An Introduction to ArcGIS Arcade
Tyson Quink
Topics
What we’re doing here…

- Overview of Arcade
  - What is Arcade
  - Why use Arcade

- Arcade Languages
  - Variables, Functions, Loops, Conditional Statements

- Arcade
  - Using Arcade
Arcade is an expression language that can be used across the ArcGIS Platform. Arcade provides a simple scripting syntax to deliver feature rendering and labeling.
Arcade

• Arcade is NOT intended to be a Full Programming / Scripting Language
  - Lightweight and Simple
  - Similar to spreadsheet calculations

• Embedded Expressions
  - Labeling, Rendering, Symbol Variations, …
  - Sharing

Not a replacement for geoprocessing and automation
Challenge
Accepted

Mobile Challenge
- Size of Executables
- Security
  - Native Access

Browser Challenge
- Page Size
- No Binaries
- Security
  - XSS

Scripting Challenge
Arcade Goals

- **Portable**
  - Write an expression in ArcGIS Pro and have it work across the platform

- **Secure**
  - Expression has no impact on security

- **Lightweight**
  - The language should be small and fast

- **Geospatial**
  - GIS data is the main use case

```javascript
var mapPoint = true;
if (mapPoint) {
    return "Star";
} else {
    return "Circle";
}
```
Arcade Language

What’s available
## Functions

### Data Functions
- Console
- Count
- Dictionary
- Distinct
- DomainCode
- DomainName
- Feature
- First
- Guid
- HasKey
- IndexOf
- Number
- Reverse
- Sort
- Text
- Top
- TypeOf

### Date Functions
- Date
- DateAdd
- DateDiff
- Millisecond
- Second
- Minute
- Hour
- Month
- Weekday
- Year
- Day
- Now
- Today
- Timestamp
- ToLocal
- ToUTC

### Geometry Functions
- Area
- AreaGeodetic
- Buffer
- BufferGeodetic
- Centroid
- Clip
- Contains
- Crosses
- Cut
- Difference
- Disjoint
- Distance
- Equals
- Extent
- Geometry
- Intersection
- Intersects
- Length
- LengthGeodetic
- MultiPartToSinglePart

### Logical Functions
- IsEmpty
- DefaultValue
- When
- Decode
- If
- Boolean

### Mathematical Functions
- Abs
- Acos
- Asin
- Atan
- Atan2
- Average
- Ceiling
- Constrain
- Cos
- Exp
- Floor
- Log
- Mean
- Min
- Max
- Pow
- Random
- Round
- Sin
- Sqrt
- Sdev
- Sum
- Tan
- Variance

### Text Functions
- Concatenate
- Find
- Lower
- Left
- Mid
- Proper
- Replace
- Right
- Split
- Trim
- Upper
Global Variables and Profiles

**Rendering**

Use expressions to provide values required for different types of renderers.

- Global Variables
  - $feature
  - $view

**Labeling**

Use expressions to provide values required for different types of renderers

- Global Variables
  - $feature
  - $view

**Future Profiles**
Language Features

• Designed for Simple Expressions
  - Single Line – Implicit Returns
  - Case insensitive Language
  - Dictionary Syntax for Field Access

• Has language features to support more complex scripts
  - Type system
  - Implicit and Explicit type casting
  - Flow control statements: if, for
  - Variable declaration and assignments
  - User defined functions

• Arcade scripts run and return a value. They are self contained and cannot alter their environment.
Case Insensitive

- Language is case insensitive
  - Quicker to author
  - Simpler for working with data and field names

`UpPeR('Hello World')`

`$feAtuRe.Field_nAmE`
Type System

- Simple Types
  - Numbers
  - Booleans
  - Dates
  - Strings

- Object Types
  - Dictionary
  - Feature
  - Array
  - Point
  - Line
  - Polygon
  - Multipoint

```javascript
var myNumber = 10;
var myText = "Hello";
var myDate = Date(2015,1,1);
var myBool = true;
var myDictionary = {"key1": 10};
var myFeatures = Feature({"geometry":{...},
                         "attributes":{"key1": 10}});
var myArray = [1, 2, 3];
var myPoint = Point({...});
```

Dates, Dictionary, Feature, Point, Line….all have overloaded constructors.
Implicit and Explicit Type Casting

**Implicit Casting**
- For Function Parameters
- For Expressions

**Explicit Casting**
- Functions cast between types
- Number, Text, Date, Boolean

**Implicit Casting:**
```
return 10 + "Star"
```

**Explicit Casting:**
```
return text(10) + "Star"
```
if Statement

- if statements are supported
- Simple single line expressions
  - IIF
  - Decode
  - When

Traditional if Statement:
```java
if (mapPoint){
    return 'Star';
} else {
    return 'Circle';
}
```

Equivalent:
```java
IIF(mapPoint, 'Star', 'Circle');
```
for Statement

- for Statements
  - Same syntax as JavaScript

- for in Statements
  - Iterates over indices of an array, or field names of a Dictionary or Feature

- Break, Continue, Return Statements supported inside block

```javascript
for(var i=1; i<100; i++) {
  if (i==3) continue;
  if (i==5) break;
  n+=i;
}

for(var k in myArray) {
  n+=myArray[k];
}
```
User Defined Functions

- Function keyword used to declare functions
  - Must be declared ahead of use
  - Variables defined inside function have local scope
  - Functions cannot be declared inside of functions

```javascript
function MultiplyNumber(number) {
    return number * 50;
}

MultiplyNumber(10);
```
Variable Assignments

- Variables can have their values reassigned
  - Objects types may be immutable if passed into Arcade
  - Arrays are not Sparse. They must be sequential
  - Geometry types are immutable inside. You cannot change the points in a path of a line.

```cpp
++myNumber
i = i + 1;
myArray[10] = 11;
myFeature.Field1 = 'arcgis';
```
The visualization profile allows the map author to write an expression that evaluates to a value used to drive the visualization.

```javascript
var dem = $feature.votes_dem;
var rep = $feature.votes_gop;
if (dem > rep) {
  return "Democrat";
} else {
  return "Republican";
}
```
Since Arcade expressions execute for each feature, using multiple geometry operations within the context of the labeling and visualization profiles can be expensive and severely impact the performance of the application. Also note that geometries fetched from feature services, especially polylines and polygons, are generalized according to the view's scale resolution. Be aware that using a feature's geometry (i.e. `Geometry($feature)`) as input to this function will yield results only as precise as the view scale. Therefore, results returned from geometry operations in the visualization and labeling profiles may be different at each scale level. Use these functions at your discretion within these contexts.
var json = {"x": -77.009268, "y": 38.889750, "spatialReference": {"wkid": 4326}};
var source_point = Point(json);
var feature_point = Geometry($feature);
var dist = Distance(feature_point, source_point, 'meters');
return dist;
Labeling

When the label is to be drawn, the script will be evaluated. It will be passed the feature that is about to be labeled. It is expected that the script returns a text string, comprising the label to be drawn.

```javascript
var windSpeed = $feature.WIND_SPEED;
var temp = $feature.TEMP;
var windChill = Round(35.74 + (0.6215*temp) - 35.75*POW(windSpeed,0.16) + 0.4275*temp*POW(windSpeed,0.16), 2);

if (windChill <= 32) {
    return '<CLR blue = "255">Wind Chill: ' + windChill + '°</CLR>';
} else if (windChill > 32 && windChill <= 55) {
    return '<CLR yellow = "255">Wind Chill: ' + windChill + '°</CLR>';
} else {
    return '<CLR red = "255">Wind Chill: ' + windChill + '°</CLR>';
}
```
// return the value to the map, to be mapped
IIf(total > 0 && domAttainment == "Irish", domAttainment, null);
JavaScript
Using Arcade

```javascript
<script type="text/plain" id="wind-direction">
  // WIND DIRECTION

  var DEG = $feature.WIND_DIRECT;
  var SPEED = $feature.WIND_SPEED;
  var DIR = When( SPEED == 0, null,
                  (DEG < 22.5 && DEG >= 0) || DEG > 337.5, 'N',
                  DEG >= 22.5 && DEG < 67.5, 'NE',
                  DEG >= 67.5 && DEG < 112.5, 'E',
                  DEG >= 112.5 && DEG < 157.5, 'SE',
                  DEG >= 157.5 && DEG < 202.5, 'S',
                  DEG >= 202.5 && DEG < 247.5, 'SW',
                  DEG >= 247.5 && DEG < 292.5, 'W',
                  DEG >= 292.5 && DEG < 337.5, 'NW', null);

  var WIND = SPEED + ' mph ' + DIR;
  return WIND;
</script>

<script>
  var windArcade = document.getElementById("wind-direction").text;

  var windClass = new LabelClass({
    labelExpressionInfo: {
      expression: windArcade
    },
    labelPlacement: "above-right",
    minScale: 2500000
  });

  windClass.symbol = createTextSymbol("#3ba53f", 11, { x: 3, y: 3 });
</script>
```
GeoAnalytics
Arcade in analysis

- Create Buffers
- Reconstruct Tracks
- Join Features
- Detect Incidents
- Calculate Field
Arcade Demonstrations

ArcGIS Pro and Web Maps
Resources

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

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

Using Arcade in JS API

Using Arcade in GeoAnalytics Server
Print Your Certificate of Attendance
Print stations located in the 140 Concourse

Tuesday
12:30 pm - 6:30 pm
GIS Solutions Expo
Hall B

5:00 pm - 6:30 pm
GIS Solutions Expo Social
Hall B

Wednesday
10:30 am - 5:15 pm
GIS Solutions Expo
Hall B

6:30 pm - 9:00 pm
Networking Reception
Smithsonian National Portrait Gallery
Please Take Our Survey in the Esri Events App

Download the Esri Events app and find your event

Select the session you attended

Scroll down to find the feedback section

Complete answers and select “Submit”