Imagery in a WebGIS

Wenxue Ju & Naila Khan
What’s in store for today’s session…

- Introduction to ArcGIS Image Services
- Capabilities of Image Services in ArcGIS Online and Portal for ArcGIS
- Imagery support in Web AppBuilder for ArcGIS
- Creating rich applications using ArcGIS API for JavaScript
- All new ArcGIS API for JavaScript 4.0
- Road Ahead
- Summary
- Q&A
Introduction to Image Services
**Image Services: Data + Visualization + Analysis**

- An image service provides access to raster data through a web service
- Raster data and its processing capabilities can be shared through ArcGIS Server and Portal
- Can be published with WMS and WCS capabilities

- Supported by many clients
  - Desktop, mobile, Web Clients, REST etc.
Image services in ArcGIS Online
Image Service: Capabilities

- Visualization
  - Fast dynamic display

- Image catalog
  - Identify
  - Query

- Processing
  - Well-known server-side raster functions
  - Register raster models with image service
  - Client-side processing
Image services in Web AppBuilder for ArcGIS
Web AppBuilder for ArcGIS

- Enables new apps to be created without coding
- Interactive user experience
- Runs on any device; in a web browser
- Built using ArcGIS API for JavaScript and HTML 5 technology
- Extensible
- Fully integrated with the ArcGIS Platform
Web AppBuilder: Available Imagery widgets

- Layer List
- Legend
- Query
- Attribute Table
- Popups
- Image Measurement*
- Oblique Viewer*

* Specifically designed to work with image services
Demo
Application built using Web AppBuilder for ArcGIS
Imagery Custom Widgets: [https://github.com/Esri/WAB-Image-Services-Widgets](https://github.com/Esri/WAB-Image-Services-Widgets)

- Designed to work specifically with image services
- Live apps
- Download and unzip the file
- Follow instructions in the ‘WEBAPP BUILDER USER DOC.docx’
- Renderer, Time Filter, Swipe, Change Detection, Image Classification and Spectral Profile

View live app here with IS Mensuration and IS Classification widgets
Unlock Earth’s Secrets:
Image services in
ArcGIS API for JavaScript
ArcGIS API for JavaScript: Image Service Layers

- **ArcGISTiledMapServiceLayer**
  - Renders image data, retrieves cached image tiles

- **ArcGISImageServiceLayer**
  - Renders image data, retrieves data as an image (PNG, JPEG, JPGPNG, etc.)

- **RasterLayer**
  - Renders image data, retrieves data as pixels (PNG, JPEG, TIFF, LERC etc.), support client side pixel filtering

- **ArcGISImageServiceVectorLayer**
  - Renders image data as vectors, retrieves data as pixels, supports scientific data, define symbology

*esri/layers/*
Image Service Layers: Constructor

```javascript
var imageLayer = new ArcGISImageServiceLayer(url, options)
                        = new RasterLayer(url, options)
                        = new ArcGISImageServiceVectorLayer(url, options)

- options:
  - opacity
  - visible
  - useMapTime
  - ImageServiceParameters (format, bandIds, compressionQuality, interpolation, noData, timeExtent)
```
Image Service Layers: Common Properties and Methods

**Raster Dataset**
- Pixel Size
- Pixel Type
- Rendering Rule
- Raster Attribute Table
- KeyProperties
- Identify

**Mosaic Dataset (MD)**
- +
  - MosaicRule
  - VisibleRasters
  - Query

**MultiDimensional MD**
- +
  - MultiDimensional Info
  - Dimensional Definition
Image Service Layers: Webmap and Widgets

- **WebMap**: (query definition, display order, server template, popup, etc.)
- **Widgets**: Legend, Popup, ImageServiceMeasure, ObliqueViewer, MultidimensionalFilter, MultidimensionalSlider, mosaicRule, renderingRule
Demo

Custom application built using ArcGIS API for JavaScript
Processing & Visualization: Client Side Pixel Filter

HTML ArcGISImageServiceLayer VS RasterLayer

- **IMG:** `<img src="image.jpeg"/>`
  
  Can only handle jpeg, png, gif (http url or inline base64)

- **CANVAS:** `<canvas/>`
  
  Puts in RGBA array → data manipulation in JS
  
  `ctx.putImageData(imagedata, dx, dy)`

Processing & Visualization: Client Side Pixel Filter

LERC -- Limited Error Raster Compression

- Request raster data from server (10.3+)
- Decode raster data (typed arrays, JS API 3.13+)
- Manipulate pixels
- Put RGBA data to canvas

On Github (Apache 2.0 License)

- https://github.com/Esri/lerc
- http://esri.github.io/lerc/js/
Processing & Visualization: Client Side Pixel Filter

**PixelBlock** (esri/layers/PixelBlock)

```javascript
{
  width: 1200,
  height: 800,
  pixels: [[32.5, 82.9, 25.2, ...], [], [], ...],
  pixelType: "F32",
  statistics: [{minValue:-50.8, maxValue:80.5}, {}, {}, ...],
  mask : [1,0,1,1,0,...]
}
```

*Pixels array structure is BSQ*, representing pixels in image space. Each element is an array representing a band, pixels are organized row by row. For each row, it goes from the first column to last.
Processing & Visualization: Client Side Pixel Filter

**Raster (esri/layers/Raster)**

- Makes exportImage request to image service
- Calls decoders to decode the returned data
  --- Supports LERC, BIL, BSQ, JPG, PNG format
- Stores data in PixelBlock
Processing & Visualization: Client Side Pixel Filter

RasterLayer (esri/layers/RasterLayer)

- Modern browsers
- Support your own pixel filters
- WebGL ok (IE11, Firefox, Chrome, Safari)

```javascript
var isRasterLayer = new RasterLayer(isUrl, { opacity: 1, pixelFilter: maskPixels });

function maskPixels(pixelData) {
    var pixelBlock = pixelData.pixelBlock;
    var pixels = pixelBlock.pixels;
    var band1 = pixels[0];
    var mask = pixelBlock.mask;
    var numPixels = pixelBlock.width * pixelBlock.height;
    for (var i = 0; i < numPixels; i++) {
        mask[i] = (band1[i] >= Math.floor(currentMin) &&
                    band1[i] <= Math.floor(currentMax)) ? 1 : 0;
    }
}
```
Demo

Client Side Pixel Filters
Processing & Visualization: Server Side Raster Functions

Algorithms ready with ArcGIS Server
Processing & Visualization: Server Side Raster Functions
Chaining, customizing

- Templating: raster function editor
- Extending: python or ArcObjects
Processing & Visualization: Server Side Raster Functions

Raster Functions (esri/layers/RasterFunction)

```javascript
var rf = new RasterFunction();
rf.functionName = "Remap";
rf.functionArguments = {
    "InputRanges" : [0,100,101,200],
    "OutputValues" : [5,125],
    "ZFactor" : 0.3,
    "Raster" : <another_raster_function_object_if_chaining_or_do_not_set_this_property>,
};
rasterLayer.setRenderingRule(rf);

//Can be used on ArcGISImageServiceLayer and RasterLayer.
```
Demo
Server Side Raster Functions
Vectorization and Charting

- **PixelBlock**
  Client decides what to do with all pixel values
- **identify / getSamples**
  Ask server for pixel values at multiple locations: profile, wind rose
- **computeStatisticsHistograms**
  Ask server for histograms of given geometry: column chart
Vectorization and Charting: Vector Field Data

**ArcGISImageServiceVectorLayer** (esri/layers/ ArcGISImageServiceVectorLayer)

- **Draws raster data as vectors using predefined symbology**
  
  *Single Arrow, Simple Scalar, Wind Barbs, Beaufort Wind, Ocean Current*

- **Decodes the returned data**

- **Visualize wind, Ocean current**

- **Default pixelFilter: computeMagnitudeAndDirection**

```javascript
var isVectorLayer = new ArcGISImageServiceVectorLayer(isUrl, {
  imageServiceParameters: params,
  symbolTileSize: 60,
  rendererStyle: "single_arrow"
});

var mr = new MosaicRule();
mr.multidimensionalDefinition = [];
mr.multidimensionalDefinition.push(new DimensionalDefinition({
  variableName:"Salinity",dimensionName:"StdZ", values:[-20]})�
```
Demo
Vectorization & Charting
Image services in

ArcGIS API for JavaScript 4.0
ArcGIS API for JavaScript 4.0: ImageryLayer

- Renders image data
- Retrieves data as pixels
- Supports LERC, BIL, BSQ, JPG, PNG formats
- View Popups
- Query
- Load webmap
- Apply client side processing using PixelFilters
- Define server side processing using raster functions
- Multi-dimensional API
- Supports 2D & 3D visualization
ArcGIS API for JavaScript 4.0: ImageryLayer

```javascript
var layer = new ImageryLayer({
  url: "https://myServer.arcgisonline.com/arcgis/rest/services/NLCDLandCover/ImageServer"
});

/**********************************
 * Add image layer to map
 **********************************/

var map = new Map({
  basemap: "gray",
  layers: [layer]
});

var view = new MapView({
  container: "viewDiv",
  map: map,
  center: [-100, 40],
  zoom: 5
});
```
Demo
Visualization and processing in 2D & 3D
Imagery in a WebGIS

Road Ahead...
Road Ahead: ArcGIS Online and Portal for ArcGIS

• ArcGIS Online: Enhancements in the Image Display UX

• Portal For ArcGIS: Raster Analytics
  - Execute spatial analysis and raster models which leverages distributed storage and analytics
  - Creates persisted products and optimizes your data for distributed analytics
  - Works with your existing GIS data and imagery and is designed to scale with your organization’s demands.
Road Ahead: ArcGIS API for JavaScript

• 3.x:
  - WCS layer
  - Raster Analytics widgets
  - More client-side processing algorithms

• 4.x:
  - Visualizing pixels as vectors (ArcGISImageServiceVectorLayer)
  - Improve performance of ImageryLayer by supporting tiled request
  - Imagery specific widgets
Documentation Resources

• ArcGIS API for JavaScript:  
  https://developers.arcgis.com/javascript/3/

• ArcGIS API for JavaScript 4.0  
  https://developers.arcgis.com/javascript

• Web AppBuilder Online help documentation  

• Web AppBuilder Developer Edition help documentation  
  https://developers.arcgis.com/web-appbuilder/guide/xt-welcome.htm
Summary

- Capabilities of image services
- Support of image services in applications like ArcGIS Online map viewer and Web AppBuilder
- ArcGIS API for JavaScript:
  - Different layer types
  - Widgets
  - Client-side processing using pixel filters
  - Server-side processing
  - Multi-dimensional API
- ArcGIS API for JavaScript 4.0:
  - Imagery layer and it’s capabilities
  - 2D and 3D support
Thank you…

- Please fill out the session survey in your mobile app
- Search and select the technical workshop ‘Imagery in a WebGIS’
- Go to the feedback section and answer a few short questions

Q & A