Client-Side Web GIS

Arcade

Image from NASA: https://commons.wikimedia.org/wiki/File:Earth_Western_Hemisphere_transparent_background.png#filelinks

This short module provides an introduction to the Arcade expression language. This language was created by ESRI and is used to build custom content across different components of the ESRI software ecosystem. We are discussion this here as opposed to earlier in the course and alongside the ArcGIS Online modules since the language is very similar to JavaScript. As a result, it is easier to understand once you have learned some basic JavaScript.

Module Topics

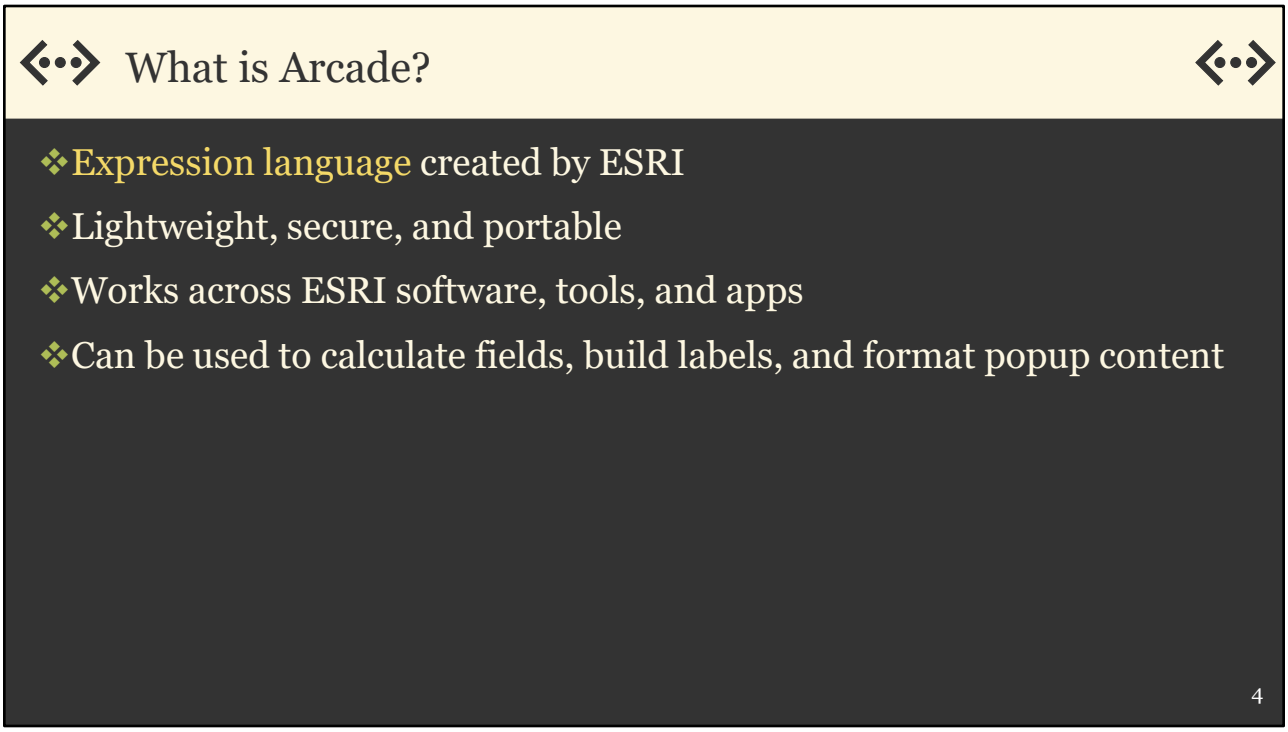
1. Uses of Arcade
2. Relation to JavaScript
3. Variables
4. Data types
5. Arithmetic, assignment, comparison, and logical operators
6. Profiles
7. Functions
8. Control flow
9. IIF
10. When
11. Decode
12. String formatting and concatenation

2

These are the topics covered in this module.

Preparation and Resources

3

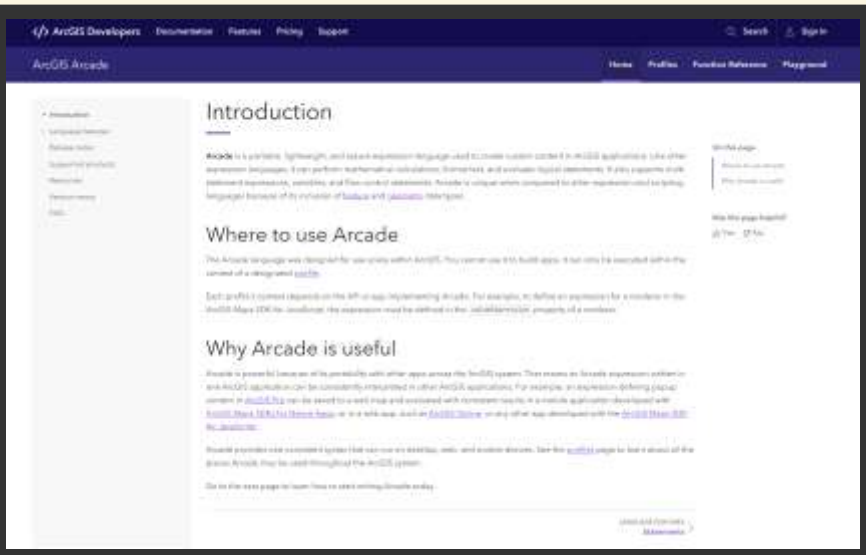


Arcade is primarily a language to build or define expressions. It is lightweight, secure, and portable. It works across the ESRI software ecosystem, including in the desktop, online, and mobile environments.

More specifically, it can be used to define popup content in ArcGIS Pro or ArcGIS Online. It can also be used to define labels in ArcGIS Pro and ArcGIS Online.

One of the benefits of using Arcade is that code generated in one environment, such as ArcGIS Pro, will be interpreted the same in other ESRI environments, such as ArcGIS Online.

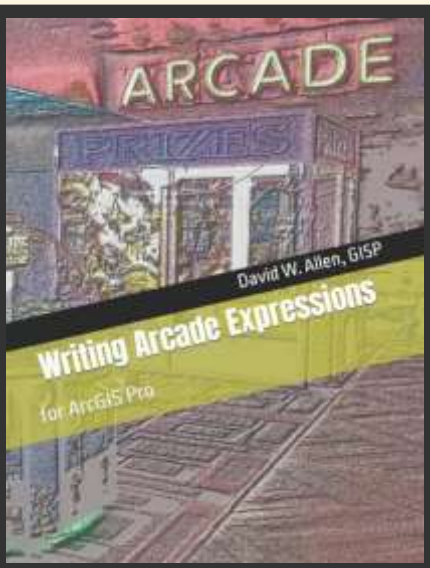
Documentation



https://developers.arcgis.com/arcade/

The page linked on this slide provides a reference and documentation for the Arcade language.

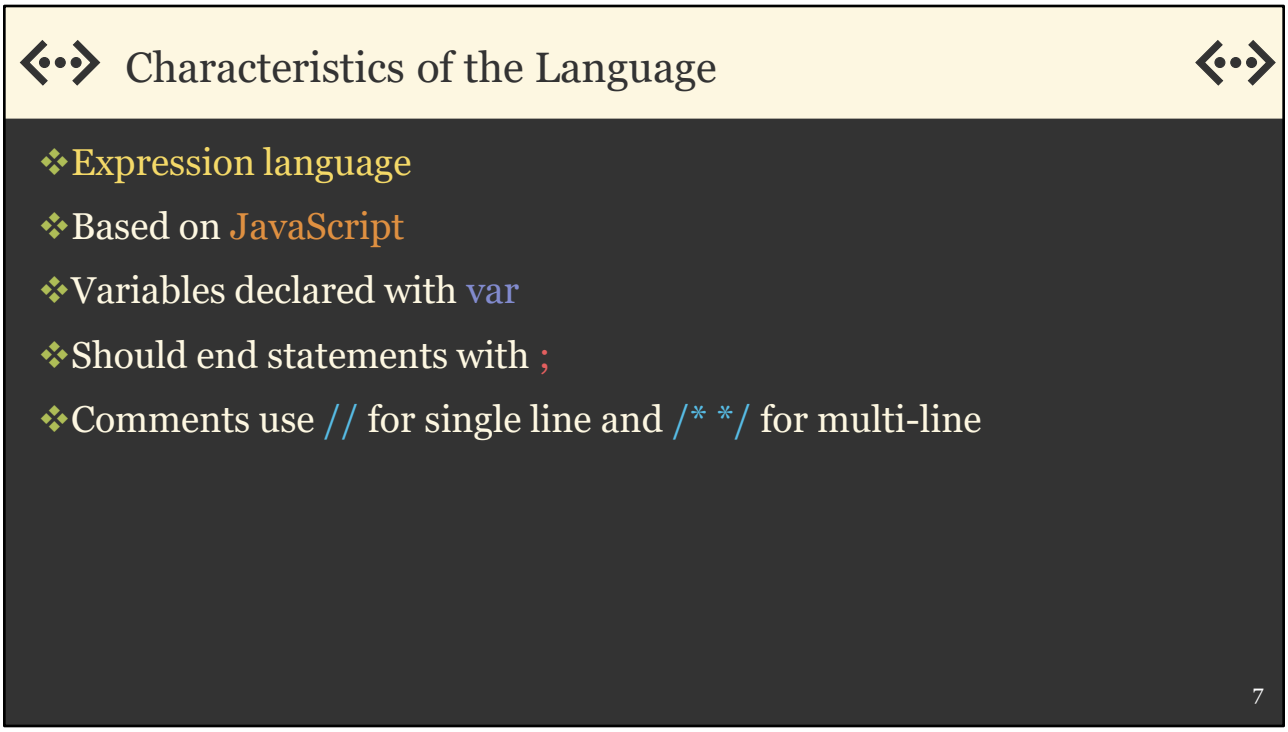
Text Recommendation



https://www.amazon.com/Writing-Arcade-Expressions-ArcGIS-Pro/dp/0578533006

6

This is a good, short text for getting started with the Arcade language.



Arcade expressions look a lot like JavaScript. So, if you are familiar with JavaScript, you should be able to pick up Arcade quickly.

If Arcade is so similar to JavaScript, then why not just use JavaScript? First, Arcade has been built specifically for building expressions and custom content within the ESRI software ecosystem, which is not the case for JavaScript. JavaScript is a more general language. Secondly, Arcade provides data types specific to geospatial data, such as features and geometry, which makes it easier to interact with geospatial data using Arcade as opposed to JavaScript.

This slide notes some of the similarities between Arcade and JavaScript. For example, variables are declared with the var keyword. Note that let and const are not used. It is generally considered good practice to end statements with a semicolon. However, this is not strictly required. Lastly, Arcade comments are defined the same as JavaScript comments.

Variables

- Names:
  - begins with a Latin letter, underscore, or $
  - contains only Latin letters, Latin numbers, underscore, or $
  - must not match any reserved words
- Variables may be reassigned to new values at any point in the expression
- Variables declared outside of a function have global scope
- Variables declared inside of a function have local scope
- Arcade does not use block scope

8

This slide outlines some other general characteristics of the Arcade language. Variable names must begin with a Latin letter, _, or $. Names can only contain Latin letters, _, or $. Reserved or keywords cannot be used as variable names.

Variables can be reassigned to new values in your code. Variables declared outside of a function have global scope while those declared in a function have local scope. Block scope is not implemented in Arcade.

Data Types

- Array = list of elements []
- Attachment = info about feature service attachments
- Boolean = logically true or false
- Date = dates
- Dictionary = collection of properties stored as name:value pairs
- Feature = geometry with dictionary of attributes
- FeatureSet = connection to layer in memory
- FeatureSetCollection = collection of FeatureSets
- Geometry = point, multipoint, polyline, polygon, extent
- Number = number treated as a number
- Portal = instance of ArcGIS Portal (ArcGIS Online or ArcGIS Enterprise Portal)
- Text = text or numbers treated as text

9

This slide briefly describes some of the different data types that are implemented in Arcade. Note again that special data types are available for geospatial data, which is one of the key benefits of using this expression language for building custom content within the ESRI software ecosystem.

We will not discuss all available data types in detail in this introductory module.

Arithmetic Operators

Operator	Name	Description	Example
+	Addition	Adds two numbers; can also concatenate two text values	5+5
-	Subtraction	Subtracts two numbers	5-5
*	Multiplication	Multiplies two numbers	5*5
/	Division	Divides two numbers	5/5
%	Modulus	Returns the remainder of a number divided by another number	10 % 2
++	Increment by one	Increments a number variable by one	x++ or ++x
--	Decrement by one	Decrements a number variable by one	--x or x--

https://developers.arcgis.com/arcade/guide/operators/

10

This slide highlights the arithmetic operators available in Arcade, which are equivalent to those used in JavaScript.

Assignment Operators

Operator	Name	Description	Example
=	Equals	Assigns a value to a variable	x = 10
+=	Add assign	Adds a number to the value of a number variable and assigns the result to the variable	x += 10
-=	Minus assign	Subtracts a number from a value of a number variable and assigns the result to the variable	x -= 10
*=	Multiply assign	Multiplies a number by the value of a number variable and assigns the result to the variable	x *= 10
/=	Divide assign	Divides the value of a number variable by another number and assigns the result to the variable	x /= 2

https://developers.arcgis.com/arcade/guide/operators/

11

This slide highlights the different variable assignment operators, which are also equivalent to those used in JavaScript.

Comparison Operators

Operator	Name	Description	Example
==	Equal to	Evaluates to true if the x-value is equal to the y-value	x==y
!=	Not equal to	Evaluates to true if the x-value is not equal to the y-value	x!=y
>	Greater than	Evaluates to true if the x-value is greater than the y-value	x>y
>=	Greater than or equal	Evaluates to true if the x-value is greater than or equal to the y-value	x>=y
<	Less than	Evaluates to true if the x-value is less than the y-value	x<y
<=	Less than equal to	Evaluates to true if the x-value is less than or equal to the y-value	x<=y

https://developers.arcgis.com/arcade/guide/operators/

12

The comparison operators are also identical. However, there are no operators available for testing for both equivalent value and type, such as === in JavaScript.

Logical operators

Operator	Name	Description	Example
||	Logical or	Returns true if one of two statements evaluates to true	if (x==9 || y==0){}
&&	Logical and	Returns true if both statements evaluate to true	if (x==9 && y==0){}
!	Logical not	Returns true if the statement does not return true	if(!didItWork) { console("No. It didn't work.") }

https://developers.arcgis.com/arcade/guide/operators/

13

Lastly, these are the logical operators used.

Profiles

- Labeling
- Popups
- Field calculation
- Visualization
- Many others

https://developers.arcgis.com/arcade/profiles/ 14

Profiles are a key concept in the Arcade language. One key consideration is that different input data are available in different profiles. Example profiles include alias, attribute rules, dashboard, dictionary renderer, field calculation, form calculation, labeling, layout, popups, and visualization.

Note that all profiles are not available in all software. For example, the dashboard profile is only available in ArcGIS Dashboards. Other profiles have wider uses. For example, the labeling profile is applicable to ArcGIS Pro, ArcGIS Maps SDKs for Native Apps, ArcGIS Maps SDK for JavaScript, ArcGIS Enterprise, and ArcGIS Online.

In this module, we primarily focus on using Arcade within the context of popups, labeling, and field calculation.

Functions

```
function abmult(a, b) {
  return a * b;
}
```

15

Arcade supports functions, control flow, and loops. Functions are defined using the same syntax as implemented in JavaScript.

Control flow

```
var x = 8;
if (x === 10) {
  console.log("Value is 10.");
} else if (x < 10) {
  console.log("Value is less than 10.");
} else {
  console.log("Value is larger than 10.");
}
```

16

Control flow is also implemented using the same syntax as JavaScript.

IIF

```
var x = IIF($feature.SUB_REG == 'N Eng', "New England", "Not New England");
return x;
```

17

Arcade implements the IIF() function, which can be used as an alternative to a control flow statement that only has if and else statements. This is similar to a conditional statement in Raster Calculator within ArcGIS Pro. The first argument is the condition being tested, the second is what to return if the condition evaluates to true, and the last is what to return if the condition evaluates to false.

In this example, if a features subregion is “N Eng”, then “New England” is returned. If it is any other value, “Not New England” is returned.

When

```
return When(
  $feature.SUB_REG == 'E N Cen', "East North Central",
  $feature.SUB_REG == 'E S Cen', "East South Central",
  $feature.SUB_REG == 'Mid Atl', "Mid-Atlantic",
  $feature.SUB_REG == 'Mtn', "Mountain",
  $feature.SUB_REG == 'N Eng', "New England",
  $feature.SUB_REG == 'Pacific', "Pacific",
  $feature.SUB_REG == 'S Atl', "South Atlantic",
  $feature.SUB_REG == 'W N Cen', "West North Central",
  $feature.SUB_REG == 'W S Cen', "West South Central",
  "Other"
);
```

18

When() can be used similar to a switch statement in JavaScript or to recode values. Here, I am specifying what text to return based on the text stored in an attribute column. The last argument is a default value to return.

Decode

```
return Decode($feature.SUB_REG,
 'E N Cen', "East North Central",
 'E S Cen', "East South Central",
 'Mid Atl', "Mid-Atlantic",
 'Mtn', "Mountain",
 'N Eng', "New England",
 'Pacific', "Pacific",
 'S Atl', "South Atlantic",
 'W N Cen', "West North Central",
 'W S Cen', "West South Central",
 "Other"
);
```

19

Decode() can also be used to recode values. Here, it is not necessary to provide a test within each statement. Instead, an input attribute field is defined followed by sets of input values and associated output values. The last argument is a default value to return.

For Loop

```
for
for(variable, condition, expression){
        Code to execute;
}
for...in
for(variable in array){
        Code to execute;
}
```

20

For loops are implemented the same in Arcade as they are in JavaScript.

However, in the case of iterating over elements in an array, the code is a bit different than JavaScript syntax.

Format a temperature value

- Round off value
- Convert from Celsius to Fahrenheit
- Add degree symbol and “F”

```
Round(($feature.tempmn*(9/5))+32, 1) + "\u00B0" + "F";
```

21

We will end this section with some example expressions that could be used to format text in a popup, labels, or to perform field calculations.

In this example, a temperature value in Celsius is being converted to Fahrenheit. Note the syntax to access the attribute feature: $feature.tempmn.

In the code, I am also rounding off the temperature to 1 decimal place using the Round() function, adding the degrees sign, and adding “F” to the end of the string.

Format currency

- Add dollar sign
- Convert to text
- Add comma separators
- Add zeros

```
"$" + Text($feature.med_ncm, '#,###') + ".00";
```

22

This is an example for formatting currency. The Text() function allows for formatting numeric data that is converted to text. The '#,###' argument indicates to add comma separators. I am also adding zeros to the end of the generated string.

Note that, similar to JavaScript, Arcade is capable of implementing type casting as needed. For example, data from a numeric attribute can be concatenated with strings without explicitly converting the data to strings.

Format percentages

- Round off value
- Add percent sign

```
Round($feature.per_for,1) + "%";
```

23

This example demonstrates formatting numeric data representing a percentage by rounding off the value and adding the percent sign.

Format text

- Replace portion of text in a string

```
Replace($feature.NAME, " County", "");
```

24

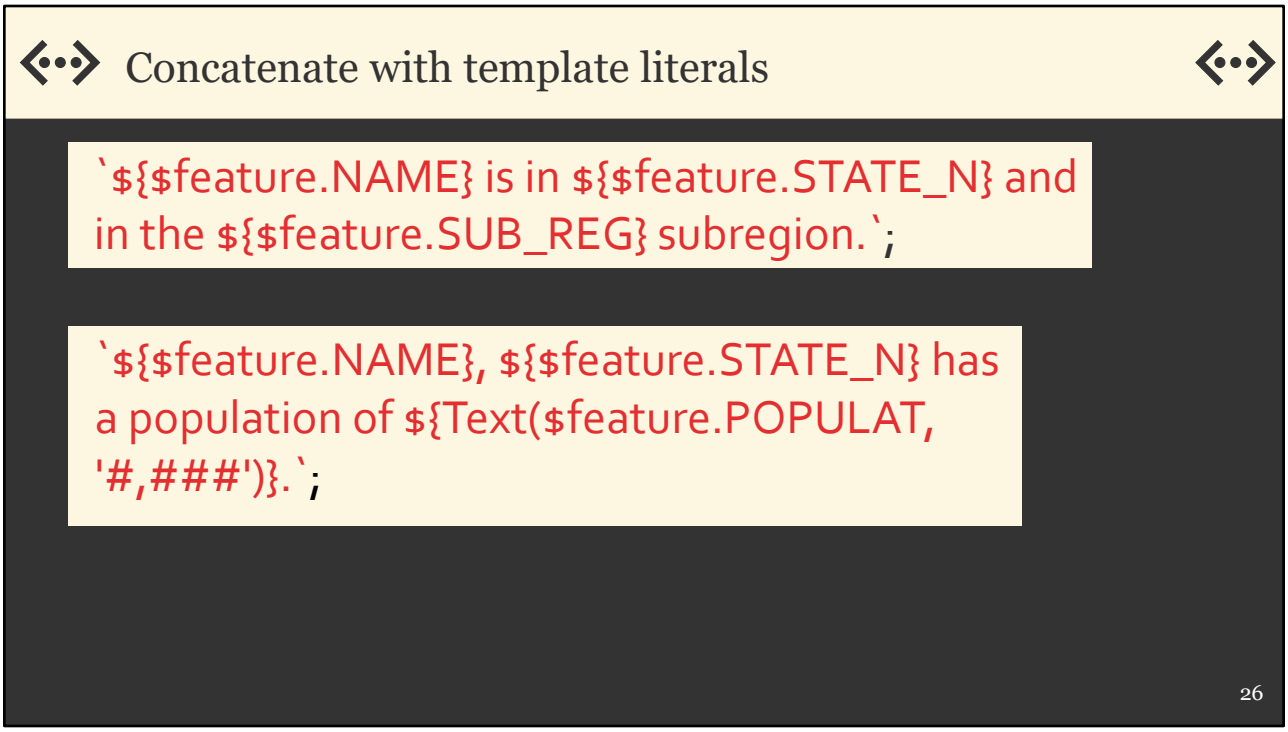
Here, I am using the Replace method to remove the “County” text from county names and replace it with an empty string.

Concatenate strings

```
$feature.NAME + " is in " + $feature.STATE_N + "
and in the " + $feature.SUB_REG + " subregion.";
```

25

This is an example of concatenating strings. This is accomplished using the + sign.



Strings can also be concatenated using template literals, similar to JavaScript.

Control Flow

```
if ($feature.med_ncm > 60000) {
        return "High income"
} else if ($feature.med_ncm <= 60000 && $feature.med_ncm > 40000) {
        return "Medium income"
} else {
        return "Low income"
};
```

27

This is an example of an expression that incorporates control flow. The expression will return either “High income”, “Medium income”, or “Low income” depending on the median income recorded in the “med_ncm” attribute.

This is the end of this lecture module.

Please return to the West Virginia View Webpage for additional content.

Thanks! Hope you found this useful.