TIME-SAVING TIPS FOR CREATING MAP SETS FROM SUBSURFACE CHEMISTRY DATA

Parker Wittman
Senior Data Scientist/GIS Manager
Aspect Consulting – Seattle, WA

2014 ESRI INTERNATIONAL USER CONFERENCE
TUESDAY, 15 JUL 2014, 3:15PM - 4:30PM
SAN DIEGO CONVENTION CENTER

UC1146
WHAT’S THE PROBLEM?

1. We’ve got mountains of data.
2. The data mountain keeps growing.
3. We’re producing stacks of maps from these mountains of data.
4. We have ever-changing cleanup criteria.
5. We’ve got data in three* dimensions... and only two dimensions on which to display it.
6. Everyone likes a pretty map.

* four, if you count time.
SOIL BORINGS

Sample #1
ONE X/Y LOCATION, WITH SAMPLES/RESULTS FROM MULTIPLE DEPTHS

Sample #2

Sample #3

Soil Sample

- Petroleum
- Metals
- Volatiles
- Semi-Volatiles
- PCBs
- Dioxins
- Etc.
- Etc.
- Etc.
MONITORING WELLS

ONE X/Y LOCATION, WITH SAMPLES/RESULTS FROM MULTIPLE DATES

1/15/07
1/28/09
7/4/12
10/15/13

Petroleum
Metals
Volatile
Semi-Volatiles
PCBs
Dioxins
Etc., Etc.
AND WE’RE ALWAYS ADDING EXPLORATIONS…
SO WHAT DO WE DO?

1. Create thoughtful templates and organize input data to simplify map updates.
2. Use ‘small multiples’.
3. Let logic live outside the MXDs.
4. Break out a little bit of arcpy.mapping.
5. Use sneaky tricks to create “table labels”.
...AND WE NEED MAP UPDATES TO BE AS EASY AND FAST AS POSSIBLE.
CARTOGRAPHIC APPROACH

Small Multiples
KEEP YOUR BASE SIMPLE, DESATURATED, AND VECTOR…

(your data will thank you for it)
IT’S ALL IN THE SETUP.
IT’S ALL IN THE SETUP.

Free the logic from your MXDs.
DON’T DO THIS:

• Symbolize based on *quantities* (binning)

• Hard-code screening levels into definition queries

• Use separate featureclasses or shapefiles

DO THIS:

• Add an attribute that defines it’s “bin”

• Compare your data to screening levels *before* it becomes map data

• Use one big dataset and make use of definition queries!

ASK YOURSELF:

“Can I make this a new map simply by updating a few definition queries?”
SAME X/Y DATA TABLE FEEDS ALL MAPS + FRAMES.

EACH FRAME IS A UNIQUE COMBO OF INTERVAL + ANALYTE.

SYMBOLOGY DRIVEN BY FIELD CALLED “SYMBOL CODE”.

SAMPLED \rightarrow \text{‘101’} \rightarrow \text{EXCEEDED}

(DETECTED)

(SAME FOR ALL)
PYTHON TO THE RESCUE!
mxd = arcpy.mapping.MapDocument("CURRENT")
findtext = "Arsenic"
replacetext = "Mercury"

for lyr in arcpy.mapping.ListLayers(mxd):
    if lyr.supports("DEFINITIONQUERY"):
        lyr.definitionQuery = lyr.definitionQuery.replace(findtext, replacetext)

del mxd, findtext, replacetext
...BUT WHY STOP THERE?
...BUT WHY STOP THERE?

If you **stay consistent** in your naming conventions you can (easily) find/replace globally!

(THAT IS, IF YOU CAN HELP IT, DON’T CALL IT “MERCURY” ALL OVER THE LAYOUT, “HG” IN ONE DATA TABLE AND “MERC” IN ANOTHER...)
FIND AND REPLACE IN DEFINITION QUERIES AND TEXT ELEMENTS:

```python
mxd = arcpy.mapping.MapDocument("CURRENT")
findtext = "Arsenic"
replacetext = "Mercury"

# Loop through each text element in the map document
for textElement in arcpy.mapping.ListLayoutElements(mxd, "TEXT_ELEMENT"):
    # Find and replace string in Text Element
    textElement.text =
    textElement.text.replace(findtext,replacetext)

# Find and replace string in MXD title/summary
mxd.title = mxd.title.replace(findtext,replacetext)
mxd.summary = mxd.summary.replace(findtext,replacetext)

# Find and replace in def queries
for lyr in arcpy.mapping.ListLayers(mxd):
    if lyr.supports("DEFINITIONQUERY"):
        lyr.definitionQuery =
        lyr.definitionQuery.replace(findtext,replacetext)

del mxd, findtext, replacetext
```
AND THAT'S JUST THE TIP OF THE ICEBERG

(you can iterate, create parameterized tools, and so much more.)
NEVER MAKE ONE MAP WHEN TEN WILL DO.
YET... SOMETIMES THINGS AREN’T THAT SIMPLE
ATTACK OF THE “TABLE LABELS”
WHAT IF WE COULD CREATE THEM DYNAMICALLY INSIDE OF ARCMAP?
**[LabelField](Value):**

<UND><BOL>MW-01</BOL>***LEAD***MERCURY</UND>#0-2FT***2.0U***0.1U#<CLR red='255'><BOL>2-4FT</BOL></CLR>***<CLR red='255'><BOL>50</BOL></CLR>*****0.1U#8-12FT**8.0****<CLR red='255'><BOL>10</BOL></CLR></BOL></CLR>

**Label Expression:**

Replace(Replace([LabelField],"*"," "),"#",VbNewLine)

---

Beware of field length limits!

```
<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>MW-01</td>
<td>LEAD</td>
<td></td>
</tr>
<tr>
<td>MERCURY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0-2FT</td>
<td>2.0U</td>
<td></td>
</tr>
<tr>
<td>0.1U</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2-4FT</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>8-12FT</td>
<td>8.0</td>
<td>10</td>
</tr>
</tbody>
</table>
```

---

199 Chars (yikes)
REMINDER: LEAVE THE LOGIC OUTSIDE OF ARCMAP.
[LabelField](Value):

?{MW-01}***LEAD***MERCURY@#0-2FT***2.0U***0.1U#!{2-4FT}i***!{50}i******0.1U#8-12FT**8.0*****!{10}i

**Label Expression:**

Replace(Replace(Replace(Replace(Replace(Replace(Replace(

Replace([LabelField],"{","<BOL>"),"}" ,"</BOL>" ),"?","<UND>"),"@","</UND>"),"!","<CLR red='255'>"),

"i","</CLR>"),"#",vbNewLine),"*","" )

<table>
<thead>
<tr>
<th>MW-01</th>
<th>LEAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>MERCURY</td>
<td></td>
</tr>
<tr>
<td>0-2FT</td>
<td>2.0U</td>
</tr>
<tr>
<td>0.1U</td>
<td></td>
</tr>
<tr>
<td>2-4FT</td>
<td>50</td>
</tr>
<tr>
<td>0.1U</td>
<td></td>
</tr>
<tr>
<td>8-12FT</td>
<td>8.0</td>
</tr>
</tbody>
</table>

{ = <BOL> = (Begin Bold Formatting) 
} = </BOL> = (End Bold Formatting) 
? = <UND> = (Begin Underline Formatting) 
@ = </UND> = (End Underline Formatting) 
! = <CLR red='255'> = Begin Red Text Formatting 
! = </CLR> = End Red Text Formatting 
# = vbNewLine = Carriage Return 
* = Space
If you stay consistent in your library of formatting marks, you can use the same label expression for all sorts of “table labels”.

{ = <BOL> = (Begin Bold Formatting) } = </BOL> = (End Bold Formatting)
? = <UND> = (Begin Underline Formatting)
@ = </UND> = (End Underline Formatting)
! = <CLR red='255'> = Begin Red Text Formatting
  = </CLR> = End Red Text Formatting
# = vbNewLine = Carriage Return
* = Space
<table>
<thead>
<tr>
<th>SB-16</th>
<th>SHAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEEP</td>
<td></td>
</tr>
<tr>
<td>ARSENIC</td>
<td>1.3</td>
</tr>
<tr>
<td>ND</td>
<td></td>
</tr>
<tr>
<td>COPPER</td>
<td>10</td>
</tr>
<tr>
<td>37</td>
<td></td>
</tr>
<tr>
<td>LEAD</td>
<td>ND</td>
</tr>
<tr>
<td>140</td>
<td></td>
</tr>
<tr>
<td>MERCURY</td>
<td>ND</td>
</tr>
<tr>
<td>Nickel</td>
<td>ND</td>
</tr>
<tr>
<td>1.3</td>
<td></td>
</tr>
<tr>
<td>ZINC</td>
<td>18</td>
</tr>
<tr>
<td>1.9</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SB-15</th>
<th>SHAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEEP</td>
<td></td>
</tr>
<tr>
<td>ARSENIC</td>
<td>ND</td>
</tr>
<tr>
<td>5.6</td>
<td></td>
</tr>
<tr>
<td>COPPER</td>
<td>8.1</td>
</tr>
<tr>
<td>ND</td>
<td></td>
</tr>
<tr>
<td>LEAD</td>
<td>ND</td>
</tr>
<tr>
<td>ND</td>
<td></td>
</tr>
<tr>
<td>MERCURY</td>
<td>ND</td>
</tr>
<tr>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>NICKEL</td>
<td>ND</td>
</tr>
<tr>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>ZINC</td>
<td>1.1</td>
</tr>
<tr>
<td>1.9</td>
<td></td>
</tr>
</tbody>
</table>

[LabelField] (for SB-16):

```text
{SB-16}*****SHAL****DEEP@#ARSENIC***1.3*****ND#
{COPPER}****{10}****37#!{LEAD}****ND******{140}#*
{MERCURY}***ND*****{1.5}#NICKEL*****ND*****{1.6}#*
{ZINC}**{18}**{12}**1.9
```

Same Label Expression (!):

Replace (Replace (Replace (Replace (Replace (Replace (Replace (Replace (Replace ([LabelField],"{","<BOL>"),"}","</BOL>")
,"?","<UND>"),"@","</UND>")),"!","<CLR red='255'>")
,"i","</CLR>"),"#",vbNewLine),"*"," "
)
Col 1 Width = Max(length) + a couple

Width doesn’t matter. Just add a line break.

<table>
<thead>
<tr>
<th>SB-15</th>
<th>SHAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEEP</td>
<td></td>
</tr>
<tr>
<td>ARSENIC</td>
<td>ND</td>
</tr>
<tr>
<td>5.6</td>
<td></td>
</tr>
<tr>
<td>COPPER</td>
<td>8.1</td>
</tr>
<tr>
<td>ND</td>
<td></td>
</tr>
<tr>
<td>LEAD</td>
<td>ND</td>
</tr>
<tr>
<td>ND</td>
<td></td>
</tr>
<tr>
<td>MERCURY</td>
<td>ND</td>
</tr>
<tr>
<td>0.2</td>
<td></td>
</tr>
<tr>
<td>NICKEL</td>
<td>1.3</td>
</tr>
<tr>
<td>1.3</td>
<td></td>
</tr>
<tr>
<td>ZINC</td>
<td>1.1</td>
</tr>
<tr>
<td>1.9</td>
<td></td>
</tr>
</tbody>
</table>

THE FORMULA TO CREATE THE STRING LOOKS SOMETHING LIKE:

```
"?" & [LocName] & Repeat("*", [Col1Width] – Len([LocName])) ... & "@" & "#"...
& [Row1Name] & Repeat("*", [Col1Width] – Len([Row1Name])) & [Row1Value1] & Repeat("*", [Col2Width] – Len([Row1Value1]))...
```

THE TRICKY PART IS CREATING THE LABEL STRING…
(but even that’s not so bad once you get the hang of it.)
THANKS!

Parker Wittman
Senior Data Scientist/GIS Manager
Aspect Consulting - Seattle, WA

pwittman@aspectconsulting.com
@ParkerPW
EXTRA TIME?
MULTI-VARIABLE SYMBOLOLOY (IF YOU MUST)
SKIP THE FANCY LOGIC.
LAYER IT UP.