

Regex Testing to Clean Data

Use Brackets to do Regex. VSCode confused.

<https://regex101.com> for testing.

Example URL:

```
"<a target=\"_blank\" href=\"https://maps.psiee.psu.edu/PSU_ARB/px/Abelia x grandiflora_2 (479).jpg\"><img src=\"https://maps.psiee.psu.edu/PSU_ARB/px/rs/Abelia x grandiflora_2 (479).jpg\"></a>"
```

Removes up to the first https:// link.

```
"<a[^>]+href=\\
```

Almost selects just the URL

```
"(.*)\"[>]*
```

Selects the img src:

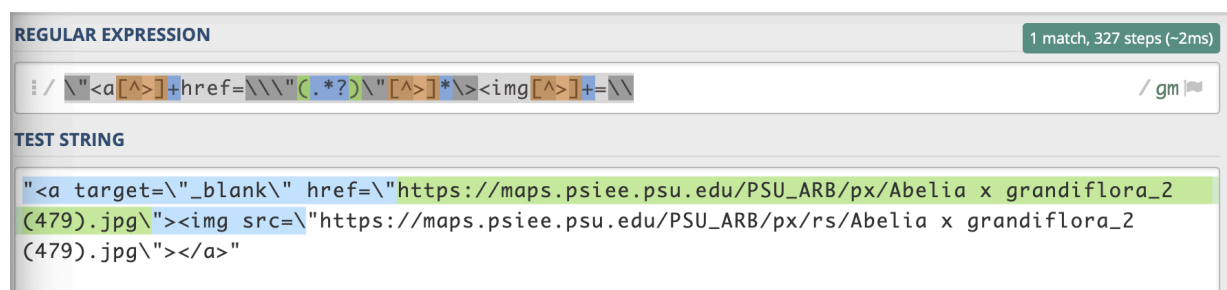
```
<img[^>]+=\\
```

Strategy: Remove the "a target" portion up to img src

Step 1:

Selects up to the first quote on the img src link:

```
"<a[^>]+href=\\\"(.*)\"[>]*\\><img[^>]+=\\
```



Step 2:

Remove the end <\a> tag and slash after jpg. Leaves the quote mark on the end.

```
\\\">|<\/a>
```

REGULAR EXPRESSION
1 match, 10 steps (~3ms)

/ \>\<\|a> /gm

TEST STRING

""

Step3:
Find all spaces in the url strings.

(\ *?)

Then add "%20" in each of the spaces of the URL to make it a valid URL.

Replace with space

(*? \| *?)

Captures all dots:

(\.)

Working on clearing geometry in each

"geometry": {\s.*\.".*\s.*\s.*\s.*\s.*\s*},

REGULAR EXPRESSION
1 match, 53 steps (~0ms)

/ "geometry": {\s.*\.".*\s.*\s.*\s.*\s.*\s*}, /gm

TEST STRING

```
"geometry": {
  "x": -8668240.472865166,
  "y": 4983705.080650414,
  "spatialReference": {
    "wkid": 102100,
    "latestWkid": 3857
  }
},
```

Removing Picture2-5, not needed, causes errors.

`,\s.*"Picture2": ".*\n.*\n.*\n.*`

Removes comma, JSON remains valid.

REGULAR EXPRESSION

1 match, 1050 steps (~112ms)

`,\s.*"Picture2": ".*\n.*\n.*\n.*`

TEST STRING

```
{
  "attributes": {
    "OBJECTID_12": 288,
    "OBJECTID_1": 0,
    "OBJECTID": 0,
    "AccYear": "2009",
    "AccNo": "254",
    "ItemNo": "AA",
    "ItemCoordX": 40.80564235,
    "ItemCoordY": -77.86821838,
    "AccNoFull": "2009-0254*AA",
    "Accession": "2009-0254*AA",
    "Family": "Cornaceae",
    "Genus": "Cornus",
    "Taxon_Name": "Cornus sanguinea 'Midwinter Fire'",
    "Common_Nam": "bloodtwig dogwood",
    "Status": "Inspected",
    "Condition": "Excellent",
    "Number_of": "15",
    "Location": "Strolling Garden Bed 3",
    "Latitude": 40.805642,
    "Longitude": -77.868218,
    "Taxon_Dist": "AS",
    "Life_Form": "Deciduous shrub/sub-shrub",
    "Picture": "https://maps.psiee.psu.edu/PSU_ARB/px/rs/Cornus sanguinea 'Midwinter Fire'_2 (562).jpg",
    "Picture2": "https://maps.psiee.psu.edu/PSU_ARB/px/rs/Cornus sanguinea 'Midwinter Fire'_2 (334).jpg",
    "Picture3": " ",
    "Picture4": " ",
    "Picture5": " "
  }
},
```

Found that OBJECTID and OBJECTID_1 have duplicate values for plants. Like 0 has 1057 matches. Not good for identifiable. Using OBJECTID_12 instead. Not removing.

NOTE: Just ignoring for now.

`.*"OBJECTID_1": ".*\n.*\n`

REGULAR EXPRESSION

1 match, 1126 steps (~2ms)

/

.*"OBJECTID_1": .*\\n.*\\n

/ gm

TEST STRING

```
{
  "attributes": {
    "OBJECTID_12": 288,
    "OBJECTID_1": 0,
    "OBJECTID": 0,
    "AccYear": "2009",
    "AccNo": "254",
```

Removing "Latitude, Longitude", preferring to use ItemCoordX and Y due to accuracy of Coordinates. No need for duplicate values.

```
.*"Latitude": .*\\n.*\\n
```

REGULAR EXPRESSION

1 match, 1106 steps (~2ms)

// `.*"Latitude":.*\n.*\n`

/ gm

TEST STRING

```
{
  "attributes": {
    "OBJECTID_12": 288,
    "OBJECTID_1": 0,
    "OBJECTID": 0,
    "AccYear": "2009",
    "AccNo": "254",
    "ItemNo": "AA",
    "ItemCoordX": 40.80564235,
    "ItemCoordY": -77.86821838,
    "AccNoFull": "2009-0254*AA",
    "Accession": "2009-0254*AA",
    "Family": "Cornaceae",
    "Genus": "Cornus",
    "Taxon_Name": "Cornus sanguinea 'Midwinter Fire'",
    "Common_Nam": "bloodtwig dogwood",
    "Status": "Inspected",
    "Condition": "Excellent",
    "Number_of": "15",
    "Location": "Strolling Garden Bed 3",
    "Latitude": 40.805642,
    "Longitude": -77.868218,
    "Taxon_Div": "ASC"
```