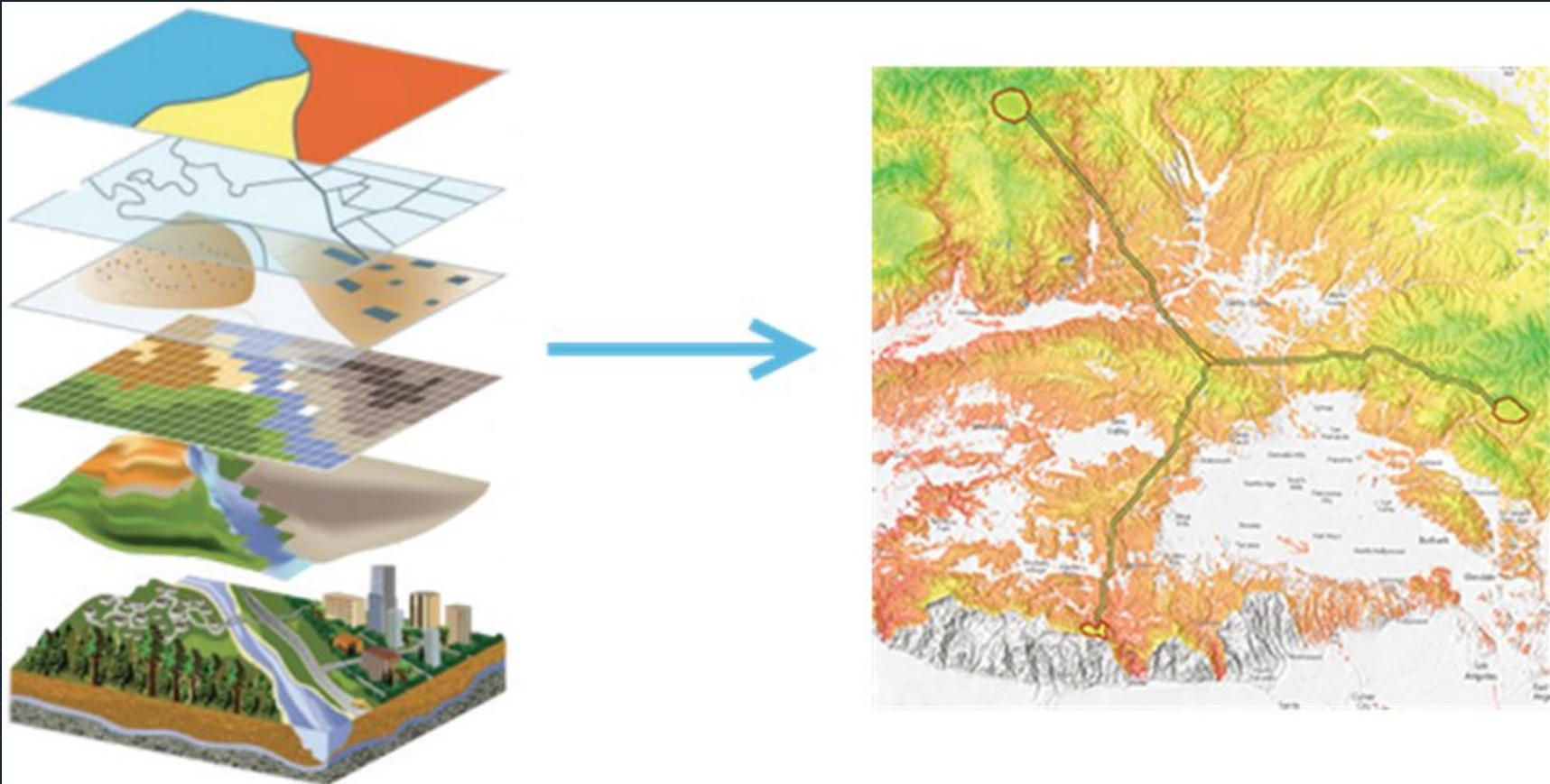


FID	Shape *	Year	Month	Day	Time_UTC	Latitude	Longitude	Magnitude	Depth
8818	Point	2009	12	31	12:00:00 AM	-9.46	117.71	4.5	75
8819	Point	2009	12	31	12:00:00 AM	-28.85	-178.17	4.9	218
8820	Point	2009	12	31	12:00:00 AM	-23.93	-66.59	4.7	204
8821	Point	2009	12	31	12:00:00 AM	23.7	121.73	4.1	10
8822	Point	2009	12	31	12:00:00 AM	-2.01	137.45	5.6	19
8823	Point	2009	12	31	12:00:00 AM	-5.99	105.1	4.7	35
8824	Point	2009	12	31	12:00:00 AM	-55.69	-27.2	4.9	10
8825	Point	2009	12	31	12:00:00 AM	-36.59	177.31	4.1	196
8826	Point	2009	12	31	12:00:00 AM	-7.5	127.93	5.1	176
8827	Point	2009	12	31	12:00:00 AM	-59.4	-150.98	6	10
8828	Point	2009	12	31	12:00:00 AM	-59.4	-151.03	5.6	10
8829	Point	2009	12	31	12:00:00 AM	-59.63	-151.21	5.4	10
8830	Point	2009	12	31	12:00:00 AM	27.32	91.51	5.5	10
8831	Point	2009	12	31	12:00:00 AM	16	-87.4	4.8	10
8832	Point	2009	12	31	12:00:00 AM	9.68	-70.02	4.4	5
8833	Point	2009	12	31	12:00:00 AM	19.41	122.77	5	47
8834	Point	2009	12	31	12:00:00 AM	1.22	126.86	4.6	52
8835	Point	2009	12	31	12:00:00 AM	-30.14	-177.82	4.5	49
8836	Point	2009	12	31	12:00:00 AM	-19.75	-177.75	5	391
8837	Point	2009	12	31	12:00:00 AM	-39.07	174.93	4	227
8838	Point	2009	12	31	12:00:00 AM	-12.32	-14.9	4.5	10



Depth and magnitude of Earthquakes
in Christchurch, New Zealand

Putting it together



QUIZ TIME!



ALL ABOUT SPATIAL DATA

Go to <https://kahoot.it/> and key in
the game pin that I'll be providing
you shortly

OUTLINE OF THIS WORKSHOP

1 Intro to GEE
The "Why?"
And the "What?"

2 Components of GEE
Intro to the Code Editor

3 Code Along Session
Hands-on using the Code Editor

4 Moving Forward
Available Resources



Programming Language for GEE



Javascript vs Python

- You will be learning the Earth Engine API
- Javascript API is the most mature and easiest to get started
 - No installation required
 - No need to worry about authentication
 - Very easy to share scripts, ask for help
 - Building and deploying apps is very easy
- Python API is much more powerful
 - Integrate with other data science libraries for data processing and plotting
 - Automate launching and managing Exports

The Earth Engine Code Editor

The screenshot shows the Google Earth Engine Code Editor interface. At the top, there are three red, yellow, and green window control buttons. Below them is the main workspace, which includes:

- Script manager**: A sidebar with links to "Owner (16)", "Writer", "Reader", and "Examples". Examples include "Image", "Image Collection", "Feature Collection", "Charts", "Arrays", "Primitive", "Cloud Masking", "Landsat457 Surface Reflectance", "Landsat8 Surface Reflectance", "Landsat8 TOA Reflectance QA Band", "MODIS Surface Reflectance QA Band", and "Sentinel2".
- API documentation**: A link to the Earth Engine API documentation.
- Asset manager**: A link to the asset manager.
- Search bar**: A search bar with placeholder text "Search places and datasets..." and a dropdown menu.
- Tool buttons**: Buttons for "Get Link", "Save", "Run", "Reset", and "Help".
- Code Editor**: The central area where code is written. The code shown is a script for cloud masking Sentinel-2 data:

```
1 // This example uses the Sentinel-2 QA band to cloud-mask  
2 // the collection. The Sentinel-2 cloud flags are less  
3 // selective, so the collection is also pre-filtered by the  
4 // CLOUDY_PIXEL_PERCENTAGE flag, to use only relatively  
5 // cloud-free granules.  
6  
7 // Function to mask clouds from the Sentinel-2 QA band.  
8 function mask2Clouds(image){  
9  var qa = image.select('QA60');  
10  
11  // Bits 10 and 11 are clouds and cirrus, respectively.  
12  var cloudBit = 1<(1<<10);  
13  var cirrusBitMask = 1<(1<<11);  
14  
15  // Both flags are set to 1 for pixels indicating clear sky.  
16  var mask = qa.bitwiseAnd(cloudBit).eq(0).and(  
17    qa.bitwiseAnd(cirrusBitMask).eq(0));  
18  
19  // Return the masked and scaled data, without the QA band.  
20  return image.updateMask(mask).divide(10000)  
21    .select('*').copyProperties(true).start();  
22}  
23  
24 // Run the function on the collection.  
25 var maskedCollection = imageCollection  
26   .map(mask2Clouds);  
27  
28 // Print the masked collection to the console.  
29 print(maskedCollection);
```
- Inspector**: A panel showing the results of the "print" command.
- Console**: A panel for writing commands to the console.
- Tasks**: A panel for managing tasks.
- Map**: A satellite map view with a zoom control (+/-) and geometry tools (magnifying glass, selection tool).
- Layer manager**: A panel for managing map layers.
- Help button**: A button for help.
- Feedback button**: A button for feedback.

At the bottom of the page is the URL code.earthengine.google.com.



Link to Repository

https://code.earthengine.google.com/?accept_repo=users/crispliming/GEE101_students

01 - Javascript Syntax

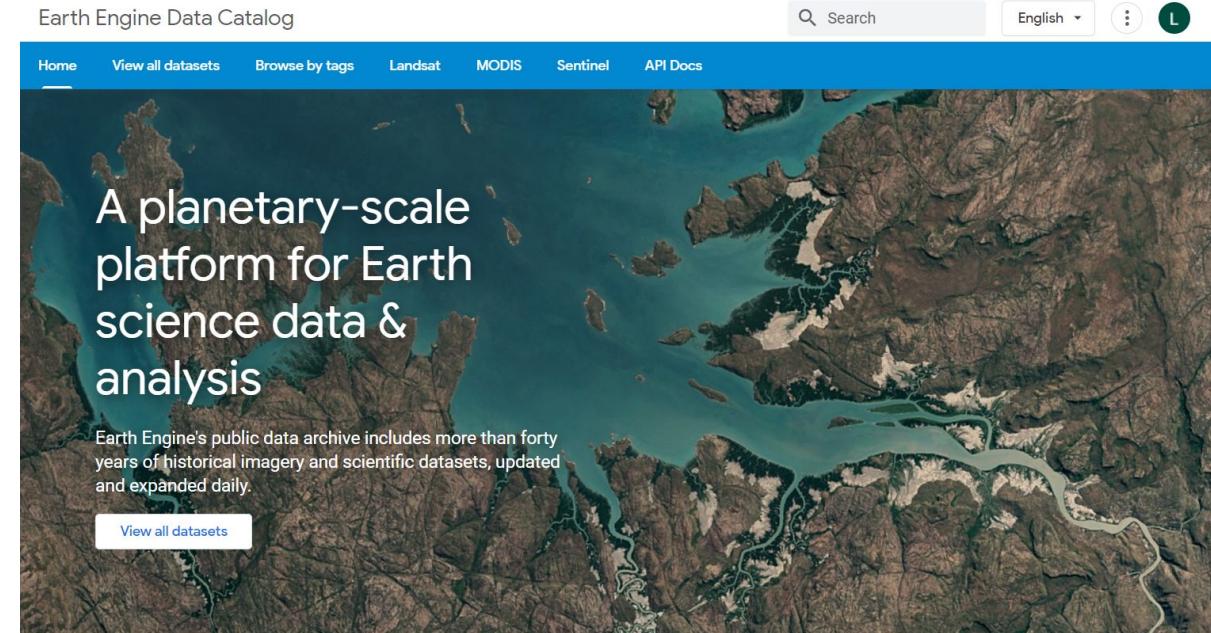


```
1 // The first thing you do in any programming language:  
2 print('Hello, World!');  
3  
4 // Line comments start with two forward slashes. Like this line.  
5  
6 /* Multi line comments start with a forward slash and a star,  
7 and end with a star and a forward slash. */  
8  
9 // Variables are used to store objects, and are defined using the keyword var.  
10 var the_answer = 42;  
11  
12 // String objects start and end with a single quote.  
13 var my_variable = 'I am a string';  
14  
15 // String objects can also start and end with double quotes.  
16 // But don't mix and match them.  
17 var my_other_variable = "I am also a string";  
18  
19 // Statements should end in a semi-colon, or the editor complains.  
20 var test = 'I feel incomplete...'  
21  
22 // Parentheses are used to pass parameters to functions.  
23 print('This string will print in the Console tab.');//  
24  
25 // Square brackets are used for selecting items within a list.  
26 // The zero index refers to the first item in the list.  
27 var my_list = ['eggplant', 'apple', 'wheat'];  
28 print(my_list[0]);  
29  
30 // Curly brackets (or braces) can be used to define dictionaries (key:value pairs)  
31 var my_dict = {'food': 'bread', 'color': 'red', 'number': the_answer};  
32 // Square brackets can be used to access dictionary items by key.  
33 print(my_dict['color']);  
34 // Or you can use the dot notation to get the same result.  
35 print(my_dict.color);  
36  
37 // Functions can be defined as a way to reuse code and make it easier to read  
38 var my_hello_function = function(string) {  
39     return 'Hello ' + string + '!';  
40 };  
41 print(my_hello_function('world'));
```

02 - Working with Image Collections



- Most datasets in Earth Engine comes as an [ImageCollection](#).
- An [ImageCollection](#) is a dataset that consists of images taken at different time and locations - usually from the same satellite or data provider.
- Earth Engine Data Catalog:
<https://developers.google.com/earth-engine/datasets>



02 - Working with Image Collections



- Example: Sentinel-2 Level 1C dataset
- Search for Sentinel-2 Level 1C dataset and you will find its COPERNICUS/S2_SR
- Visit the [Sentinel-2 Level 1C](#) page and see Explore in Earth Engine section to find the code snippet to load and visualise the collection
- This snippet is a great starting point for your work with this dataset.
- Click the [Copy Code Sample](#) button and paste the code into the code editor. Click Run and you will see the image tiles load in the map.

Explore in Earth Engine

```
/*
 * Function to mask clouds using the Sentinel-2 QA band
 * @param {ee.Image} image Sentinel-2 image
 * @return {ee.Image} cloud masked Sentinel-2 image
 */
function maskS2clouds(image) {
  var qa = image.select('A00');

  // Bits 10 and 11 are clouds and cirrus, respectively.
  var cloudBitMask = 1 << 10;
  var cirrusBitMask = 1 << 11;

  // Both flags should be set to zero, indicating clear conditions.
  var mask = qa.bitwiseAnd(cloudBitMask).eq(0)
    .and(qa.bitwiseAnd(cirrusBitMask).eq(0));

  return image.updateMask(mask).divide(10000);
}

// Map the function over one year of data and take the median.
// Load Sentinel-2 TOA reflectance data.
var dataset = ee.ImageCollection('COPERNICUS/S2')
  .filterDate('2018-01-01', '2018-06-30')
  // Pre-filter to get less cloudy granules.
  .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 20))
  .map(maskS2clouds);

var rgbVis = {
  min: 0.0,
  max: 0.3,
  bands: ['B4', 'B3', 'B2'],
};

Map.setCenter(-9.1695, 38.6917, 12);
Map.addLayer(dataset.median(), rgbVis, 'RGB');
```

[Open in Code Editor](#)



02 - Working with Image Collections

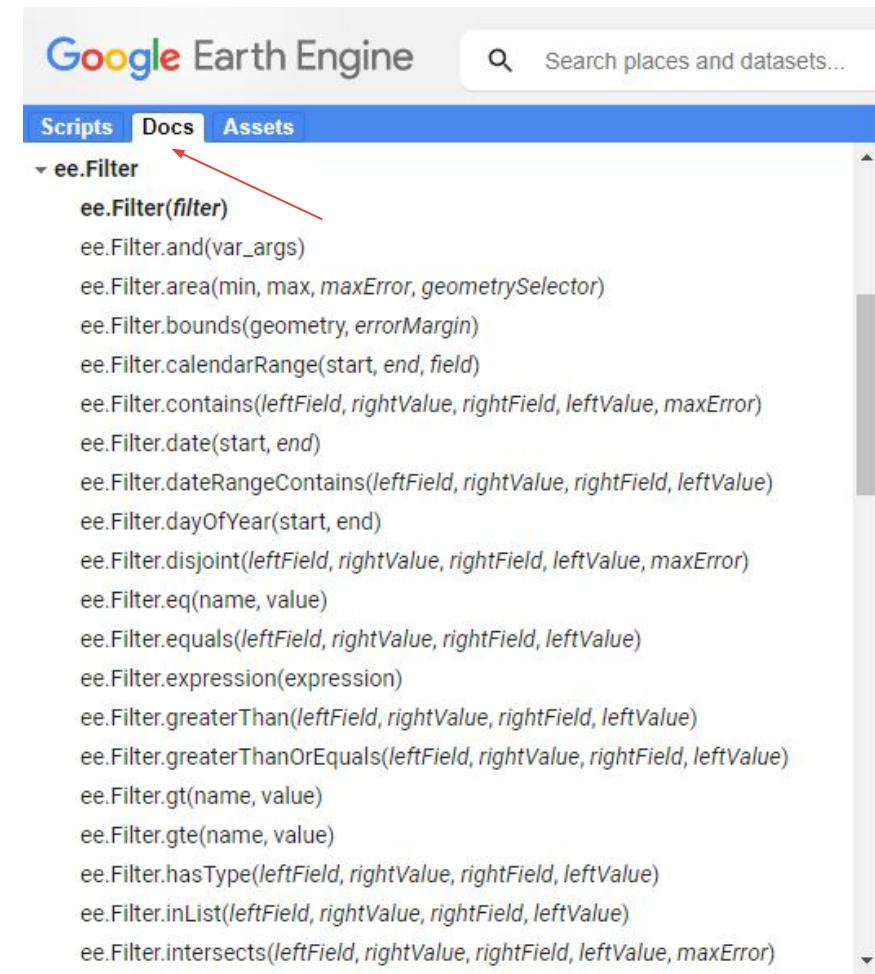


- In the code snippet, You will see a function `Map.setCenter()` which sets the viewport to a specific location and zoom level.
- The function takes the X coordinate (longitude), Y coordinate (latitude) and Zoom Level parameters.
- Replace the X and Y coordinates with the coordinates of Athens, Greece and click Run to see the images.

```
25 }
26 // Map the function over one year of data and take the median.
27 // Load Sentinel-2 TOA reflectance data.
28 var dataset = ee.ImageCollection('COPERNICUS/S2')
29   .filterDate('2018-01-01', '2018-06-30')
30   // Pre-filter to get less cloudy granules.
31   .filter(ee.Filter.lt('CLOUDY_PIXEL_PERCENTAGE', 20))
32   .map(maskS2clouds);
33
34
35 var rgbVis = {
36   min: 0.0,
37   max: 0.3,
38   bands: ['B4', 'B3', 'B2'],
39 };
40
41 Map.setCenter(-9.1695, 38.6917, 12);
42
43 Map.addLayer(dataset.median(), rgbVis, 'RGB');
```

03 - Filtering Image Collections

- The collection contains all imagery ever collected by the sensor.
- The entire collections are not very useful.
- Most applications require a subset of the images.
- We use **filters** to select the appropriate images.
- There are many types of filter functions, look at ee.Filter... module to see all available filters.
- Select a filter and then run the filter() function with the filter parameters.



03 - Filtering Image Collections



We will learn about 3 main types of filtering techniques

- **Filter by metadata:** You can apply a filter on the image metadata using filters such as `ee.Filter.eq()`, `ee.Filter.lt()` etc. You can filter by PATH/ROW values, Orbit number, Cloud cover etc.
- **Filter by date:** You can select images in a particular date range using filters such as `ee.Filter.date()`.
- **Filter by location:** You can select the subset of images with a bounding box, location or geometry using the `ee.Filter.bounds()`. You can also use the drawing tools to draw a geometry for filtering.

After applying the filters, you can use the `size()` function to check how many images match the filters.

The screenshot shows the Google Earth Engine interface. The top bar includes 'Google Earth Engine', a search bar, and various menu options like 'Get Link', 'Save', 'Run', 'Reset', 'Apps', and 'Inspector'. The left sidebar shows a 'Scripts' tab with a list of scripts: 'Owner (2)' (including '01 - Javascript Syntax', '02 - Image Collection', '02 - Image Collection (Complete)', '02 - Image Collection (Exercise)', '03 - Filtering Image Collection (C...)', '03 - Filtering Image Collection (C...', '03 - Filtering Image Collections (...'), 'users/crispliming/LandCover', 'Writer', and 'Image Compositing'); 'Writer' is currently selected. The 'Assets' tab is also visible. The main area contains a script titled '03 - Filtering Image Colle...' with the following code:

```
// Apply date filter on the results
var filtered2 = filtered.filter(
  ee.Filter.date('2021-01-01', '2021-01-31'));

// Lastly apply the location filter
var filtered3 = filtered2.filter(ee.Filter.bounds(geometry));

// Instead of applying filters one after the other, we can 'chain' them
// Use the & notation to apply all the filters together
var filtered = $2.filter($1).filter('CLOUDY_PIXEL_PERCENTAGE', 30)
  .filter(ee.Filter.date('2021-01-01', '2021-01-31'))
  .filter(ee.Filter.bounds(geometry));

print(filtered.size());

var rgbVis = {
  min: 0,
  max: 3000,
  bands: ['B4', 'B3', 'B2'],
};

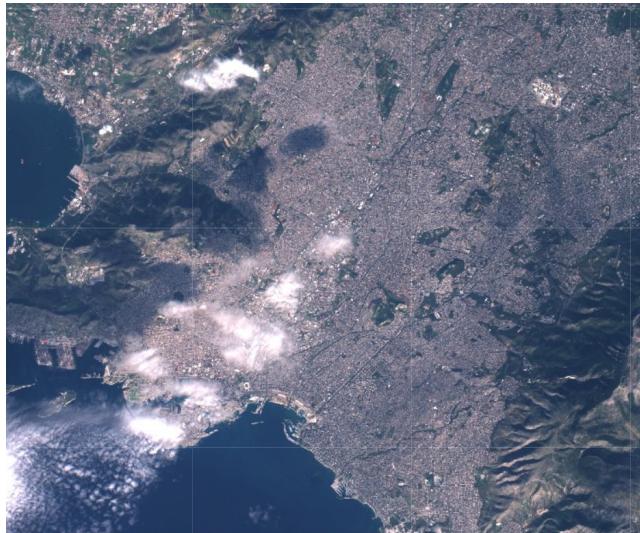
Map.addLayer(filtered, rgbVis, 'Filtered');
```

A red circle highlights the number '6' in the status bar at the bottom right of the code editor. Below the code editor is a satellite map of a coastal region, showing land and water. The bottom of the map has standard Google Earth Engine controls for 'Layers', 'Map', and 'Satellite'.

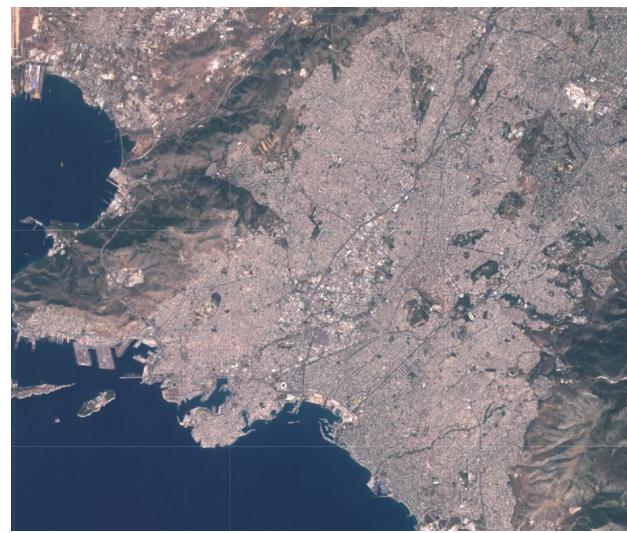
04 - Creating Mosaics and Composites from Image Collections



- The default order of the collection is by date. So when you display the collection, it implicitly creates a mosaic with the latest pixels on top. You can call `.mosaic()` on a `ImageCollection` to create a mosaic image from the pixels at the top.
- We can also create a composite image by applying selection criteria to each pixel from all pixels in the stack. Here we use the `median()` function to create a composite where each pixel value is the median of all pixels from the stack.



Mosaic

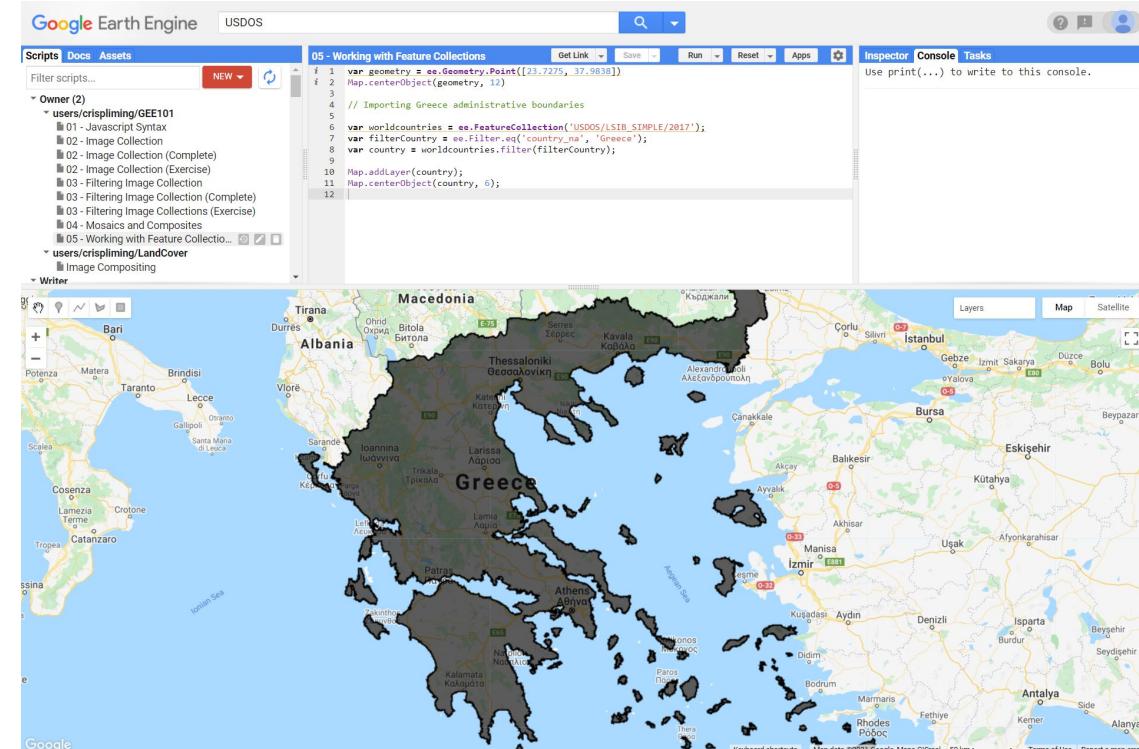


Median Composite

Tip: If you need to create a mosaic where the images are in a specific order, you can use the `.sort()` function to sort your collection by a property first.

05 - Working with Feature Collections

- Feature Collections are similar to Image Collections - but they contain *Features*, not images. They are equivalent to Vector Layers. We can load, filter and display Feature Collections using similar techniques that we have learned so far.
- Search for LSIB 2017: Large Scale International Boundary Polygons, Simplified and load the collection. This is a global collection that contains all country boundaries. We can apply a filter using the `country_na` property to get the specific country.



The screenshot shows the Google Earth Engine interface with a map centered on the Balkans and the Mediterranean Sea. A specific area of Greece is highlighted in dark gray, while the rest of the map is in a lighter greenish-blue. The map includes labels for various countries like Italy, Albania, Macedonia, and Turkey, along with numerous cities and geographical features. At the top, there's a header bar with the title '05 - Working with Feature Collections' and several navigation buttons. Below the header is a code editor window containing JavaScript code for working with Feature Collections, specifically filtering for Greece. The code is as follows:

```
var geometry = ee.Geometry.Point([23.7275, 37.9838]);
Map.centerObject(geometry, 12);

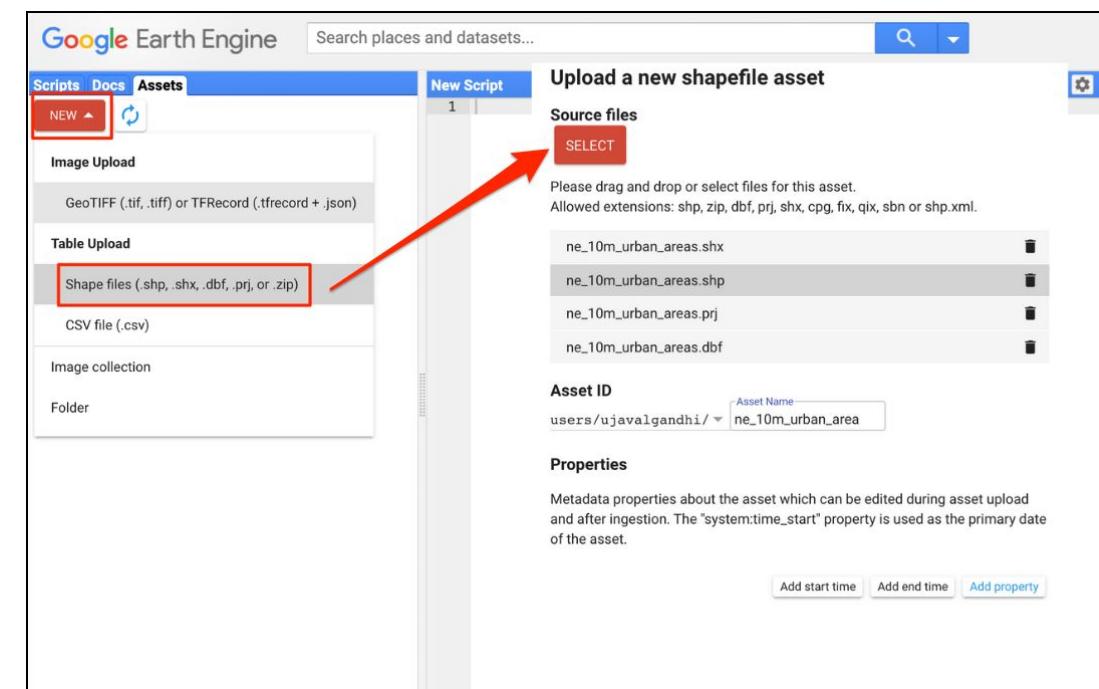
// Importing Greece administrative boundaries
var worldcountries = ee.FeatureCollection('USDS/LSIB_SIMPLE/2017');
var filterCountry = ee.Filter.eq('country_na', 'Greece');
var country = worldcountries.filter(filterCountry);

Map.addLayer(country);
Map.centerObject(country, 6);
```

06 - Importing Data



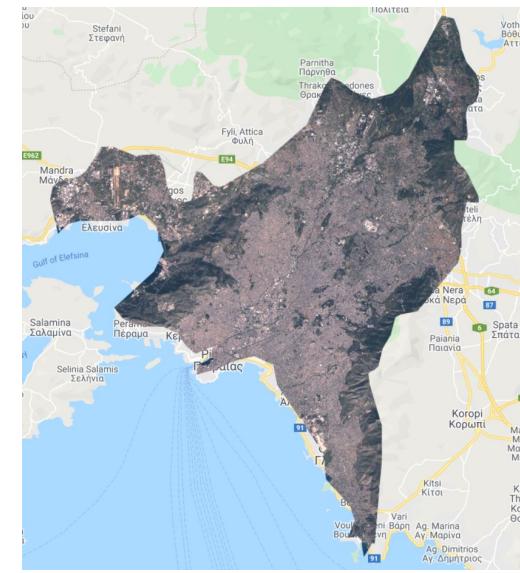
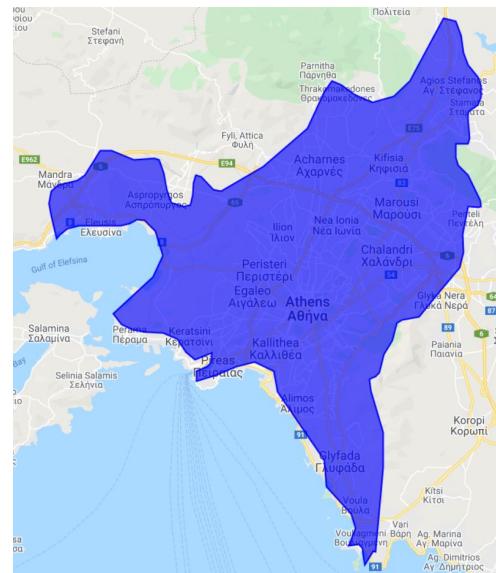
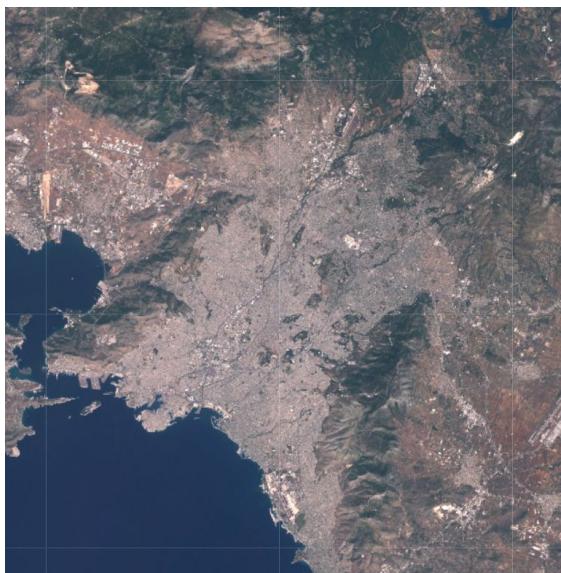
- You can import vector or raster data into Earth Engine. We will now import a shapefile of [Urban Areas](#) for Natural Earth.
- Unzip the ne_10m_urban_areas.zip into a folder on your computer.
- In the Code Editor, go to Assets → New → *Table Upload* → *Shape Files*. Select the .shp, .shx, .dbf and .prj files.
- Enter ne_10m_urban_areas as the Asset Name and click *Upload*.
- Once the upload and ingest finishes, you will have a new asset in the Assets tab.
- The shapefile is imported as a Feature Collection in Earth Engine. Select the ne_10m_urban_areas asset and click *Import*.
- You can then visualize the imported data.



07 - Clipping Images



- It is often desirable to clip the images to your area of interest. You can use the `clip()` function to mask out an image using a geometry.
- While in a Desktop software, clipping is desirable to remove unnecessary portion of a large image and save computation time. However, in Google Earth Engine, clipping can actually increase the computation time. As described in the [Earth Engine Coding Best Practices](#) guide, avoid clipping the images or do it at the end of your script.

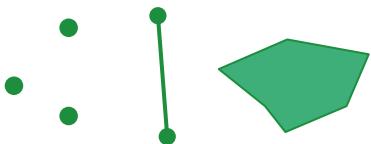


08 - Objects



Geometry

Points, lines, and polygons



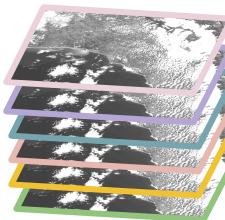
Feature

A Geometry + attributes

Geometry	Biome	NDVI
	"grassland"	0.32

Image

An arbitrary number of bands + band designations + metadata



- + B1 properties
- + B2 properties
- + B3 properties
- + B4 properties
- + B5 properties
- + B6 properties

+ metadata

Collection

A bag of a certain object type, i.e. a FeatureCollection

Geometry	Biome	NDVI
	"grassland"	0.32
	"savanna"	0.15

09 - Exporting Data



- Earth Engine allows for exporting both vector and raster data to be used in an external program. Vector data can be exported as a CSV or a Shapefile, while Rasters can be exported as GeoTIFF files. We will now export the Sentinel-2 Composite as a GeoTIFF file.
- Once you run this script, the *Tasks* tab will be highlighted. Switch to the tab and you will see the tasks waiting. Click *Run* next to each task to start the process.

The screenshot shows the Earth Engine interface with the **Inspector**, **Console**, and **Tasks** tabs. The **Tasks** tab is highlighted with an orange background. The console below it displays the message: "Use print(...) to write to this console." Below the console are two items: "Image (16 bands)" and "Image (3 bands)", each with a "JSON" link to its right. A red arrow points from the text "Use print(...)" in the console to the "Console" tab.

The screenshot shows the **Tasks** tab selected. The interface includes a header "Manage tasks." and a sub-header "Search or cancel multiple tasks in the Task Manager [🔗](#)". Below this is a section titled "UNSUBMITTED TASKS" containing two entries: "Athens_Composite_Raw" and "Athens_Composite_Visualized", each with a "RUN" button to its right. A red arrow points from the "RUN" button for the first task to the "RUN" button for the second task.

The screenshot shows a dialog box titled "Task: Initiate image export". It contains the following fields:

- Task name (no spaces) ***: Athens_Composite_Raw
- Coordinate Reference System (CRS)**: EPSG:3857
- Scale (m/px)**: 20
- DRIVE**: Drive folder earthengine
- CLOUD STORAGE**:
- EE ASSET**:
- Filename ***: athens_composite_raw
- File format ***: GEO_TIFF

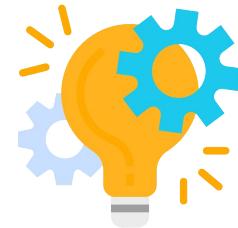
At the bottom are "CANCEL" and "RUN" buttons.

10 - Case Study: NO₂ concentration levels in Italy



Activity

Task: You need to retrieve, analyse, and visualise NO₂ concentration levels in Italy in 2019 and 2020.



In your groups, discuss the following:

- What datasets do you need?
- Briefly describe the steps taken to visualise NO₂ concentration levels in Italy in 2019 and 2020

Go to mentimeter.com

Type in your answer to the above questions

5 minutes

Tip: Sometimes, having an end in mind can help you with your workflow.



11 - Calculating Indices



- Spectral Indices are central to many aspects of remote sensing. Whether you are studying vegetation or tracking fires - you will need to compute a pixel-wise ratio of 2 or more bands.
- The most commonly used formula for calculating an index is the *Normalized Difference* between 2 bands. Earth Engine provides a helper function `normalizedDifference()` to help calculate normalized indices, such as Normalized Difference Vegetation Index (NDVI).
- For more complex formulae, you can also use the `expression()` function to describe the calculation.

$$\text{NDVI} = \frac{(\text{NIR} - \text{Red})}{(\text{NIR} + \text{Red})}$$

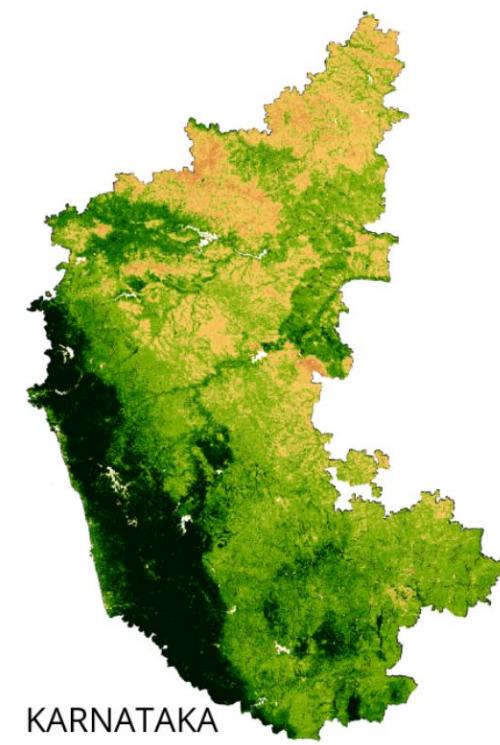
12 - Computations on ImageCollections



- So far we have learnt how to run computation on single images. If you want to apply some computation - such as calculating an index - to many images, you need to use `map()`.
- You first define a function that takes 1 image and returns the result of the computation on that image. Then you can `map()` that function over the `ImageCollection` which results in a new `ImageCollection` with the results of the computation.
- This is similar to a *for-loop* that you maybe familiar with - but using `map()` allows the computation to run in parallel. Learn more at [Mapping over an `ImageCollection`](#)



KARNATAKA
COMPOSITE



KARNATAKA
NDVI

NDVI computation on an `ImageCollection`

13 - Reducers



- Reducers are the way to aggregate data over time, space, bands, arrays and other data structure in Earth Engine.
- The ee.Reducer class specifies how data is aggregated.
- The reducers in this class can specify a simple statistic to use for the aggregation (e.g. Minimum, maximum, mean, median, standard deviation, etc.), or a more complex summary of the input data (e.g. histogram, linear regression, list).
- Reductions may occur over time (`imageCollection.reduce()`), space (`image.reduceRegion()`, `image.reduceNeighborhood()`), bands (`image.reduce()`), or the attribute space of a FeatureCollection (`featureCollection.reduceColumns()` or FeatureCollection methods that start with `aggregate_`).

13 - Reducers

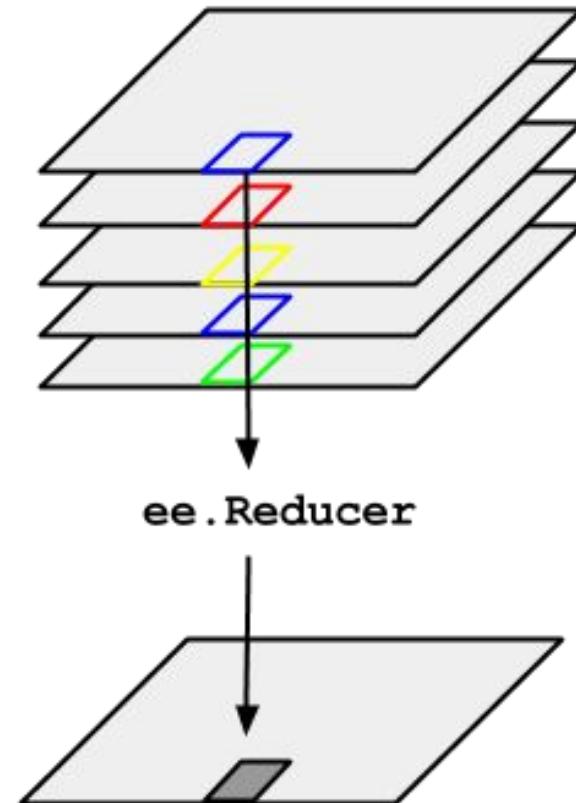


- Consider the example of needing to take the median over a time series of images represented by an ImageCollection.
- To reduce an ImageCollection, use `imageCollection.reduce()`. This reduces the collection of images to an individual image as illustrated.
- Specifically, the output is computed pixel-wise, such that each pixel in the output is composed of the median value of all the images in the collection at that location.

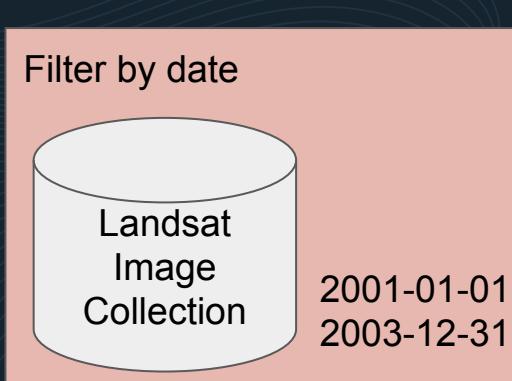
Tip: See the Docs tab in the Code Editor for a list of all the reducers currently available.

The screenshot shows the Google Earth Engine Code Editor interface. At the top, there are tabs for "Scripts", "Docs", and "Assets". The "Docs" tab is selected. Below the tabs, the "ee.Reducer" namespace is expanded, showing a list of methods:

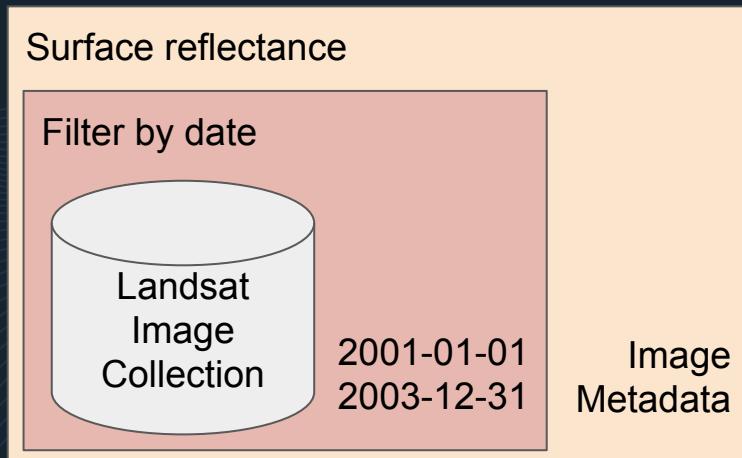
- ee.Reducer.allNonZero()
- ee.Reducer.and()
- ee.Reducer.anyNonZero()
- ee.Reducer.autoHistogram(maxBuckets, minBucketWidth, maxRaw, cum...)
- ee.Reducer.bitwiseAnd()
- ee.Reducer.bitwiseOr()
- ee.Reducer.centeredCovariance()
- ee.Reducer.circularMean()
- ee.Reducer.circularStddev()
- ee.Reducer.circularVariance()
- ee.Reducer.count()
- ee.Reducer.countDistinct()
- ee.Reducer.countDistinctNonNull()
- ee.Reducer.countEvery()
- ee.Reducer.countRuns()
- ee.Reducer.covariance()
- ee.Reducer.first()



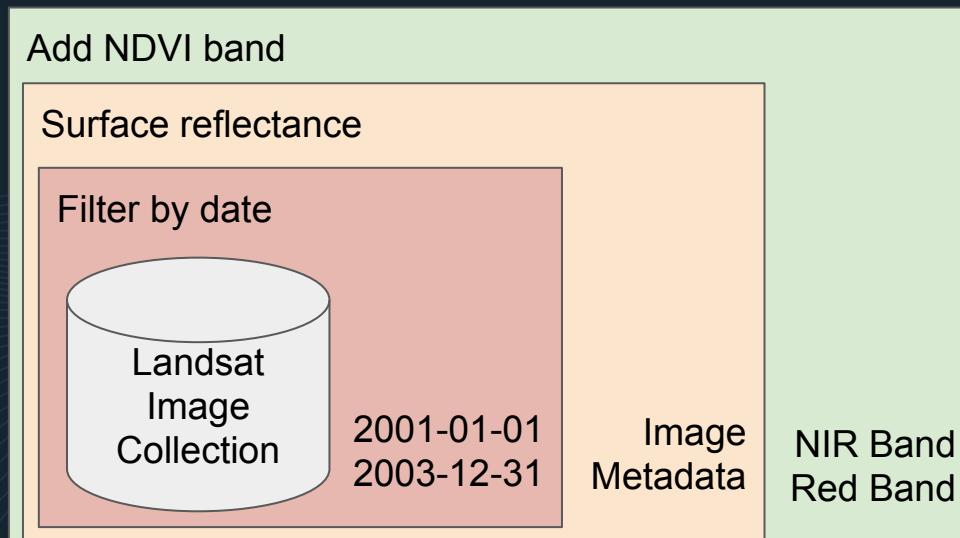
14 - A Computation Example



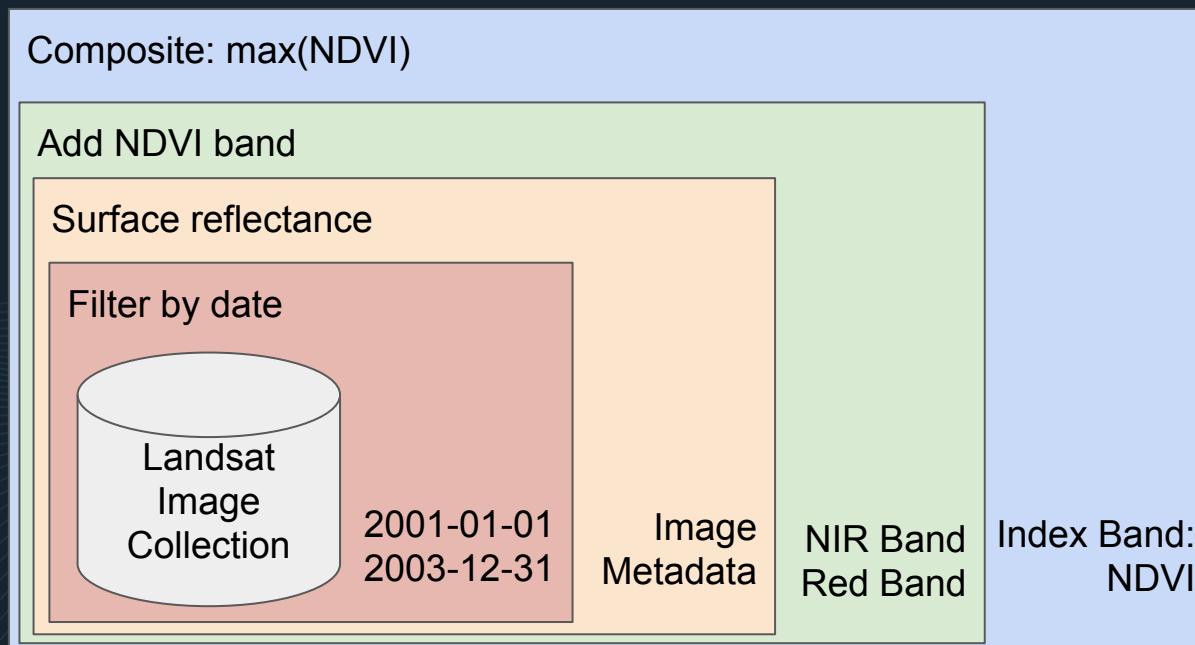
14 - A Computation Example



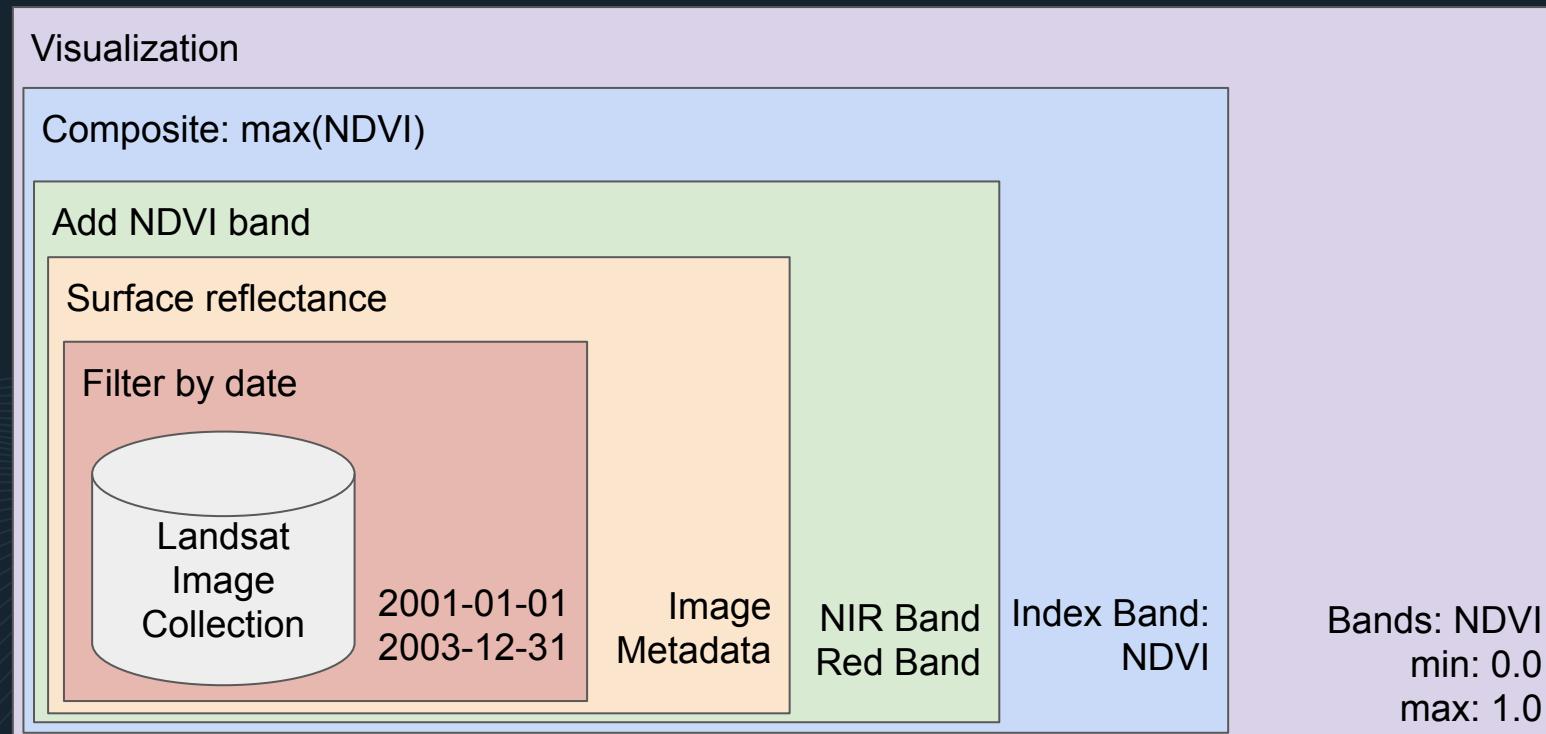
14 - A Computation Example



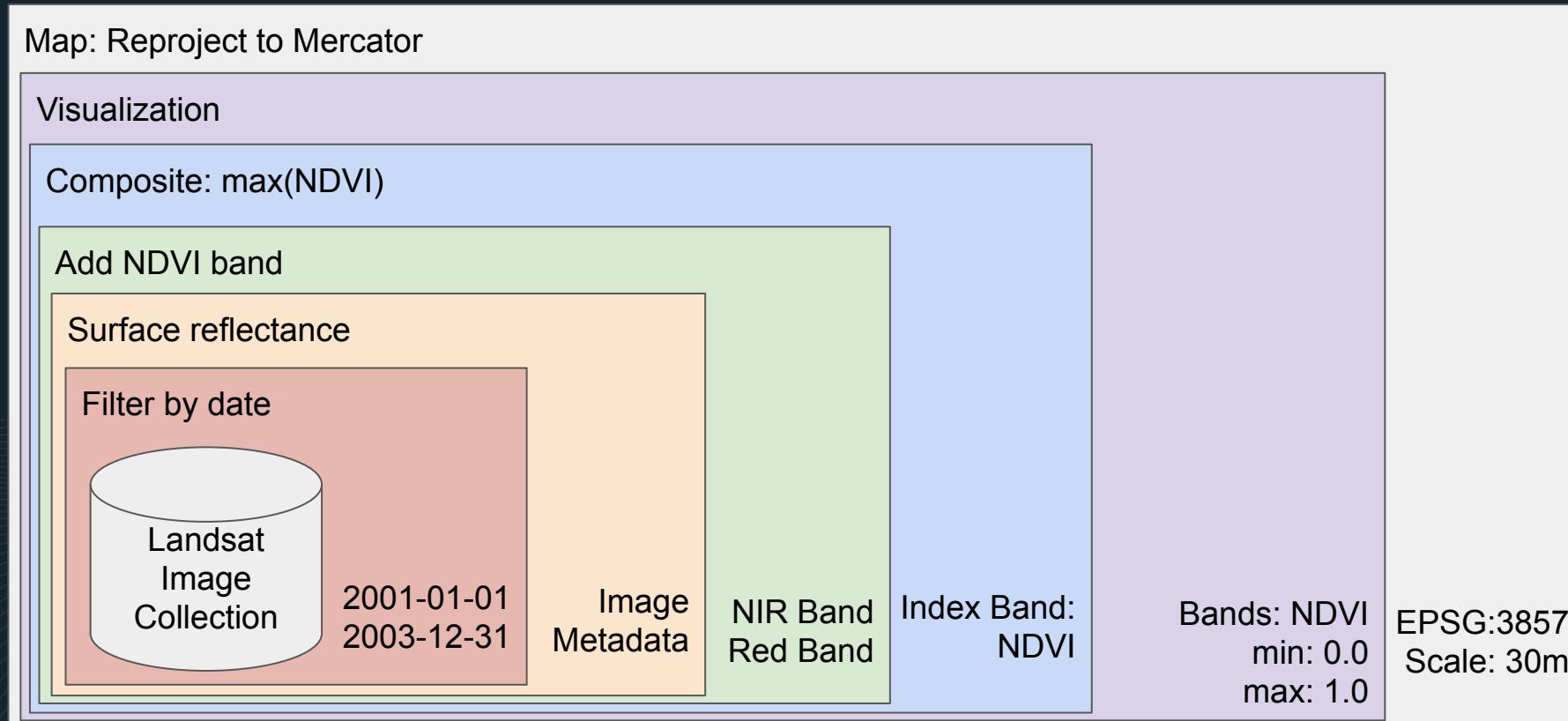
14 - A Computation Example



14 - A Computation Example



14 - A Computation Example

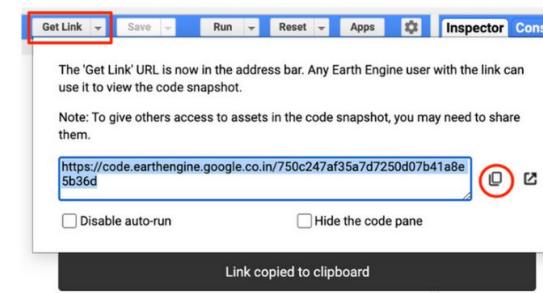


Sharing a Single Script

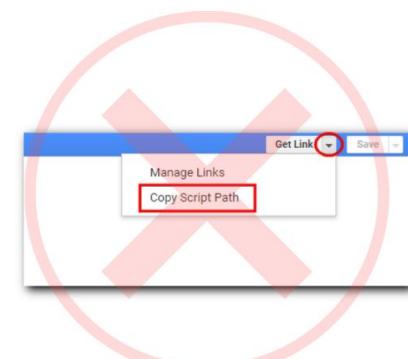


- To share your code from a single script, you need to use the **Get Link** button in the code editor.
- As you click the button, the contents of your code editor is captured and encoded into a URL.
- When you share this URL with someone, they will be able see same content as your code editor. This is a great way to send a snapshot of your code so others can reproduce your output.
- Remember that the script links are just snapshots, if you change your code after sending the link to someone, they will not see the updates.

Note: When trying to send someone a link, do NOT click the *Copy Script Path* button. Sending this path to someone will NOT give them access to your code. The script path only works only on public or shared repositories.



✓ **Correct**



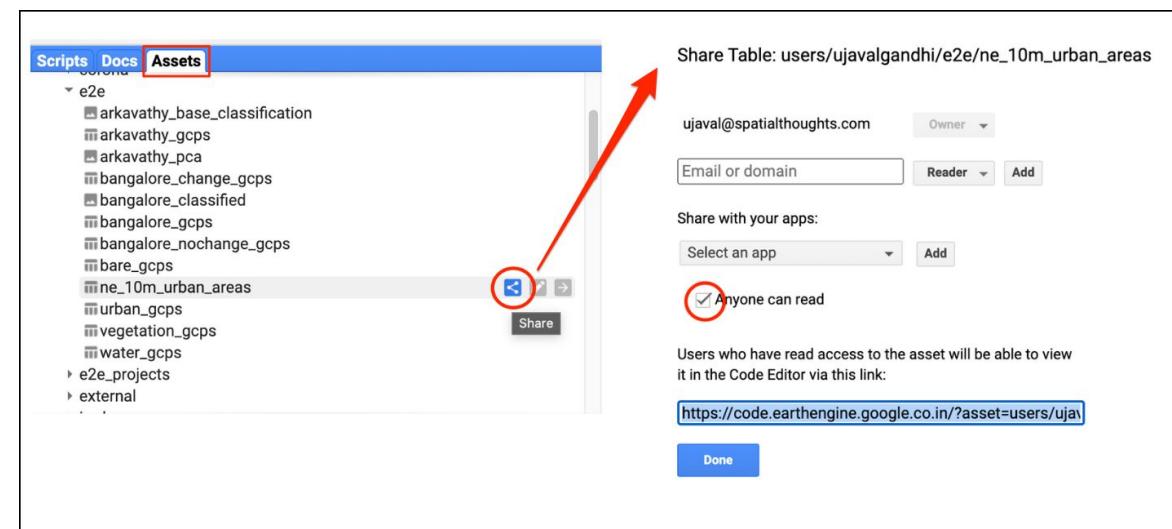
Wrong

Sharing Uploaded Assets



- While sharing the script using *Get Link*, you should also share any private **Assets** that you may have uploaded and are using in the script.
- You can share the asset with a specific email address, or check the *Anyone can read* box if you want anyone with the script link to be able to access it.
- Failing to do so will prevent others from running your script.

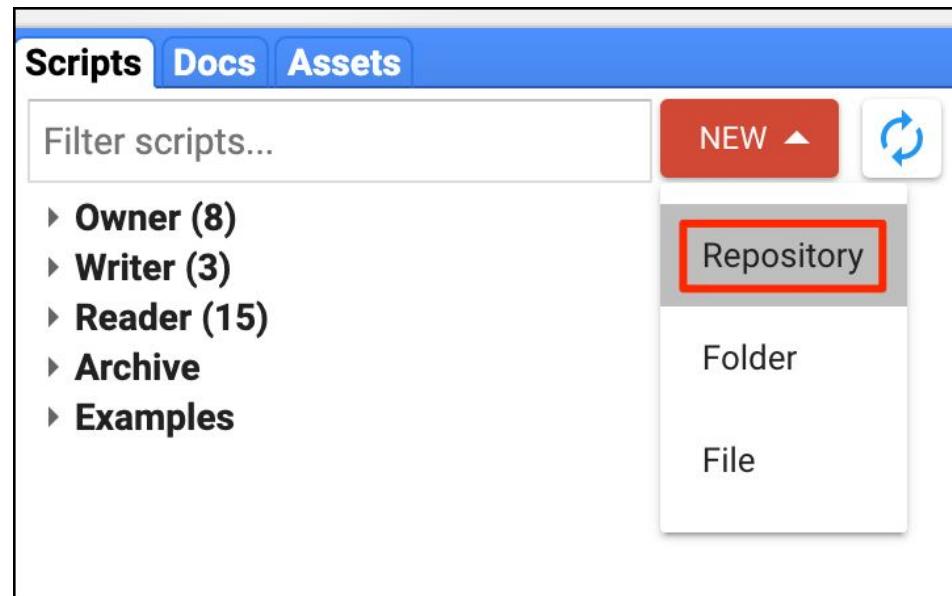
Learn more in the [Script links](#) section of the Google Earth Engine User Guide



Sharing Multiple Scripts



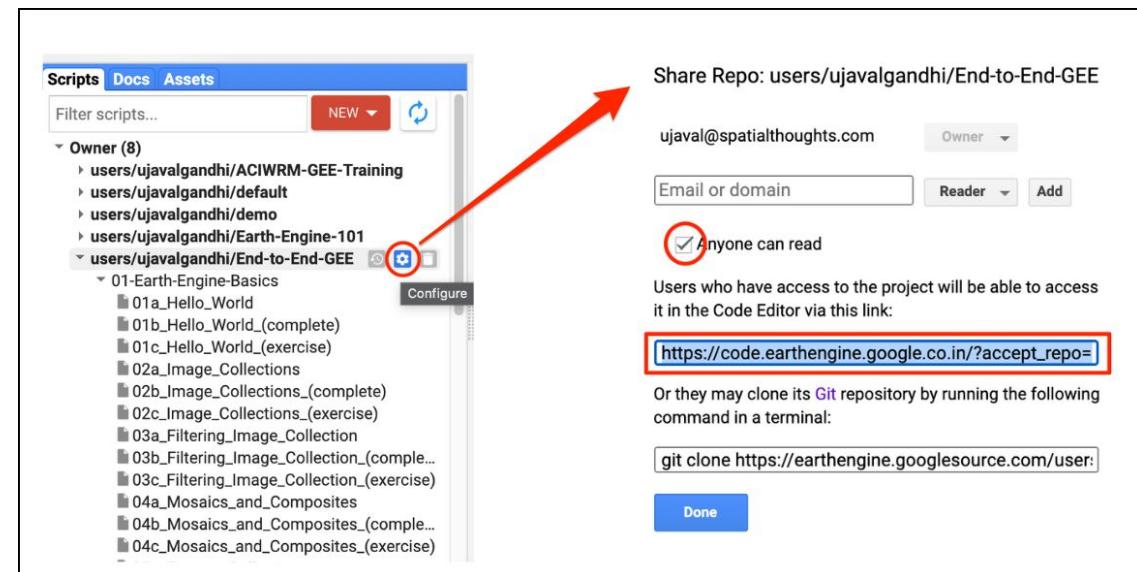
- If you want to share a collection of scripts with other users or your collaborators, the best way is to create a new Repository.



Sharing Multiple Scripts



- You can put multiple scripts within the repository and share the repository with other users. You can grant them **Reader** or **Writer** access so they can view/add/modify/delete scripts in that repository. If you want to make it readable by **Public**, you can check the *Anyone can read* option. You will see a URL in the form of https://code.earthengine.google.co.in/?accept_repo=..... When you share this URL with other users and they visit that link, your repository will be added to their Code Editor under the *Reader* or *Writer* folder depending on their access.



OUTLINE OF THIS WORKSHOP



1 Intro to GEE

The "Why?"
And the "What?"



2 Components of GEE

Intro to the Code Editor



3 Code Along Session

Hands-on using the Code Editor



4 Moving Forward

Available Resources

Resources - Band Combinations



Landsat 8

<https://gisgeography.com/landsat-8-bands-combinations/>



Sentinel 2

<https://gisgeography.com/sentinel-2-bands-combinations/>



Resources



Getting started with Earth Engine

This Get Started guide is intended as a quick way to start programming with Earth Engine JavaScript API.

<https://developers.google.com/earth-engine/guides/getstarted>

Google Earth Engine Data Catalog

Earth Engine's 60+ Petabyte public data archive includes more than 40 years of historical imagery and scientific datasets, updated and expanded daily

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

Getting Help



Google Earth Engine Developers Group

This group represents nearly a decade of support and open development on the Google Earth Engine platform. If you can't find the answer you are looking for in the archive, the members of this forum are friendly, prompt and helpful with GEE questions.

<https://groups.google.com/g/google-earth-engine-developers>

GIS Stack Exchange

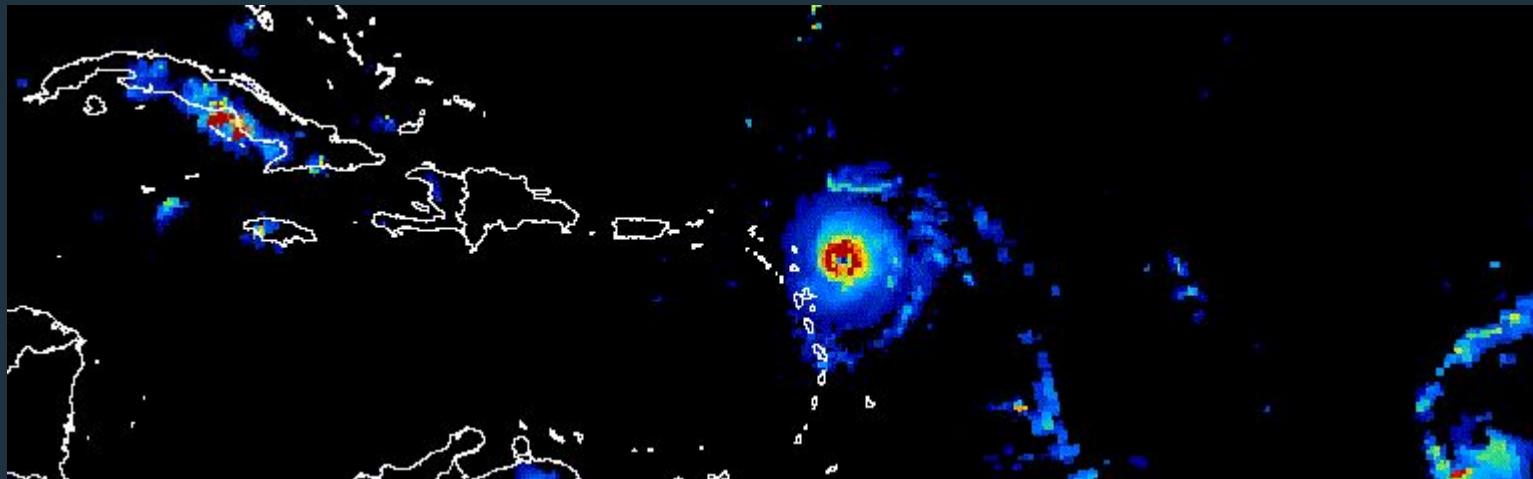
<https://gis.stackexchange.com/questions/tagged/google-earth-engine>

Resources - Visualisation



ImageCollection Visualisation

https://developers.google.com/earth-engine/guides/ic_visualization?hl=en



Animation showing a 3-day progression of Atlantic hurricanes in September 2017

References/ Resources



Google Earth Outreach Community Training Resources

<https://developers.google.com/earth-engine/tutorials/ttt>

Google Earth Engine 101

<https://storymaps.arcgis.com/stories/cdfc91d050634a5294ac897acc959d55>

End-to-End Google Earth Engine

<https://courses.spatialthoughts.com/end-to-end-gee.html>

Awesome Earth Engine

<https://github.com/giswqs/Awesome-GEE>

Open Geo Blog

<https://mygeoblog.com/category/google-earth-engine/>