Article

← FME Desktop (/S/Topic/0TO4Q000000QL9uWAG/Fme-...

Advanced JSON Reading

Aug 4, 2022 • Knowledge

Product Type
FME Desktop

FME Version
2022.0

Tutorial: Tutorial: Getting Started with JSON (/s/article/tutorial-getting-started-with-json) | **Previous:** Extracting Location from JSON (/s/article/converting-from-json-to-a-spatial-format-gis) | **Next:** Writing JSON (/s/article/json-writing-overview)

Introduction

Some web services and APIs return complex JSON, including transactional calls where multiple queries are involved with accessing the nested data. This article shows how to work with such data, in this case, making a call to a flood data API that returns JSON data, extracting a URL from the JSON that points to GeoJSON data of flood extents, and then reading in that GeoJSON as geometry into the workspace. The output will be written to a KML file for viewing in Google Earth. (https://www.google.com/earth/versions/)

Step-by-step Instructions

1. Check API for Current Floods

Before we begin reading the data, let's first confirm that there is in fact data to read. We will be using data from Defra's Real Time flood-monitoring API (https://environment.data.gov.uk/flood-monitoring/doc/reference). Since we are reading from a live API that monitors floods, if there are no floods the day you are working through this tutorial, there will be no data to read.
Click on this link to open it up in a web browser:

```
https://environment.data.gov.uk/flood-monitoring/id/floods (https://environment.data.gov.uk/flood-monitoring/id/floods)
```

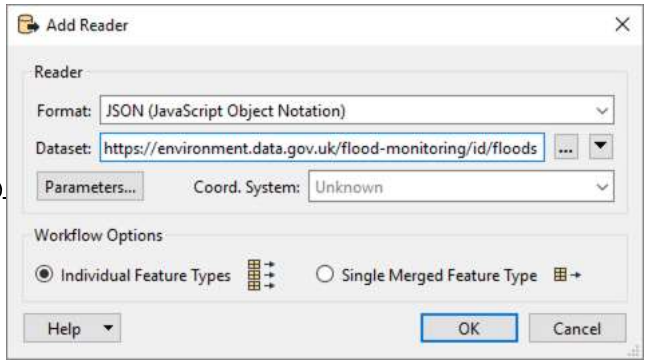
```
{
  "@context" : "http://environment.data.gov.uk/flood-
monitoring/meta/context.jsonld" ,
  "meta" : {
    "publisher" : "Environment Agency" ,
    "licence" : "http://www.nationalarchives.gov.uk/doc/open-government-
licence/version/3/" ,
    "documentation" : "http://environment.data.gov.uk/flood-
monitoring/doc/reference" ,
    "version" : "0.9" ,
    "comment" : "Status: Beta service" ,
    "hasFormat" : [ "http://environment.data.gov.uk/flood-
monitoring/id/floods.csv", "http://environment.data.gov.uk/flood-
monitoring/id/floods.rdf", "http://environment.data.gov.uk/flood-
monitoring/id/floods.ttl", "http://environment.data.gov.uk/flood-
monitoring/id/floods.html" ]
  }
  ,
  "items" : [  ]
}
```

2. Read in Flood Monitoring API

Open FME Workbench and start a new workspace. Add in a JSON (Javascript Object Notation) reader and if you had floods listed under items in the previous step, set the Dataset to:

https://environment.data.gov.uk/flood-monitoring/id/floods

_(https://environment.data.gov.uk/flood-monitoring/id/floods)



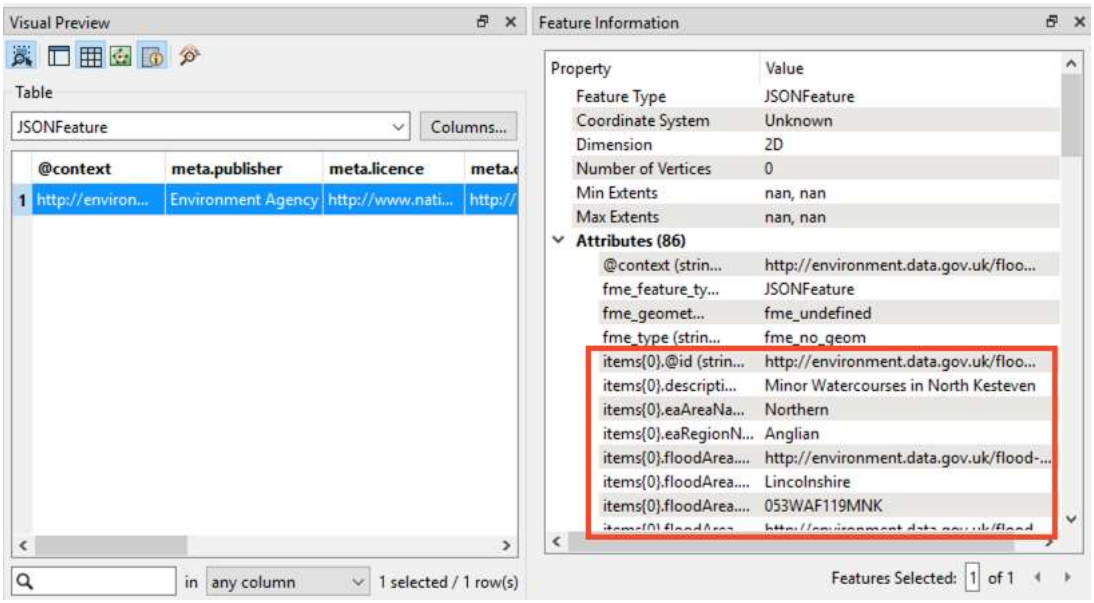
If there were no items listed, use the Floods.json dataset which is available in the Files section of this article.
Click OK to finish adding the reader.

3. Run Workspace

Run the workspace with Feature Caching enabled, then view the reader feature type cache in Visual Preview.

In Visual Preview, a single feature is displayed, with a few attributes exposed from the top level of the JSON hierarchy. However, to see the flood data itself, you have to select the feature and look at the unexposed list attributes in the Feature Information window. There you can see the rest of the data nested at lower

levels.



If you do not see the items{}. attributes, there is currently no flood data for the live API; change the reader dataset to Floods.json, which is available from the Files section of this article.

We can see that each flood alert is a child of the items node. This means if we change our Schema Scan Mode, we can flatten the JSON one level and read each flood alert as a single feature. We'll do that in the next step.

For now, notice the items{}.floodArea.polygon attributes. These contain a URL. If you open that URL in a browser (you can right-click > Copy Text to copy the URL), you'll see that this data is also JSON, but contains a "featureCollection" and geometry data, revealing it to be GeoJSON. You can confirm this by referring to the API documentation (https://environment.data.gov.uk/flood-monitoring/doc/reference#flood-areas). It will look something like this:

```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "properties": {
        "AREA": "Lincs and Northants",
        "FWS_TACODE": "053WAF119MNK",
        "TA_NAME": "Minor Watercourses in North Kesteven",
        "DESCRIP": "Heighington, Dunston, Blankney Becks, Billinghay Skirth and River Slea. Tributaries may also
affect Dunston, Scopwick, Digby, Ruskington, South Kyme, Sleaford, Ancaster, Wilsford, and Kelby",
        "LA_NAME": "Lincolnshire",
        "QDIAL": "207014",
        "RIVER_SEA": "Heighington Beck, Dunston Beck, Blankney Beck, Bil"
      },
      "geometry": {
        "type": "Polygon",
        "coordinates": [
          [
            [
              -0.513446263870744,
              53.06191620889139
            ],
...
```

Now that we know the structure of the data we are working with, we can see we'll have to extract the URLs linking to the polygon data and use them to read in geometry to our workspace.

4. Re-add Reader to Change Schema Query

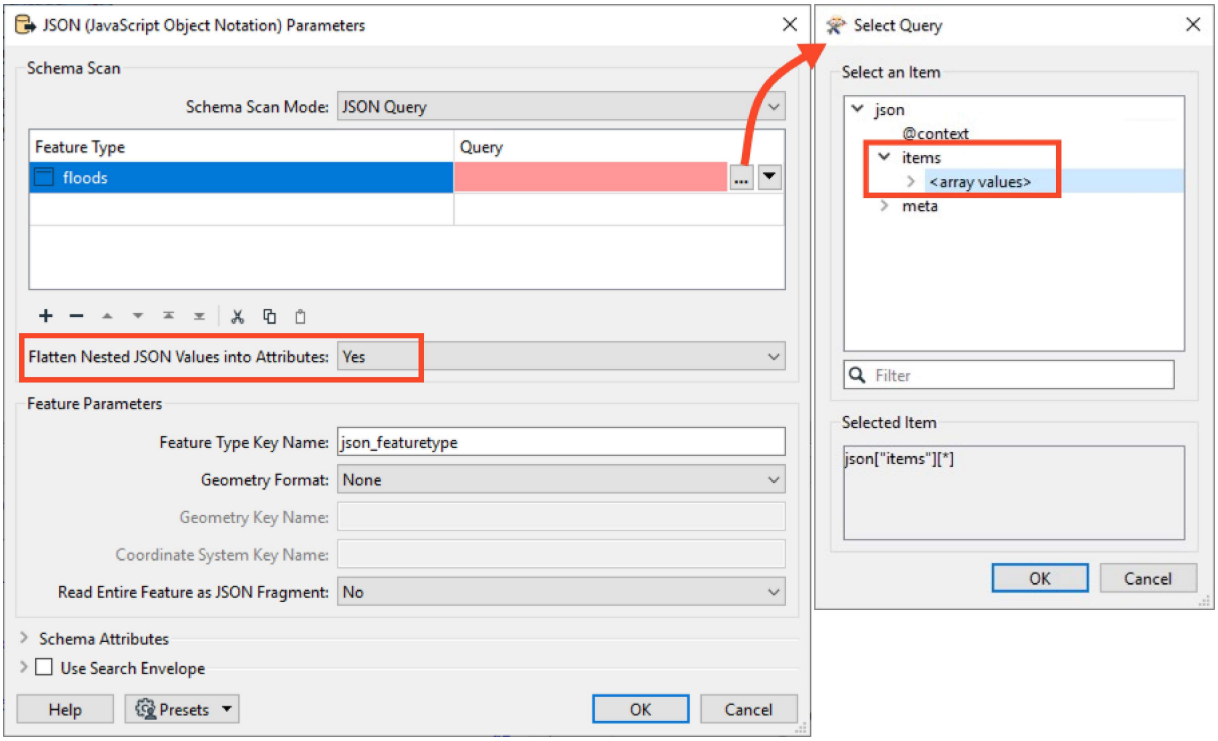
Before we can continue, we need to change the JSON Schema Query to read in the polygon data. Now there are ways to do this in a transformer, but that is covered in the Transforming JSON using the JSONExtractor, JSONFlattener, and JSONFragmenter (https://community.safe.com/s/article/json-transformations) article. For now, let's do this in the reader.

Delete the existing reader feature type from the canvas by highlighting it and pressing Delete or Backspace on your keyboard. When prompted to remove the reader/writer, click Yes.

Now add another reader to the canvas, it should have JSON cached as the Format, and either the API or Floods.json listed as the Dataset. Open the parameters, change the Schema Scan Mode to JSON Query, then create a new Feature Type called floods. For Query, click on the box to expose the ellipsis button, then click on the ellipsis to open the Query dialog. Expand items then select <array values>. The Selected Item should read:

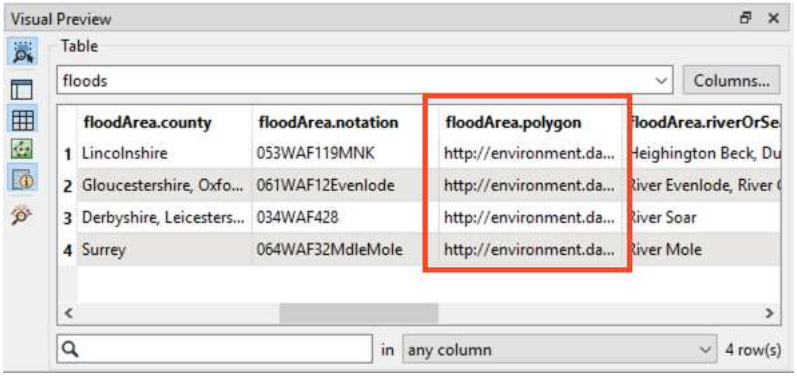
```
json["items"][*]
```

Click OK to confirm the Query. Finally, double-check that Flatten Nested JSON Values into Attributes is set to Yes. This parameter will flatten the floodArea.polygon into individual attributes. Click OK twice to finish adding the reader.



Note: You may have to click off of the Query line to accept the query for OK to be enabled.

Now if you run the workspace, you'll see we've flattened the JSON slightly, and now each feature is a flood alert with all its attributes exposed. We can see the floodArea.polygon attribute for each feature.

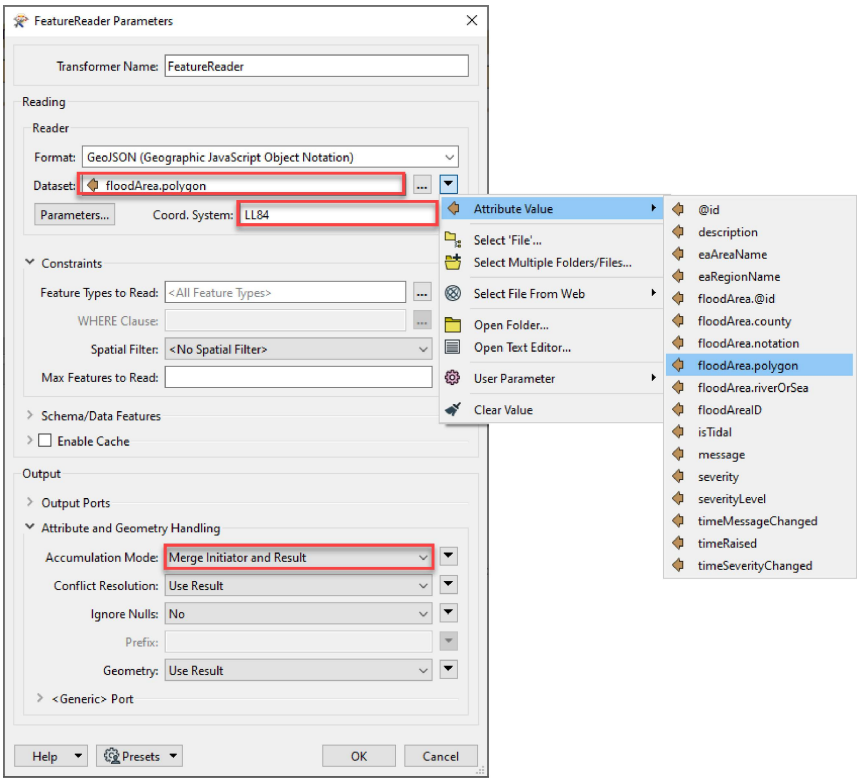


5. Read GeoJSON URL From floodArea.polygon

Add a FeatureReader transformer to the workspace and connect it to the reader feature type. We'll use this transformer to read in the GeoJSON from the URL in floodArea.polygon.

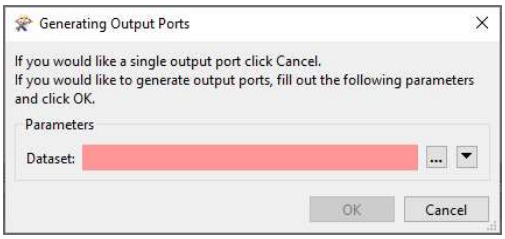
Set the following parameters in the FeatureReader:

- Reader
  - **Format**: GeoJSON (Geographic JavaScript Object Notation)
  - **Dataset**: floodArea.polygon
  - **Coord. System:** LL84
- Attribute and Geometry Handling
  - **Accumulation Mode:** Merge Initiator and Result

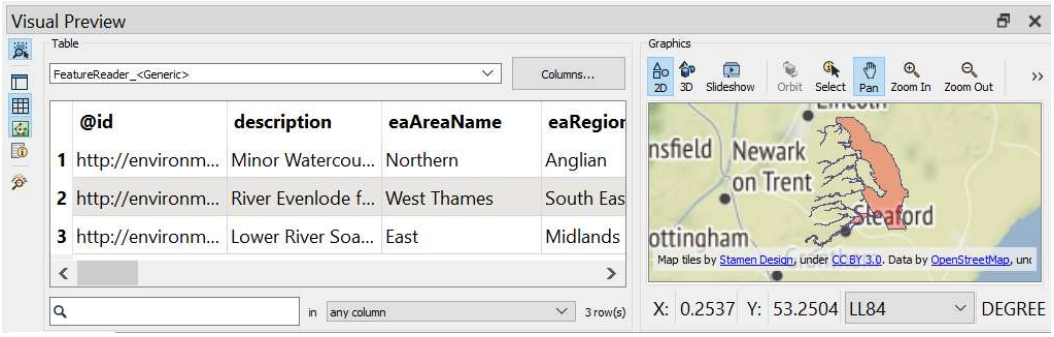


Note that if you see items.floodArea.polygon{} listed for attributes, you need to re-add the reader with Flatten Nested JSON set to Yes.

Click OK. The Generating Output Ports dialog will ask if you want to provide a dataset to use for generating output ports, but in this case, we don't need to do this, so we can click Cancel.

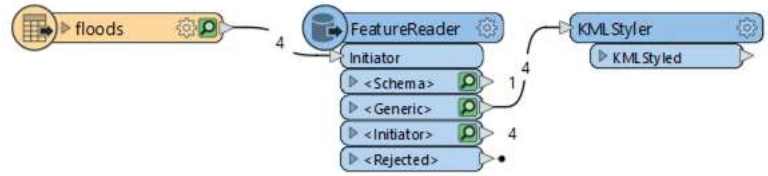


Run the workspace with Feature Caching enabled. Inspect the FeatureReader's <Generic> port cache to see that each flood alert now has a flood area polygon. Because we chose to Merge Initiator and Result, our data now has the correct exposed attributes and geometry retrieved from the URL:



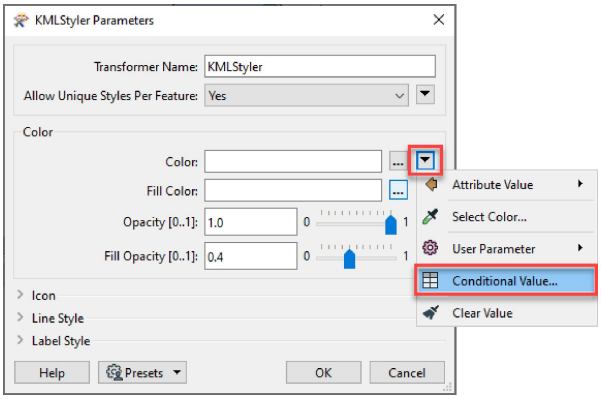
6. Add a KMLStyler

Add a KMLStyler to the canvas and connect it to the <Generic> output port on the FeatureReader.

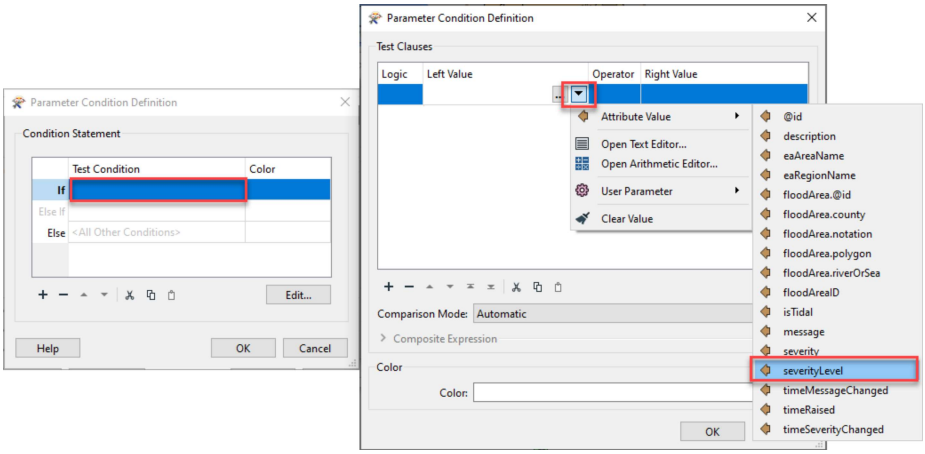


We'll use this to color the flood area polygons based on their severity, which is stored in the attribute severityLevel from 1 (least severe) to 4 (most severe).

Double-click the KMLStyler to open its parameters. Click the drop-down arrow next to Color and select Conditional Value:



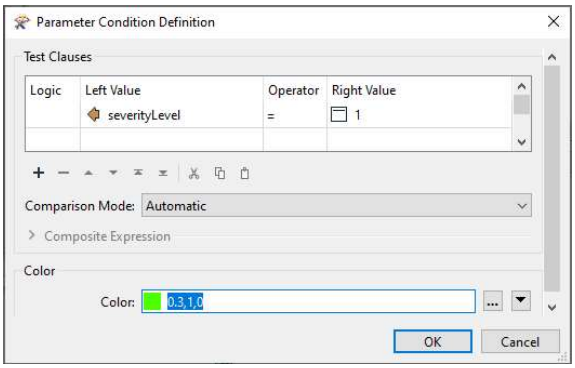
Then double-click in the first cell next to If.



Set the first If Test Condition to:

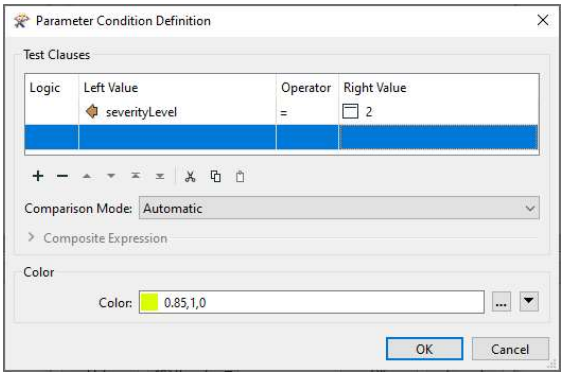
- **Left Value:** severityLevel
- **Operator:** =

- **Right Value**: 1
- **Color**: 0.3,1,0



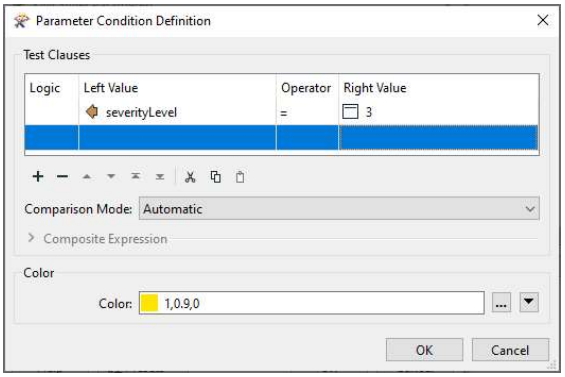
Click in the first cell of the next row next to Else If. Set the first If Else Test Condition to:

- **Left Value**: severityLevel
- **Operator**: =
- **Right Value**: 2
- **Color**: 0.85,1,0

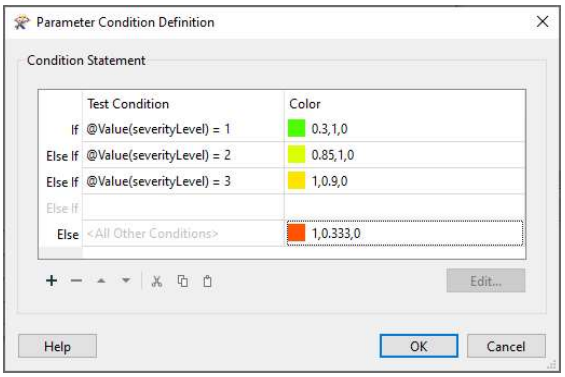


For the last Else If row, set the Test Condition to:

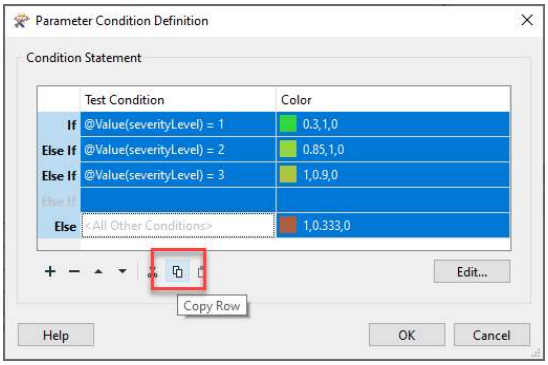
- **Left Value:** severityLevel
- **Operator**: =
- **Right Value**: 3
- **Color**: 1,0.9,0



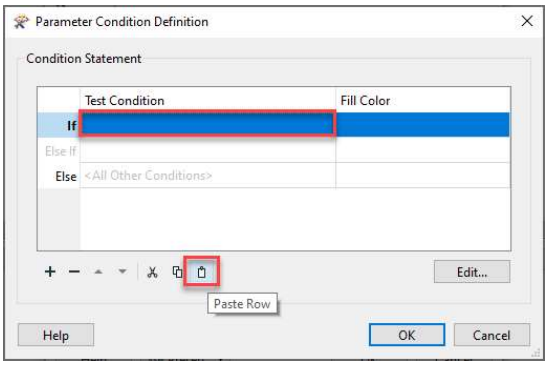
Finally, set the Color for the final Else row to 1,0.333,0. You should now have a green, yellow, and red color scheme based on severityLevel.



We want to use this same set of tests to determine the Fill Color as well, so select all rows and click the Copy button.

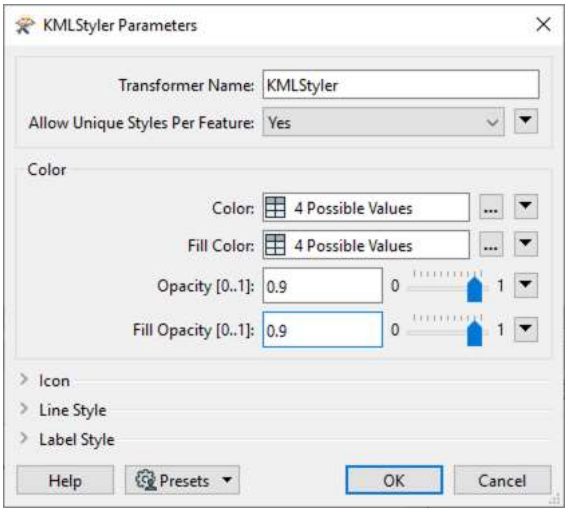


Click OK. Click the drop-down for Fill Color as well and select Conditional Values again. Select all the rows using Shift-click and click Paste.



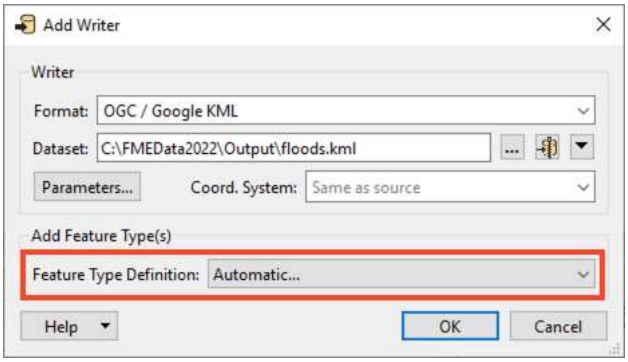
You'll have to manually type in the Else Color (1,0.333,0) again. Click OK.

Set Opacity and Fill Opacity to 0.9. Click OK.

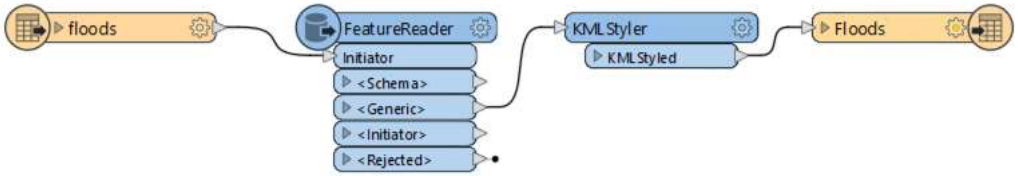


7. Add a KML Writer

Add an OGC/Google KML writer to the canvas and browse to a location to save the dataset; name it floods.kml. Change the Feature Type Definition to Automatic and click OK.

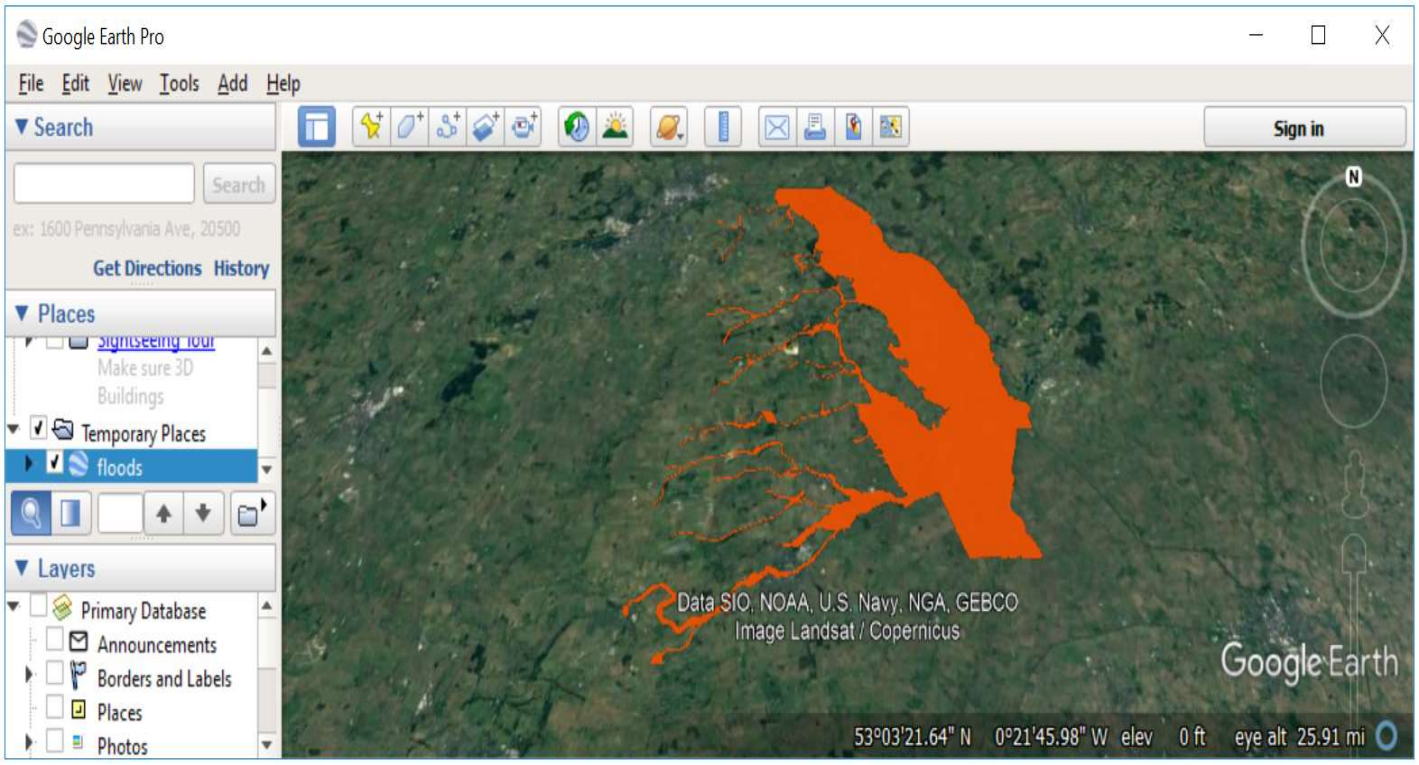


In the Feature Type dialog, change the Feature Type Names to Floods and then click OK. Connect the Floods writer feature type to the KMLStyler.



8. Run Workspace and View Output in Google Earth

Run the workspace and open the results in Google Earth Pro (https://www.google.com/earth/versions/#download-pro). You should see flood area polygons, which will display their flood alert attributes when clicked.



Note: Because this data source is updated daily, your results may look different. If multiple alerts are present in the same area, you'll have to toggle areas on and off in the Google Earth Pro Places window.

Continue to the next article: Writing JSON (/s/article/json-writing-overview)

Data Attribution

This article uses Environment Agency (https://environment.data.gov.uk/flood-monitoring/doc/reference) flood and river level data from the real-time data API (Beta) which is licensed under the Open Government Licence v3.0.

First Published Date
7/29/2020, 12:18 AM

Last Published Date
8/4/2022, 1:58 AM

Transformation (/s/topic/0TO4Q000000...

FME Desktop (/s/topic/0TO4Q000000...

Sort by:

Latest Posts

LizAtSafe (/s/profile/0050c00000CeGV0AAN) (Employee) published a new version of this Knowledge.
August 4, 2022 at 1:58 AM (/s/feed/0D54Q00009ggWD5SAM)

Like Comment

Log In to Comment

Follow

Files (3) (/s/relatedlist/ka14Q000000wn9KQAQ/AttachedContentDocuments)

advanced-json-reading-2022
Jun 15, 2022 • 104KB • fmw

floods
Jul 28, 2020 • 21.5MB • kml

floods
Jul 28, 2020 • 7KB • json

View All
(/s/relatedlist/ka14Q000000wn9KQAQ/AttachedContentDocuments)

Related Articles

Reading JSON (/s/article/json-reader-configuration)

Tutorial: Getting Started with JSON (/s/article/tutorial-getting-started-with-json)

Extracting Location from JSON (/s/article/converting-from-json-to-a-spatial-format-gis)

Writing JSON with the JSONTemplater (/s/article/json-writing-with-jsontemplater)

Advanced API Calls (/s/article/advanced-api-calls-in-fme)

Land Acknowledgement —

Safe Software respectfully acknowledges that we live, learn and work on the traditional and unceded territories of the Kwantlen, Katzie, and Semiahmoo First Nations.