Mastering Expressions

Tracy.Rijken@TeachMeGIS.com

Presentation Overview

Label Expressions
- Simple Expressions
- Functions
- Advanced Expressions
- ArcGIS Formatting Tags
- Display Expression

Query Expressions
- Simple Queries
- Combined Queries
- Wildcards
- NULL
- Functions
- Complex Queries

Calculator Expressions
- Simple Expressions
- Function Expressions
- Custom Functions
Label Expressions

Simple Expressions

Expression
- Single field
- Concatenate multiple fields
- Add static text
  - Units
  - New line
- VBScript or JScript

Single symbol for all labels
No change between ArcMap 9.3 and 10.
Single Field

Multiple Fields

TRICK
Use the Append button
Static Text

- Double quotes around text.
- Don’t forget a separator between the attributes.

```[OPERATOR] + " " + [TD] + " ft"
```

New Line

- VBScript: `vbNewLine`
- Jscript: `\n`

```[OPERATOR] + "\n" + [TD] + " ft"
```
### Functions

#### VBScript

<table>
<thead>
<tr>
<th></th>
<th>JScript</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper case string</td>
<td>UCase([field])</td>
</tr>
<tr>
<td></td>
<td>[field].toUpperCase()</td>
</tr>
<tr>
<td>Lower case string</td>
<td>LCase([field])</td>
</tr>
<tr>
<td></td>
<td>[field].toLowerCase()</td>
</tr>
<tr>
<td>Substring</td>
<td>Mid([field], start, length)</td>
</tr>
<tr>
<td></td>
<td>[field].substr(start, length)</td>
</tr>
<tr>
<td>Round</td>
<td>Round([field], decimals)</td>
</tr>
<tr>
<td></td>
<td>Math.round([field])</td>
</tr>
<tr>
<td>Format Date field</td>
<td>FormatDateTime([field], format_code)</td>
</tr>
<tr>
<td></td>
<td>[field].getMonth()</td>
</tr>
<tr>
<td></td>
<td>[field].getDate()</td>
</tr>
<tr>
<td></td>
<td>[field].getYear()</td>
</tr>
<tr>
<td>Format as currency</td>
<td>FormatCurrency([field])</td>
</tr>
<tr>
<td></td>
<td>No equivalent. Write advanced function.</td>
</tr>
<tr>
<td>Format as percentage</td>
<td>FormatPercent([field])</td>
</tr>
<tr>
<td></td>
<td>No equivalent. Write advanced function.</td>
</tr>
</tbody>
</table>
Label Expression Buttons

- Filter fields by data type
- Sort fields
- Append selected field
- List values in field
- Verify expression
- Help
- Save expression as *.lxp
- Load *.lxp

Advanced Expressions

- Define a complete function
  - Inputs are fields
  - Output is label string
  - Conditional statements
  - Loops

- Examples:
  - Stack a label at a comma in the text.
  - Label Percent Interest from multiple fields.
Function FindLabel ([OPERATOR])
myArray = Split([OPERATOR], ",")
strLabel = myArray(0)
For i = 1 To UBound(myArray)
strLabel = strLabel & "," & vbCrLf & myArray(i)
Next
FindLabel = strLabel
End Function
ArcGIS Formatting Tags

Use anywhere a text string and symbol are specified
- Label expressions
- Annotation
- Legend descriptions
- Graphic text

XML syntax rules
- Start and end tags
- Can be nested

Formatting Functions

<table>
<thead>
<tr>
<th>Function</th>
<th>Tag</th>
<th>Example</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; &gt;</td>
<td>&lt; &gt;</td>
<td>“&lt;” &amp; [STATUS] &amp; “&gt;”</td>
<td>&lt;/&gt; &lt;Not leased&gt;</td>
</tr>
<tr>
<td>Font</td>
<td>&lt;FNT&gt; &lt;/FNT&gt;</td>
<td>“&lt;FNT name='Comic Sans MS' size='14’” &amp; [API] &amp; ”&lt;/FNT&gt;”</td>
<td>94736283950</td>
</tr>
<tr>
<td>Color</td>
<td>&lt;CLR&gt; &lt;/CLR&gt;</td>
<td>“&lt;CLR red='255’, green='0’, blue='0’” &amp; [API] &amp; ”&lt;/CLR&gt;”</td>
<td>94736283950</td>
</tr>
<tr>
<td>Bold</td>
<td>&lt;BOL&gt; &lt;/BOL&gt;</td>
<td>“&lt;BOL&gt;” &amp; [API] &amp; “&lt;/BOL&gt;”</td>
<td>94736283950</td>
</tr>
<tr>
<td>Italic</td>
<td>&lt;ITA&gt; &lt;/ITA&gt;</td>
<td>“&lt;ITA&gt;” &amp; [API] &amp; “&lt;/ITA&gt;”</td>
<td>94736283950</td>
</tr>
<tr>
<td>Underline</td>
<td>&lt;UND&gt; &lt;/UND&gt;</td>
<td>“&lt;UND&gt;” &amp; [API] &amp; “&lt;/UND&gt;”</td>
<td>94736283950</td>
</tr>
</tbody>
</table>
Formatting Tag Help

Labels will be drawn using the symbol specified on the Labels tab (the tab you launched this Expression dialog from). You can modify or override the appearance of that symbol for particular portions of the expression by inserting ArcMap text formatting tags into the expression as text strings. This lets you create mixed-formatted labels where, for example, one field in a label is underlined. Here are the tags you can use:

Formatting Tags in a Legend

Webster-Union
- BORDERLY
- BORDLEY
- DIXON
- POOLE
- PROVIDENCE
- STURGIS
- WAVERLY
Display Expression

ArcGIS 10
  – Replaces Primary Display Field

Used for
  – MapTips
  – Identify
  – Attributes

Query Expressions
What is a SQL Query?

- SQL = Structured Query Language
- Asks a question of the data in the table
  - Queries are used to extract data from tables
- SQL where clause format:
  `<field_name> <operator> <value>`
- Examples:
  - “TOTAL_DEPTH” > 500
  - “API_NUM” = ‘89762535’

Where can a Query be Used?

- Anywhere ArcGIS talks to a table:
  - Select By Attribute
  - Field Calculator
  - Definition Query
  - Label Classes
  - Etc.
Syntax Variations

File Geodatabase (*.gdb), Shapefile, ArcSDE, dBASE table, ArcInfo coverage

- Queries are case sensitive
- '_' matches one character
- '%' matches any number of characters
- Enclose column names in double quotes: "AREA"
- Date format: date 'yyyy-mm-dd'

Personal Geodatabase (*.mdb)

- Queries are not case sensitive
- '?' matches one character
- '*' matches any number of characters
- Enclose column names in square brackets: [AREA]
- Date format: #mm-dd-yyyy HH:MM:SS#

Combined Queries

AND
"OPERATOR" = 'Sea Shell Gas' AND "TD" > 0 AND "TD" < 3000
- Returns values that match each statement.

OR
"FIELD_TYPE" = 'OIL' OR "FIELD_TYPE" = 'GAS' OR "FIELD_TYPE" = 'OIL&GAS'
- Returns values that match any of the statements.

PARENTHESES
("TYPE" = 'OIL' AND "WELL_DEPTH" > 750) OR
("TYPE" = 'GAS' AND "WELL_DEPTH" > 500)
- Sets the order in which the operators are used.
### Complex Query Operators

**IN**

```
"FIELD_TYPE" IN ('OIL', 'GAS', 'OIL&GAS')
```

- Reduces the query size by eliminating repetitive OR statements.

**BETWEEN**

```
"MODIFIED" between date '1979-01-01' AND date '1979-12-31'
```

- Reduces the query size by eliminating repetitive AND statements.

**NOT**

```
"XREFNO" NOT between '2400163' and '2400660'
```

- Performs the opposite query of any operator.

### NULL Values

Use **NULL** keyword to find or exclude records in the table that have not had an attribute set.

**Examples:**

```
"WELL_DEPTH" IS NULL
```

- Returns all wells in the table that have no value in the WELL_DEPTH column.

```
"WELL_DEPTH" IS NOT NULL
```

- Returns all wells in the table that have any value in the WELL_DEPTH column.
Wildcards

Use LIKE when querying with wildcards
- ‘?’ and ‘_’ match exactly 1 character
- ‘*’ and ‘%’ match any number of characters
- ‘#’ matches exactly 1 digit in a text string

Examples:

"TYPE" LIKE 'O?G'
✓ Matches ‘O&G’, ‘O+G’, ‘O_G’
✗ Does not match ‘Oil’, ‘OilAndGas’

"TYPE" LIKE ‘%Shell%
✓ Returns anything containing ‘Shell’

"TYPE" LIKE ‘4865#’
✓ Matches ‘48651’, ‘48652’
✗ Does not match ‘4865’, ‘4865a’, ‘486599’

Mathematic Functions

*, /, +, -

"VOLUME_YEAR"/365 > “MAX_RATE”*0.75
- Mathematic operators without parentheses will proceed in mathematic order, as displayed above (MDAS)
- Using + or – with date fields adds or subtracts days from that date

ROUND (<numeric_exp>, <int_exp>)
ROUND(“PRICE” * 1.0825, 2) BETWEEN ‘10’ AND ‘20’
- Rounds items to the specified number of decimal places.

TRUNCATE (<numeric_exp>, <int_exp>)
TRUNC(“POPULATION”, -3) = 15000
- Replaces the specified number of decimal places to a value of 0
- Negative numbers move to the left of the decimal
String Functions

Strings must always be surrounded by single quotes.

“STATE_NAME” = 'Texas'

Use **UPPER** or **LOWER** to avoid case sensitivity issues:

UPPER (“STATE_NAME”) = ‘TEXAS’
LOWER (“STATE_NAME”) = ‘texas’

Use >, <, >=, <= to retrieve strings based on sort order.

“STATE_NAME” <= ‘E’

• Returns all states starting with the letters A through D.

---

String Functions cont’d

**MID (<string>, <start>, <length>)**

MID (“REFNO”, 3, 2)

• Returns values from within a string.
• First character has a start value of 1.

**Left (<string>, <num_characters>)**

Right (<string>, <num_characters>)

Left (“API_NUM”, 2) < 20

• Extracts characters from left or right side of the string.
More Functions

**TRIM (<string>)**

TRIM ("API_NUM") = ‘1123400000’

- Removes leading spaces from a string
- Works on all file types

*.gdb, *.shp and *.dbf allow additional syntax to remove any leading/trailing character:

TRIM (leading ‘0’ from “API_NUM”) = ‘1234’

Date Functions

**CURRENT_DATE()**

“EXPIRATION_DATE” > CURRENT_DATE()

- Returns the system’s current date.

**EXTRACT (<keyword>) FROM (<extract_source>)**

EXTRACT (YEAR from "Expiration") >= 2010 and EXTRACT (YEAR from "Expiration") < 2015

- Returns the specified portion of the date.
- Keywords: YEAR, MONTH, DAY, HOUR, MINUTE, SECOND
Save and Load

Once a SQL expression is working, save it for future use.
- Saved as an Expression (*.exp) file.

Use Load to re-use a saved expression.
- Select by Attribute
- Definition Query
- Query Builder
- etc.

Calculator Expressions
Field Calculator

- Edit many records at once (in or out of an edit session).
- Right-click on the field to update → Field Calculator.

Field Calculator cont’d

- Runs on selected features
  - All if none selected
- Build a VB Script or Python expression
Field Calculator Help

- Use the Type options to see functions available for the different data types.
  - Number
  - String
  - Date
- Drop-down for field options.
- Getting help:
  - Help button
  - F1
  - Search online

Function Examples

- Concatenate latitude and longitude:
  - `[LON] & "", " & [LAT]`
  - `str(!LON!) + ", " + str(!LAT!)`

- Convert a string field to upper case:
  - `UCase([NAME])`
  - `!NAME!.upper()`

- Extract the state code from a STACOREFNO field (the first 2 digits are the state code):
  - `Left([STACOREFNO], 2)`
  - `!STACOREFNO![0:2]`

- Convert depth in feet to meters:
  - `[DEPTH_FT] * 0.3048`
  - `!DEPTH_FT! * 0.3048`

- Round numeric data to 2 decimal places:
  - `Round([ROYALTY], 2)`
  - `round( !ROYALTY!, 2)`
### Function Notes

#### VB Script
- Field names surrounded by `[]`
- `&` for concatenation (all data types)
- Case insensitive
- `<function_name> ([<field_name>])`

#### Python
- Field names surrounded by `!!`
- `+` for concatenation (must be strings)
- Case sensitive
- `!<field_name>!.<function_name>()`

---

### Field Calculator Tricks

Have you thought of using a field calculation to...
- Re-order columns
- Concatenate columns
- Change data type
- Convert units
- Create label expression
- Force values to upper case (for case-insensitive searches)
- Remove leading/trailing spaces
- Extract values from columns
- Create incrementing values
Field Calculator Syntax

**Concatenate columns**
- `str(Township!) + "-" + str(Section!)`

**Change data type**
- `!ExpireDate!`

**Convert units**
- `!ground_ele! * 0.3048`

**Change case**
- `!name!.upper()`

**Remove leading/trailing spaces**
- `!name!.strip()`

---

String Slicing

Slice returns a **substring**:

- 2 inputs (simple)
  - Start, end
- 3 inputs (extended)
  - Start, end, and stride

All inputs are 0-based.

Negative inputs count backwards from end.

Non-inclusive.

```
>>> str = "Hello world."
>>> str[0:5]
'Hello'
>>> str[2:-4]
'llo wo'
>>> str[2:8:2]
'low'
```

**TRICK**
Blank values = start, end
Custom Functions

Create incrementing values
Use FieldA, but if it’s NULL use FieldB
Change in values (percentage increase)
Random values
Maximum value from selected values

Incrementing Value

```python
rec=0
def autoIncrement():
    global rec
    pStart = 1 # adjust start value, if req'd
    pInterval = 1 # adjust interval value, if req'd
    if (rec == 0):
        rec = pStart
    else:
        rec = rec + pInterval
    return rec
```
# If the first field is NULL, use the second field

```python
def UseNonNullAttribute(fld1, fld2):
    if fld1:
        value = fld1
    else:
        value = fld2
    return value
```

Any questions?

Tracy.Rijken@TeachMeGIS.com