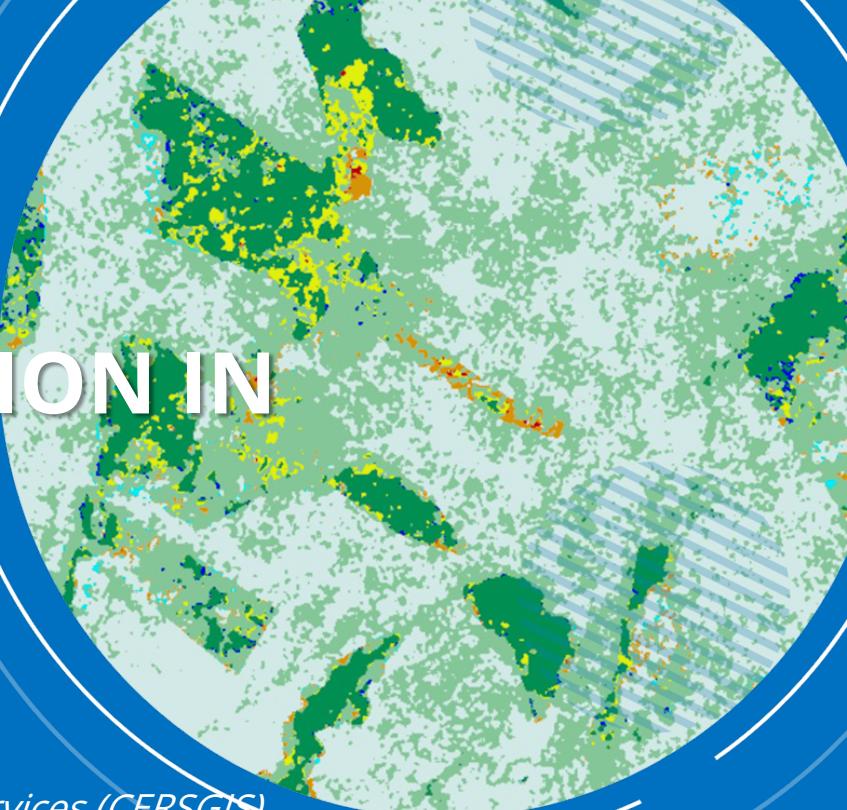


TIME SERIES AND USER INTERFACE CUSTOMISATION IN GOOGLE EARTH ENGINE

Ernest Opoku-Kwarteng eopoku-kwarteng@ug.edu.gh
Center for Remote Sensing and Geographic Information Services (CERSGIS)



Graphical User Interface

- ▶ The Google Earth Engine Code Editor is a Graphical User Interface (GUI) on its own
- ▶ While the base elements (map, task pane, etc.) cannot be altered new functionality can be added to the GUI
- ▶ Three different classes of elements can be added to the existing GUI:
 - ▶ Widgets
 - ▶ Panels
 - ▶ Events



Google Earth Engine

Widgets

- ▶ Widgets include buttons, checkboxes, sliders, textboxes and selection menus
- ▶ There are a variety of widgets for building or customizing UIs.
- ▶ Widgets can only be added to a panel once

ui.Label

- ▶ Labels are simply areas in which text is displayed. For example, the following code prints a label:

The screenshot shows the Google Earth Engine code editor interface. At the top, there's a toolbar with buttons for 'New Script *', 'Get Link', 'Save', 'Run', 'Reset', 'Apps', and a settings gear. Below the toolbar, the code editor window contains the following JavaScript code:

```
1
2
3 var label = ui.Label('MY FIRST GEE UI LABEL!');
4 print(label);
```

To the right of the code editor is a panel with tabs for 'Inspector', 'Console', and 'Tasks'. The 'Console' tab is selected, showing the output of the 'print' statement: 'MY FIRST GEE UI LABEL!'. Above the console output, there's a note: 'Use print(...) to write to this console.'



ui.Button

- ▶ A button is an interactive UI widget that can be clicked. You can specify a function to be called when a user clicks the button

The screenshot shows the Google Earth Engine code editor interface. The top bar includes buttons for 'Get Link', 'Save', 'Run', 'Reset', 'Apps', and settings. The main area is a code editor with the following JavaScript code:

```
1 // ***** BUTTON *****
2
3
4 var button = ui.Button({
5   label: 'Get Map Center',
6   onClick: function() {
7     print(Map.getCenter());
8   }
9 });
10 print(button);
11
```

To the right of the code editor is a panel with tabs for 'Inspector', 'Console' (which is selected), and 'Tasks'. The 'Console' tab contains the text 'Use print(...) to write to this console.' Below it is a text input field containing 'Get Map Center'.

ui.Slider

- ▶ A slider is a widget that lets a user adjust a slider to get a number within the slider range

The screenshot shows the Google Earth Engine code editor interface. The top bar includes buttons for 'Get Link', 'Save', 'Run', 'Reset', 'Apps', and settings. The main area is a code editor with the following JavaScript code:

```
var slider = ui.Slider();

slider.setValue(0.9); // Set a default value.
slider.onChange(function(value) {
  Map.layers().get(0).setOpacity(value);
});

Map.addLayer(ee.Image(255), {palette: 'blue'});
print(slider);
```

To the right of the code editor is a panel with tabs for 'Code Editor (JavaScript)' (which is selected), 'Inspector', and 'Tasks'. Below the code editor, the text 'The slider should look something like:' is followed by a horizontal slider control with a value of 1.

[Run Example](#)



Google Earth Engine

ui.DateSlider

- The DateSlider widget is like the Slider widget, but handles dates explicitly



Jun 11, 2018

[Jump to date](#)

ui.Select

- The select widget represents a drop-down menu of choices from which the user can choose one

```
geeu_select *  
Get Link Save Run Reset Apps  
1 // coordinates of places  
2 var places = {  
3   Accra: [-0.1807868398756315, 5.60330467698373],  
4   Kumasi: [-1.627210763408452, 6.694555759406957],  
5   Bolgatanga: [-0.8360296611868878, 10.62885517723995]  
6 };  
7 };  
8  
9 var select = ui.Select({  
10   items: Object.keys(places),  
11   onChange: function(key) {  
12     Map.setCenter(places[key][0], places[key][1], 10);  
13   }  
14 });  
15  
16 // Set a place holder.  
17 select.setPlaceholder('Choose a location...');  
18  
19 print(select);  
20  
21
```

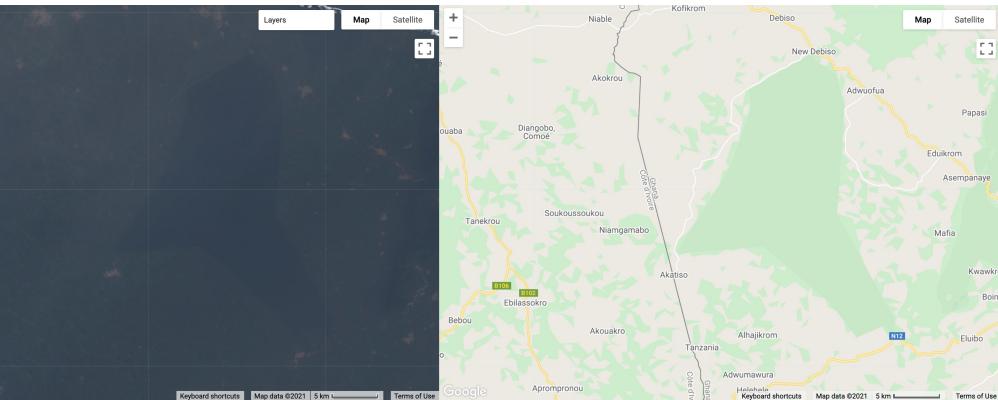
Inspector Console Tasks
Use print(...) to write to this console.
Bolgatanga

[Run Example](#)

ui.Map.Linker

- ▶ The linker is not a styleable widget. It is a behind-the-scenes utility that can be used to synchronize the movement of multiple ui.Map instances

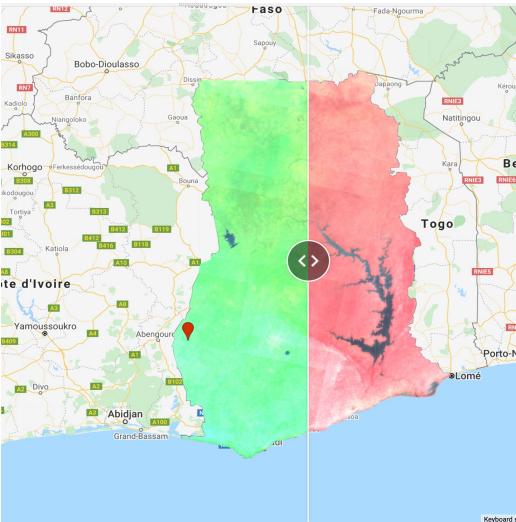
[Run Example](#)



ui.SplitPanel

- ▶ A ui.SplitPanel is useful for comparing things side-by-side. The advantage of a ui.SplitPanel over two ordinary panels is that a handle can be used to achieve a wipe transition between the panels in a ui.SplitPanel

[Run Example](#)





Google Earth Engine

Events

- ▶ Events are fired by user interaction with a widget or a programmatic change to a widget. To do something when the event occurs, register a callback function on the widget with either `onClick()` (for `ui.Map` or `ui.Button`) or `onChange()` (everything else).

The screenshot shows the Google Earth Engine Code Editor interface. On the left, the code editor displays a script titled "event *". The script uses the ee module to load SRTM and Landsat 8 images over a Ghana boundary. It then creates two dropdown menus: one for selecting between SRTM and Landsat 8 images, and another for selecting bands from the selected image. The "onChange" event is registered on both dropdowns to update the map layer. The right side of the interface shows the "Inspector", "Console", and "Tasks" panes. The "Console" pane contains two dropdown boxes labeled "Select a value...".

```
event *  
  Imports (1 entry) □  
  > var ghana: Table users/eopokukwarteng/ghana  
  1 // Load some images.  
  2 var srtm = ee.Image('CGIAR/SRTM90_V4').clip(ghana);  
  3 var landsat8 = ee.Image('LANDSAT/LC8_LIT_32DAY_TOA/20130407').clip(ghana);  
  4  
  5 // Make a drop-down menu of bands.  
  6 var bandSelect = ui.Select({  
  7   onChange: function(value){  
  8     // var layer = ui.Map.Layer(imageSelect.getValue().select(value));  
  9     // // Use set() instead of add() so the previous layer (if any) is overwritten.  
10     // Map.layers().set(0, layer);  
11   }  
12 };  
13 };  
14 );  
15  
16 // Make a drop down menu of images.  
17 var imageSelect = ui.Select({  
18   items: [  
19     {label: 'SRTM', value: srtm},  
20     {label: 'Landsat 8', value: landsat8}  
21   ],  
22   onChange: function(value) {  
23     // Asynchronously get the list of band names.  
24     value.bandNames().evaluate(function(bands) {  
25       // Display the bands of the selected image.  
26       bandSelect.items().reset(bands);  
27       // Set the first band to the selected band.  
28       bandSelect.setValue(bandSelect.items().get(0));  
29     });  
30   }  
31 };  
32 });  
33  
34 print(imageSelect);  
35 print(bandSelect);  
36
```

[Run in Code Editor](#)



Google Earth Engine

Panels and Layouts

ui.Panel

- ▶ A ui.Panel is an upper-level UI container in which to arrange widgets
- ▶ Each ui.Panel has a ui.Panel.Layout object that controls how its widgets are arranged on the screen

Layouts

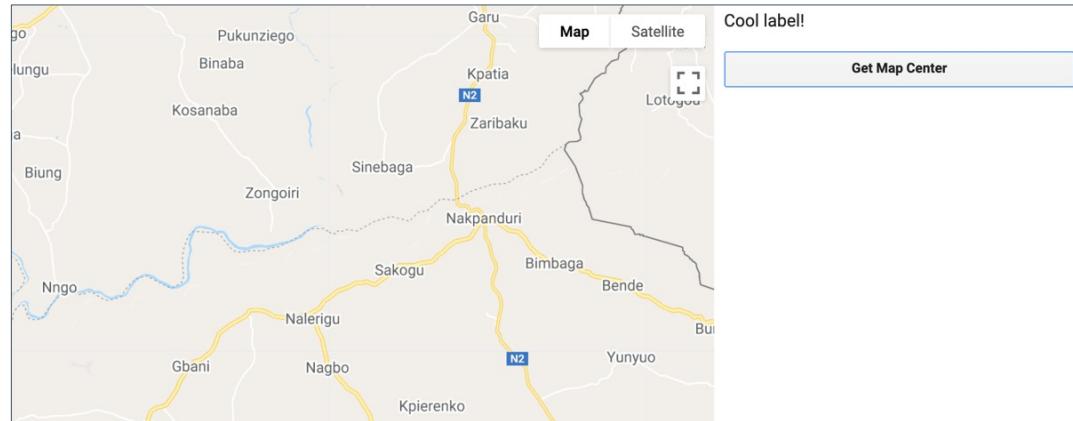
- ▶ Layouts control how widgets in a panel are arranged for display



Google Earth Engine

Panels and Layouts

```
// Create a panel with vertical flow layout.  
var panel = ui.Panel({  
  layout: ui.Panel.Layout.flow('vertical'),  
  style: {width: '300px'}  
});  
  
panel.add(ui.Label('Cool label!'))  
  
// Add a bunch of buttons.  
  
var button = ui.Button({  
  label: 'Get Map Center',  
  style: {stretch: 'horizontal'},  
  onClick: function() {  
    print(Map.getCenter());  
  }  
});  
  
panel.add(button);  
  
// ui.root.clear();  
ui.root.add(panel);
```



[Run in Code Editor](#)

Time-Series Charts

- ▶ Earth Engine API supports charting functions based on the Google Chart API
- ▶ Use the `ui.Chart.image.series()` function to create a time-series chart.

We will apply all the knowledge in the following ;

- ▶ filter
- ▶ map
- ▶ reduce
- ▶ cloud-masking

Exercise

- ▶ Create a chart of average NDVI values for a given farm over 1 year.

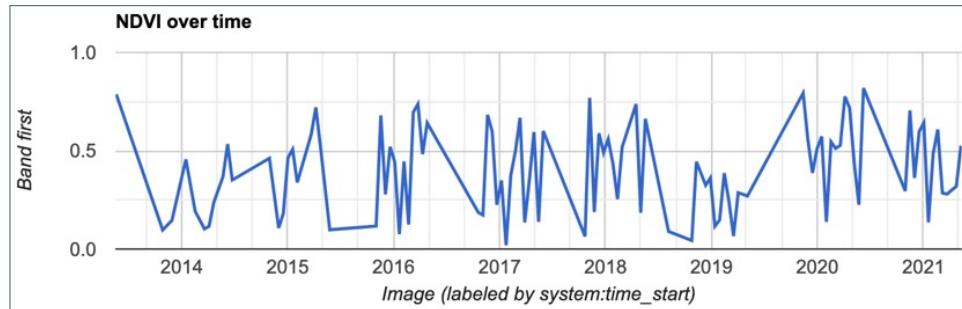


Google Earth Engine

Basic Time Series Analysis

```
// Import the Landsat 8 TOA image collection.  
var l8 = ee.ImageCollection('LANDSAT/LC08/C01/T1_TOA');  
  
// Map a function over the Landsat 8 TOA collection to add an NDVI band.  
var withNDVI = l8.map(function(image) {  
  var ndvi = image.normalizedDifference(['B5', 'B4']).rename('NDVI');  
  return image.addBands(ndvi);  
})  
  
// Create a chart.  
var chart = ui.Chart.image.series({  
  imageCollection: withNDVI.select('NDVI'),  
  region: roi,  
  reducer: ee.Reducer.first(),  
  scale: 30  
}).setOptions({title: 'NDVI over time'});  
  
// Display the chart in the console.  
print(chart);
```

[Run in Code Editor](#)



Creating an Interactive Time Series App



Google Earth Engine

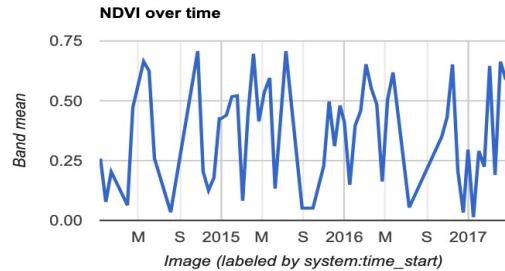
Layers

Map

Satellite



lon: -3.11 lat: 6.49

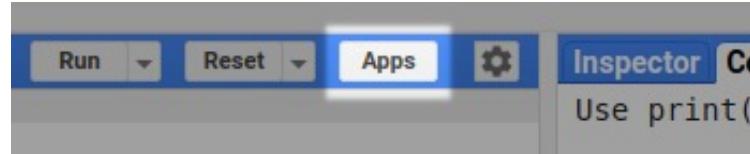




Google Earth Engine

Publishing your App - 1

- ▶ To publish an App from the Code Editor:
 - ▶ First load the script that you want to make into an App
 - ▶ Open the App Management panel, which you can access by clicking the **Apps** button above the script section in the Code Editor
 - ▶ Click on the **NEW APP** button
 - ▶ In the dialog, choose an App name, select a Google Cloud Project, and specify the location of the App's source code



Manage Apps

VIEW GALLERY NEW APP

App Name (click to launch)	ID (click to update app)	Delete



Google Earth Engine

Publishing your App - 2

- ▶ When all fields are filled out and validated, the PUBLISH button will be enabled
- ▶ Click the button to complete publishing

Publish New App

Owner
users/eopokukwarteng

App Name ⓘ
Timeseries App
Your App ID will be timeseries-app [Edit](#)

URL: <https://eopokukwarteng.users.earthengine.app/view/timeseries-app>

Google Cloud Project ⓘ
ee-eopokukwarteng [CHANGE](#)

Access Restriction ⓘ
 Restrict access to this app

Public Gallery
 Feature this app in your [Public Apps Gallery](#)

[Set Thumbnail \(Optional\)](#) [Description \(Optional\)](#)

 This app is powered by Google Earth Engine.

Source Code ⓘ
 Current contents of editor
 Repository script path

When the app is published, it's public and anyone can view it. The published source code will be publicly readable. All assets must also be shared publicly or with the app to display properly. See <https://developers.google.com/earth-engine/apps> for more information about publishing apps.

[CANCEL](#) [PUBLISH](#)



Google Earth Engine

Managing your Apps

- ▶ To manage an App from the Code Editor:
 - ▶ Open the App Management panel by clicking the Apps button above the script section in the Code Editor.
 - ▶ Update the App's configuration
 - ▶ Or delete the App

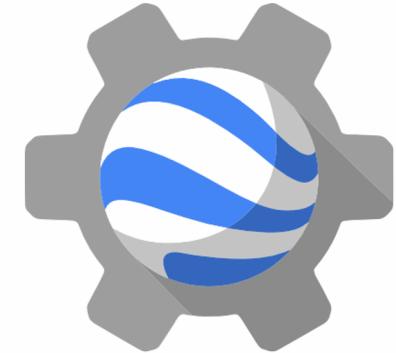
Manage Apps

VIEW GALLERY NEW APP

App Name (click to launch)	ID (click to update app)	Delete
ernest	users/eopokukwarteng/bbb	
Land degradation	users/eopokukwarteng/land-degradation	
locust monitoring	users/eopokukwarteng/locust-monitoring	
uiapp	users/eopokukwarteng/uiapp	
visualizationdemo	users/eopokukwarteng/visualizationdemo	
water	users/eopokukwarteng/water	
waterdetection	users/eopokukwarteng/waterdetection	

CLOSE

Google Earth Engine



Thank You



Accra, Ghana
September, 2021

