Updating GIS Content

Written By: MATINGO KUMBIRAI N

Welcome yet again to another script of the **#LearnArcGIS** tutorial series.

In our last script and tutorial, we learn't how to publish to an ArcGIS Online Organization account. In this tutorial we are going to be focusing on two things which are:

1. Removing content &
2. Adding Content

to WebMaps. We are going to be mainly dealing with *Feature layers* in this tutorial.

Preparation

Like in previous tutorials, this first step is crucial when beginning a project.

In this step, we are going to:

- *Import GIS libraries and the Python display libraries*
- Enter login credentials (`username` and wait for password prompt)

In [1]:

```
from arcgis.gis import GIS
from IPython.display import display

# Login to Organization account
gis = GIS("http://arcgis.com/", "africansurveyors")
```

```
Enter password: ········
```

Search for Web map

We want to render the Web Map & Prepare it for updating with a Feature Layer which we are going to introduce later in the tutorial.

- specify the `title` which you are searching for
- define the `item_type` to filter through the results and remove irrelevant content to our case.
- finally, display the search output results.

The output result, will basically list the basic information of the Web Map that has been rendered

In [2]:

```
# Return the default Zimbabwe Districts Map
search_result = gis.content.search("title: Zimbabwe Districts", item_type="Web Map")
display(search_result)
```

```
[<Item title:"Zimbabwe Districts" type:Web Map owner:africansurveyors>]
```

Read the WebMap as a WebMap object

- define a variable which is going to store the results from our search
- import the `WebMap` method from the **mapping** libraries
- define the web map object as a variable which is going to help us display the **WebMap** and then pass the variable containing the search results as a parameter.

In [3]:

```
web_map_item = search_result[0]
from arcgis.mapping import WebMap
```

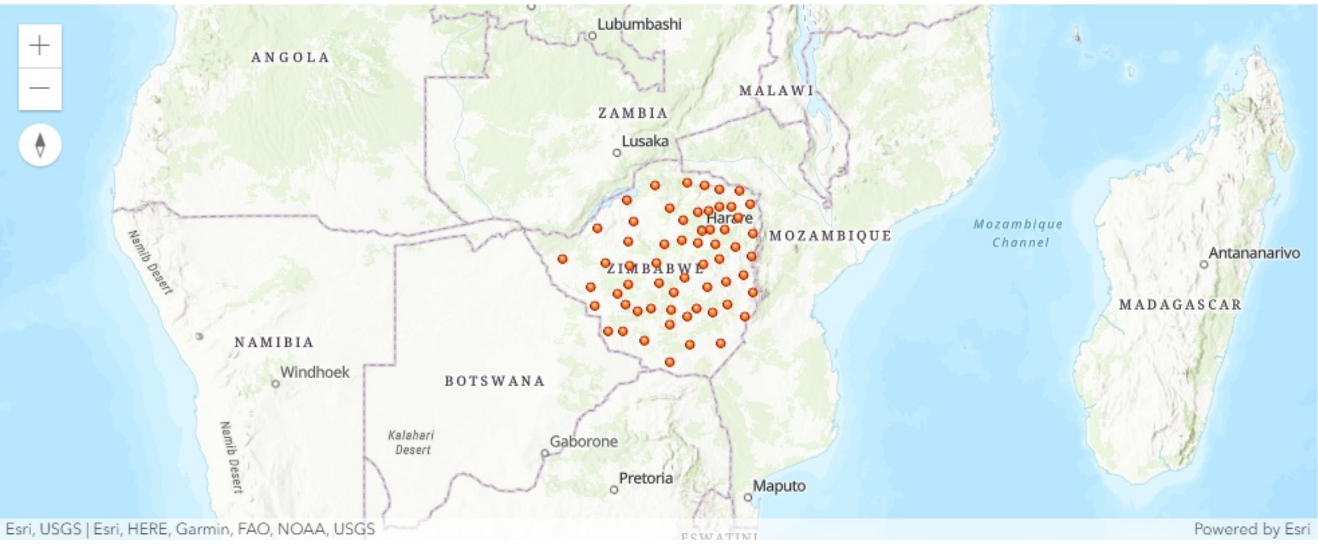
```
web_map_obj = WebMap(web_map_item)
```

Display the WebMap

Call the variable which contains the `WebMap()` method to give us the WebMap which we have requested from the search function.

In [4]:

```
web_map_obj
```



Get Map Layer Details

Query the layers from a webmap using the `layers` property.

Using this property will return a JSON output containing all the Metadata about our WebMap. Refer to *output[5]* below.

In [5]:

```
web_map_obj.layers
```

Out[5]:

```
[{
   "id": "ZWE_Districts_7659",
   "layerType": "ArcGISFeatureLayer",
   "url":
"https://services5.arcgis.com/UMq1wI6u64peUmQF/arcgis/rest/services/ZWE_Districts/FeatureServer/0",

   "visibility": true,
   "opacity": 1,
   "title": "ZWE Districts",
   "itemId": "b7bfba7d36fc4205960cccb50678b9c6",
   "popupInfo": {
     "title": "ZWE_Districts: {VARNAME_2}",
     "fieldInfos": [
       {
         "fieldName": "ID_0",
         "label": "ID_0",
         "isEditable": true,
         "tooltip": "",
         "visible": true,
         "stringFieldOption": "textbox",
         "format": {
           "places": 0,
           "digitSeparator": true
         }
       },
       {
        {
          "fieldName": "ISO",
          "label": "ISO",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
          "stringFieldOption": "textbox"
        },
        {
          "fieldName": "NAME_0",
          "label": "NAME_0",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
          "stringFieldOption": "textbox"
        },
        {
          "fieldName": "ID_1",
          "label": "ID_1",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
          "stringFieldOption": "textbox",
          "format": {
            "places": 0,
            "digitSeparator": true
          }
        },
        {
          "fieldName": "NAME_1",
          "label": "NAME_1",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
          "stringFieldOption": "textbox"
        },
        {
          "fieldName": "ID_2",
          "label": "ID_2",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
          "stringFieldOption": "textbox",
          "format": {
            "places": 0,
            "digitSeparator": true
          }
        },
        {
          "fieldName": "NAME_2",
          "label": "NAME_2",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
          "stringFieldOption": "textbox"
        },
        {
          "fieldName": "TYPE_2",
          "label": "TYPE_2",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
          "stringFieldOption": "textbox"
        },
        {
          "fieldName": "ENGTYPE_2",
          "label": "ENGTYPE_2",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
          "stringFieldOption": "textbox"
        },
        {
          "fieldName": "NL_NAME_2",
          "label": "NL_NAME_2",
          "isEditable": true,
          "tooltip": "",
          "visible": true,
        "visible": true,
        "stringFieldOption": "textbox"
      },
      {
        "fieldName": "VARNAME_2",
        "label": "VARNAME_2",
        "isEditable": true,
        "tooltip": "",
        "visible": true,
        "stringFieldOption": "textbox"
      },
      {
        "fieldName": "x",
        "label": "x",
        "isEditable": true,
        "tooltip": "",
        "visible": true,
        "stringFieldOption": "textbox",
        "format": {
          "places": 2,
          "digitSeparator": true
        }
      },
      {
        "fieldName": "y",
        "label": "y",
        "isEditable": true,
        "tooltip": "",
        "visible": true,
        "stringFieldOption": "textbox",
        "format": {
          "places": 2,
          "digitSeparator": true
        }
      },
      {
        "fieldName": "ObjectId",
        "label": "ObjectId",
        "isEditable": false,
        "tooltip": "",
        "visible": false,
        "stringFieldOption": "textbox"
      }
    ],
    "description": null,
    "showAttachments": true,
    "mediaInfos": []
  }
}]
```

Search for the replacement layer

We are going to use the search function again, but this time we are searching for a *layer* which we want to use to replace the already existing layer within our WebMap.

In [6]:

```
search_result = gis.content.search('title: ZWE Districts(polygon)', item_type='Feature Service')
search_result[0]
```

Out[6]:

ZWE Districts(polygon)

Feature Layer Collection by africansurveyors
Last Modified: March 12, 2021
0 comments, 3 views

Update Web Map with New Feature Layer

In this step, we are going to be repeating what we did in Cell [3]. Only this time we are using the **Feature Layer** which was returned from the search results.

We will be declaring a variable named `new_layer` which stores the search results and then calls the layers property just to display the URL of the Feature Layer.

In [7]:

```
new_layer = search_result[0]
new_layer.layers
```

Out[7]:

```
[<FeatureLayer
url:"https://services5.arcgis.com/UMq1wI6u64peUmQF/arcgis/rest/services/ZWE_Districts(polygon)/Feat
erver/0">]
```

Removing the Old Layer & Add New Layer

*Remember: Earlier on, we declared a variable `web_map_obj` which contained the WebMap which is displayed above and within that WebMap, we used the **layers** property to get the layer.*

We need to keep it that way and now use the `remove_layer()` method and pass in that variable with its layer as a parameter to remove the current feature layer from the web map.

Once the layer has been removed, we can prepare to add another layer (the one which is going to replace the existing one)

In [8]:

```
web_map_obj.remove_layer(web_map_obj.layers[0])
```

Add The New Layer

In this step, the syntax is the same above. The only differene being that we are now using the `add_layer()` method which is going to get a new layer to be added to our map.

Also, we are going to be changing the paramater to be passed in the method. *Hint: In cell [7] above, we declared a variable which stores the search result with our **new** feature layer.*

- We are going be altering the title for the layer in this section (Just to show you that we can also be able to alter the properties when using the `update_layer()` method.
- if **True** is displayed as an output, that proves our layer has been added successfully.

In [9]:

```
web_map_obj.add_layer(new_layer.layers[0], options={'title': 'Districts in Zimbabwe'})
```

Out[9]:

```
True
```

Check the Layers on the WebMap

In this step, we are just going to loop through the layers which are contained in the `web_map_obj` variable just to be sure our new layer has replaced the old one before we make any updates also to the WebMap.

- check the name of the layer displayed as an output after you have ran the cell block.
- the output cell should only render one layer name. If that's not the case in your scenario, check and verify you have followed the right conventions.

In [10]:

```
for lyr in web_map_obj.layers:
    print(lyr.title + " <==> " + lyr.url)
```

```
Districts in Zimbabwe <==>
https://services5.arcgis.com/UMq1wI6u64peUmQF/arcgis/rest/services/ZWE_Districts(polygon)/FeatureSe
/0
```

Update the WebMap

Now that we have our new Feature Layer in order, it's time to update the WebMap and tweak a few things.

We are just going to make a few changes by doing the following:

- change the title (rename the webmap)
- place a thumbnail to our webmap

The cell block should again render **True** as output to prove that our WebMap has been successfully updated. If that is not the case for you, kindly refer back and trace especially in your variable naming and declarations.

In [11]:

```
web_map_obj.update(item_properties={'title': 'Districts in Zimbabwe - updated'},
                   thumbnail="https://wallpaperaccess.com/full/2017034.jpg")
```

Out[11]:

```
True
```

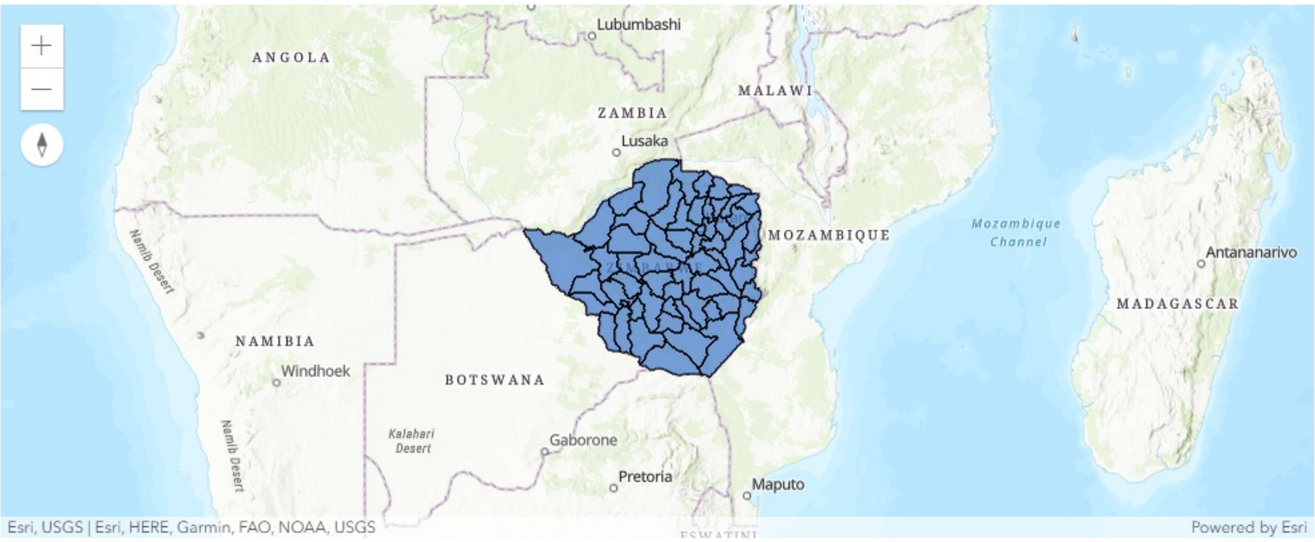
Visualization

It's time to visualize our New WebMap with a new Feature Layer.

Just call the variable which contains the WebMap and the output will be rendered as shown below.

In [12]:

```
web_map_obj
```



JUST A RE-CAP

Here is an outline about what we did and learn't in this tutorial.

We,

- Searched for our webmap

- Retrieved details for the WebMap layers in JSON format
- Searched for a layer which is going to act as a replacement
- Removed the old layer
- Added a new layer to our webmap
- Updated the Properties and added a thumbnail to our WebMap

Remeber: You can also keep these Notebooks as templates for easy operation within your Organization

Online Resources

We have also provided a list of the resources for this tutorial series which you might find useful to see your way to the end.

- GitHub Source Code
- Tutorial Raw Data
- YouTube Channel - be sure to subscribe and show your support too.
- Website(blog)

If you didn't face any errors along the way and managed to get your places geocoded then;

CONGRATULATIONS

You can easily save this as a template for any Geocoding processes you would like to perform in the future.

For anyone having trouble or fails to understand this tutorial, I am reachable via LinkedIn. Just send me a direct message and I will be sure to respond to any questions relating to the tutorials that you might have.

About Author

- 3rd Year BSc Hons in Surveying & Geomatics
- Interested in GIS for Health and Land Administration, Spatial Data Science and Programming
- currently working on a book titled: **GIS Step by Step: A Practical Guide to GIS**