

Physical Landscapes and the Geosphere (ACNA 4007)

Google Earth Engine and Exploring Water Balance

Activity Outline:

In this activity we will be using Google Earth Engine to examine flooding, changes in hydrological budget for lakes, river adjustment, seasonality, and how we can map or interpret this using simple JavaScript commands. You will have a go at running a simple code in Google Earth Engine and will then apply this code to a site of your choosing. By the end of this lab, you will be more familiar with:

- Google Earth Engine
- JavaScript code editor
- Visualising data
- Inspecting Data
- Earth Surface Processes

Background

Globally, the hydrological cycle is experiencing significant changes. It is important to be able to describe, quantify and communicate these changes. This lab is all about having a go, making mistakes, and trying out various scripts and tools to learn more about fluvial systems in Wales and beyond. We will be sharing our findings, cool visuals and graphs here on this Padlet page: [Physical Landscapes and the Geosphere \(padlet.com\)](https://padlet.com/Physical_Landscapes_and_the_Geosphere).

We will be using Google Earth Engine for this short workshop. You can sign up for GEE here: <https://signup.earthengine.google.com/>. Google Earth Engine is a platform for scientific analysis and visualization of geospatial datasets, for academic, non-profit, business and government users. Earth Engine hosts satellite imagery and stores it in a public data archive that includes historical earth images going back more than forty years. The images, ingested on a daily basis, are then made available for global-scale data mining. Earth Engine also provides APIs and other tools to enable the analysis of large datasets. For more information, please see this link: [FAQ – Google Earth Engine](#).

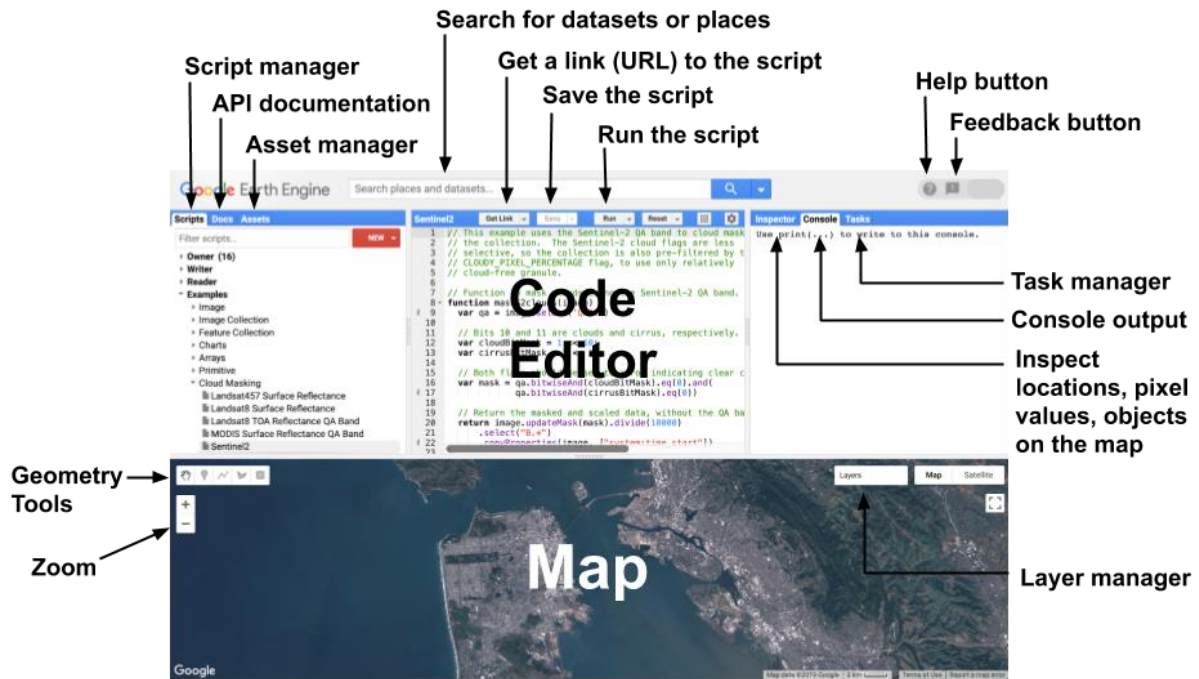


Figure 1. Image of the Google Earth Engine interface

Exercise 1: Global Surface Water Animation

Link: [JRC Global Surface Water Animation \(earthengine.app\)](https://earthengine.app)

This app animates 35 years of surface water changes at an annual time step. It uses the [Global Surface Water Explorer \(global-surface-water.appspot.com\)](https://global-surface-water.appspot.com). Here you can create a simple (and visually appealing) GIF highlighting change in surface water from 1984-2018.

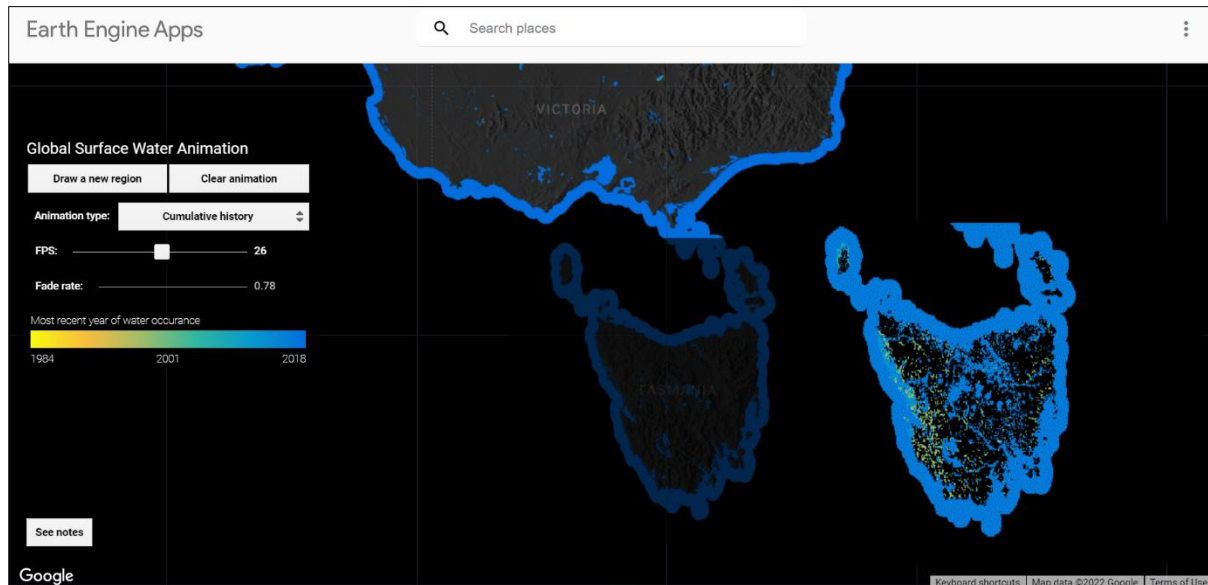


Figure 2. Surface Water Animation Example

- 1) Zoom to one of the lakes mentioned in our lecture today (Caspian Sea, Lake Superior, Lake Victoria, Lake Huron, Lake Michigan, Lake Tanganyika, Lake Baikal, Great Bear Lake)
- 2) Choose a location/state and draw a new region around the location.
- 3) Create a GIF
- 4) Right click on your GIF image to save it
- 5) Do you notice any trends for the last 35 years in your area?
- 6) Upload your GIF (and a brief description) to the Padlet.
- 7) Now choose a new location of your choice. Repeat the steps 2-6. What are the key differences between your areas? Why could that be?

Exercise 2: Water Occurrence Change Intensity

In Exercise 1 we looked at a basic surface water animation. This animation showed us surface water change through **annual timesteps**. Next, we will explore water occurrence change intensity (e.g., the severity of change). You will explore changes in hydrology for **5 sites** from those provided below. Some of these sites have lakes AND rivers. Using the output, you can think about the key controlling factors for these changes.

Link: [Water Occurrence Change Intensity | Google Earth Engine | Google Developers](#)

1. Click on the Link above to direct yourself to a tutorial page. The page breaks down the code that you will be using in this exercise. Note: There is a full script for this activity at the end of this tutorial handout.
 2. Work your way through the tutorial steps to produce a histogram of change for **five** of the example locations (you will need to uncomment your script to set the map location).
- (-90.162, 29.8597, 10); New Orleans, USA
 - (-114.9774, 31.9254, 10); Mouth of the Colorado River, Mexico
 - (-111.1871, 37.0963, 11); Lake Powell, USA
 - (149.412, -35.0789, 11); Lake George, Australia
 - (105.26, 11.2134, 9); Mekong River Basin, SouthEast Asia
 - (90.6743, 22.7382, 10); Meghna River, Bangladesh
 - (81.2714, 16.5079, 11); Godavari River Basin Irrigation Project, India
 - (14.7035, 52.0985, 12); River Oder, Germany & Poland
 - (-59.1696, -33.8111, 9); Buenos Aires, Argentina
 - (-74.4557, -8.4289, 11); Ucayali River, Peru - Example shown in Figure 3.

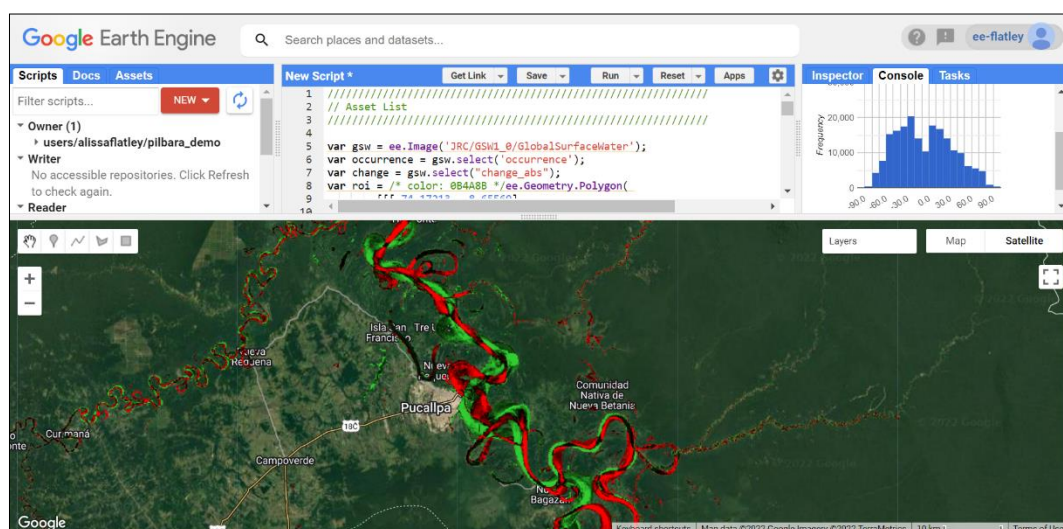


Figure 3. Water Occurrence Change Intensity example for Ucayali River, Peru.

3. Next, it would be fun to look at the intensity of change for your lake location(s) chosen in Activity 1. Create a new histogram of change for the particular location within your Surface Water Change GIF. This location will be one of the top largest lakes identified in the first activity. Describe any key observations you have.
4. Zoom out and look at the Water Occurrence Change Intensity at a global level. Are there any areas experiencing greater change than others? Can you think of reasons why this might be happening?

Exercise 3: Exploring water classes

This activity introduces the idea of different water classes. Not all rivers are perennial, many are seasonal, ephemeral etc. This script can help identify water classes and will summarize the relative size of the water transition classes.

Link: [Water Class Transition](#) | [Google Earth Engine](#) | [Google Developers](#)

1. For your chosen location, create a map layer for visualizing water transition,
2. Create a chart that summarizes the area by transition class. Add a screenshot or chart output to the Padlet for your area.

Exercise 4: Exploring Seasonality

So far, we have explored annual surface water changes and intensity. This activity uses Synthetic Aperture Radar (SAR) from Sentinel 1 to filter, classify water areas and create a time-series for your chosen location to visualise and plot inundation. This script is structured to make it easy for the user to select different regions or time periods. I do not expect you all to get through all of the exercises but I wanted to demonstrate some useful applications of GEE.

Link: [Google Earth Engine: SAR Water Body Classification \(mbonnema.github.io\)](#)

1. *This script can classify water from SAR imagery and create an interactive time series of water pixel count. Load the script here: [Link 2530f17708e876ed73cb7945fe2a1b37 - Earth Engine Code Editor \(google.com\)](#).*
2. *Work your way through the example provided.*
3. *Can you apply this script to your location? What are your key observations? What are the key seasonal variations?*
4. *Add a screenshot and a brief description to the Padlet.*

Appendices

Exercise 2: Water Occurrence Change Intensity (full script)

```
////////////////////////////////////
// Asset List
////////////////////////////////////

var gsw = ee.Image('JRC/GSW1_0/GlobalSurfaceWater');
var occurrence = gsw.select('occurrence');
var change = gsw.select("change_abs");
var roi = /* color: 0B4A8B */ee.Geometry.Polygon(
  [[[-74.17213, -8.65569],
    [-74.17419, -8.39222],
    [-74.38362, -8.36980],
    [-74.43031, -8.61293]]]];

////////////////////////////////////
// Constants
////////////////////////////////////

var VIS_OCCURRENCE = {
  min:0,
  max:100,
  palette: ['red', 'blue']
};
var VIS_CHANGE = {
  min:-50,
  max:50,
  palette: ['red', 'black', 'limegreen']
};
var VIS_WATER_MASK = {
  palette: ['white', 'black']
};

////////////////////////////////////
// Calculations
////////////////////////////////////

// Create a water mask layer, and set the image mask so that non-water areas are transparent.
var water_mask = occurrence.gt(90).mask(1);

// Generate a histogram object and print it to the console tab.
var histogram = ui.Chart.image.histogram({
  image: change,
  region: roi,
  scale: 30,
  minBucketWidth: 10
});
histogram.setOptions({
  title: 'Histogram of surface water change intensity.'
});
print(histogram);
```

```

////////////////////////////////////
// Initialize Map Location
////////////////////////////////////

// Uncomment one of the following statements to center the map on
// a particular location.
// Map.setCenter(-90.162, 29.8597, 10); // New Orleans, USA
// Map.setCenter(-114.9774, 31.9254, 10); // Mouth of the Colorado River, Mexico
// Map.setCenter(-111.1871, 37.0963, 11); // Lake Powell, USA
// Map.setCenter(149.412, -35.0789, 11); // Lake George, Australia
// Map.setCenter(105.26, 11.2134, 9); // Mekong River Basin, SouthEast Asia
// Map.setCenter(90.6743, 22.7382, 10); // Meghna River, Bangladesh
// Map.setCenter(81.2714, 16.5079, 11); // Godavari River Basin Irrigation Project, India
// Map.setCenter(14.7035, 52.0985, 12); // River Oder, Germany & Poland
// Map.setCenter(-59.1696, -33.8111, 9); // Buenos Aires, Argentina\
Map.setCenter(-74.4557, -8.4289, 11); // Ucayali River, Peru

////////////////////////////////////
// Map Layers
////////////////////////////////////

Map.addLayer({
  eeObject: water_mask,
  visParams: VIS_WATER_MASK,
  name: '90% occurrence water mask',
  shown: false
});
Map.addLayer({
  eeObject: occurrence.updateMask(occurrence.divide(100)),
  name: "Water Occurrence (1984-2015)",
  visParams: VIS_OCCURRENCE,
  shown: false
});
Map.addLayer({
  eeObject: change,
  visParams: VIS_CHANGE,
  name: 'occurrence change intensity'
});

```

Exercise 3: Water Class (full script)

```

////////////////////////////////////

// Asset List

////////////////////////////////////

```

```

var gsw = ee.Image('JRC/GSW1_0/GlobalSurfaceWater');

var occurrence = gsw.select('occurrence');

var change = gsw.select("change_abs");

var transition = gsw.select('transition');

var roi = ee.Geometry.Polygon(
  [[[105.531921, 10.412183],
    [105.652770, 10.285193],
    [105.949401, 10.520218],
    [105.809326, 10.666006]]]);

////////////////////////////////////

// Constants

////////////////////////////////////

var VIS_OCCURRENCE = {
  min: 0,
  max: 100,
  palette: ['red', 'blue']
};

var VIS_CHANGE = {
  min: -50,
  max: 50,
  palette: ['red', 'black', 'limegreen']
};

var VIS_WATER_MASK = {
  palette: ['white', 'black']
};

////////////////////////////////////

// Helper functions

////////////////////////////////////

```

```
// Create a feature for a transition class that includes the area covered.
function createFeature(transition_class_stats) {
  transition_class_stats = ee.Dictionary(transition_class_stats);
  var class_number = transition_class_stats.get('transition_class_value');
  var result = {
    transition_class_number: class_number,
    transition_class_name: lookup_names.get(class_number),
    transition_class_palette: lookup_palette.get(class_number),
    area_m2: transition_class_stats.get('sum')
  };
  return ee.Feature(null, result); // Creates a feature without a geometry.
}
```

```
// Create a JSON dictionary that defines piechart colors based on the
// transition class palette.
// https://developers.google.com/chart/interactive/docs/gallery/piechart
function createPieChartSliceDictionary(fc) {
  return ee.List(fc.aggregate_array("transition_class_palette"))
    .map(function(p) { return {'color': p}; }).getInfo();
}
```

```
// Convert a number to a string. Used for constructing dictionary key lists
// from computed number objects.
function numToString(num) {
  return ee.Number(num).format();
}
```

```
////////////////////////////////////
// Calculations
////////////////////////////////////
```

```

// Create a dictionary for looking up names of transition classes.
var lookup_names = ee.Dictionary.fromLists(
  ee.List(gsw.get('transition_class_values')).map(numToString),
  gsw.get('transition_class_names')
);

// Create a dictionary for looking up colors of transition classes.
var lookup_palette = ee.Dictionary.fromLists(
  ee.List(gsw.get('transition_class_values')).map(numToString),
  gsw.get('transition_class_palette')
);

// Create a water mask layer, and set the image mask so that non-water areas
// are transparent.
var water_mask = occurrence.gt(90).mask(1);

// Generate a histogram object and print it to the console tab.
var histogram = ui.Chart.image.histogram({
  image: change,
  region: roi,
  scale: 30,
  minBucketWidth: 10
});
histogram.setOptions({
  title: 'Histogram of surface water change intensity.'
});
print(histogram);

// Summarize transition classes in a region of interest.
var area_image_with_transition_class = ee.Image.pixelArea().addBands(transition);
var reduction_results = area_image_with_transition_class.reduceRegion({
  reducer: ee.Reducer.sum().group({

```

```

    groupField: 1,
    groupName: 'transition_class_value',
  }},
  geometry: roi,
  scale: 30,
  bestEffort: true,
});
print('reduction_results', reduction_results);

var roi_stats = ee.List(reduction_results.get('groups'));

var transition_fc = ee.FeatureCollection(roi_stats.map(createFeature));
print('transition_fc', transition_fc);

// Add a summary chart.
var transition_summary_chart = ui.Chart.feature.byFeature({
  features: transition_fc,
  xProperty: 'transition_class_name',
  yProperties: ['area_m2', 'transition_class_number']
})
.setChartType('PieChart')
.setOptions({
  title: 'Summary of transition class areas',
  slices: createPieChartSliceDictionary(transition_fc),
  sliceVisibilityThreshold: 0 // Don't group small slices.
});
print(transition_summary_chart);

////////////////////////////////////
// Initialize Map Location
////////////////////////////////////

```

```
// Uncomment one of the following statements to center the map on
// a particular location.

// Map.setCenter(-90.162, 29.8597, 10); // New Orleans, USA
// Map.setCenter(-114.9774, 31.9254, 10); // Mouth of the Colorado River, Mexico
// Map.setCenter(-111.1871, 37.0963, 11); // Lake Powell, USA
// Map.setCenter(149.412, -35.0789, 11); // Lake George, Australia
Map.setCenter(105.26, 11.2134, 9); // Mekong River Basin, SouthEast Asia
// Map.setCenter(90.6743, 22.7382, 10); // Meghna River, Bangladesh
// Map.setCenter(81.2714, 16.5079, 11); // Godavari River Basin Irrigation Project, India
// Map.setCenter(14.7035, 52.0985, 12); // River Oder, Germany & Poland
// Map.setCenter(-59.1696, -33.8111, 9); // Buenos Aires, Argentina
// Map.setCenter(-74.4557, -8.4289, 11); // Ucayali River, Peru
```

```
////////////////////////////////////
```

```
// Map Layers
```

```
////////////////////////////////////
```

```
Map.addLayer({
  eeObject: water_mask,
  visParams: VIS_WATER_MASK,
  name: '90% occurrence water mask',
  shown: false
});

Map.addLayer({
  eeObject: occurrence.updateMask(occurrence.divide(100)),
  name: "Water Occurrence (1984-2015)",
  visParams: VIS_OCCURRENCE,
  shown: false
});

Map.addLayer({
```

```

eeObject: change,
visParams: VIS_CHANGE,
name: 'occurrence change intensity',
shown: false
});
Map.addLayer({
  eeObject: transition,
  name: 'Transition classes (1984-2015)',
});

```

Exercise 4: Exploring Seasonality (full script)

```

// Load region defined by polygon and add it to the map
var roi = ee.Geometry.Polygon(
  [[[106.5311175986817, 14.88401360239504],
    [106.70144574301412, 14.88567468158789],
    [106.68221660149186, 14.996475939738637],
    [106.67432540134314, 15.050880175819685],
    [106.61113258648004, 15.046083385875264],
    [106.53764487153217, 15.035961287202868],
    [106.52682423644103, 14.968444128060433]]]);
Map.addLayer(roi, {}, 'ROI')
Map.centerObject(roi, 8)

//Load Sentinel-1 SAR collection and filter according to data collection type
var S1 = ee.ImageCollection('COPERNICUS/S1_GRD')
  .filterBounds(roi)
  .filterDate('2017-01-01','2019-04-01')
  .filter(ee.Filter.listContains('transmitterReceiverPolarisation', 'VV'))

```

```

//Add first image to map to get an idea of what a SAR image looks like
Map.addLayer(S1.first(),{bands: 'VV',min: -18, max: 0}, 'SAR image')

// Filter speckle noise
var filterSpeckles = function(img) {
  var vv = img.select('VV') //select the VV polarization band
  var vv_smoothed = vv.focal_median(100,'circle','meters').rename('VV_Filtered') //Apply a
  focal median filter
  return img.addBands(vv_smoothed) // Add filtered VV band to original image
}

// Map speckle noise filter across collection. Result is same collection, with smoothed VV
band added to each image
S1 = S1.map(filterSpeckles)

//Add speckle filtered image to map to sompare with raw SAR image
Map.addLayer(S1.first(),{bands: 'VV_Filtered',min: -18, max: 0}, 'Filtered SAR image')

//Here we are using -16. This is only an approximation and will result in some errors. Try
adjusting the
var classifyWater = function(img) {
  var vv = img.select('VV_Filtered')
  var water = vv.lt(-16).rename('Water') //Identify all pixels below threshold and set them
  equal to 1. All other pixels set to 0
  water = water.updateMask(water) //Remove all pixels equal to 0
  return img.addBands(water) //Return image with added classified water band
}

//Map classification across sentinel-1 collection and print to console to inspect

```

```
S1 = S1.map(classifyWater)
```

```
print(S1)
```

```
//Make time series of water pixels within region
```

```
var ClassChart = ui.Chart.image.series({
```

```
  imageCollection: S1.select('Water'),
```

```
  region: roi,
```

```
  reducer: ee.Reducer.sum(),
```

```
  scale: 100,
```

```
})
```

```
  .setOptions({
```

```
    title: 'Inundated Pixels',
```

```
    hAxis: {'title': 'Date'},
```

```
    vAxis: {'title': 'Number of Inundated Pixels'},
```

```
    lineWidth: 2
```

```
  })
```

```
//Set the position of the chart and add it to the map
```

```
ClassChart.style().set({
```

```
  position: 'bottom-right',
```

```
  width: '500px',
```

```
  height: '300px'
```

```
});
```

```
Map.add(ClassChart)
```

```
// Create a label on the map.
```

```
var label = ui.Label('Click a point on the chart to show the image for that date.');
```

```
Map.add(label);
```

//Create callback function that adds image to the map corresponding with clicked data point on chart

```
ClassChart.onClick(function(xValue, yValue, seriesName) {
```

```
    if (!xValue) return; // Selection was cleared.
```

```
    // Show the image for the clicked date.
```

```
    var equalDate = ee.Filter.equals('system:time_start', xValue);
```

```
    //Find image corresponding with clicked data and clip water classification to roi
```

```
    var classification = ee.Image(S1.filter(equalDate).first()).clip(roi).select('Water');
```

```
    var SARImage = ee.Image(S1.filter(equalDate).first());
```

```
    //Make map layer based on SAR image, reset the map layers, and add this new layer
```

```
    var S1Layer = ui.Map.Layer(SARImage, {
```

```
        bands: ['VV'],
```

```
        max: 0,
```

```
        min: -20
```

```
    });
```

```
    Map.layers().reset([S1Layer]);
```

```
    var visParams = {
```

```
        min: 0,
```

```
        max: 1,
```

```
        palette: ['#FFFFFF', '#0000FF']
```

```
    }
```

```
    //Add water classification on top of SAR image
```

```
    Map.addLayer(classification, visParams, 'Water')
```

```
    // Show a label with the date on the map.
```

```
    label.setValue((new Date(xValue)).toUTCString());
```

```
});
```