3D Visualization with the ArcGIS API for JavaScript

Kristian Ekenes, Jeremy Bartley, Johannes Schmid

Consistency

Principles of 2D viz ≈ 3D viz

- Code is similar
- Cartographic principles/techniques are similar

Overview

- What you can visualize in 3D
- How to do it
- Considerations and pitfalls

Overview

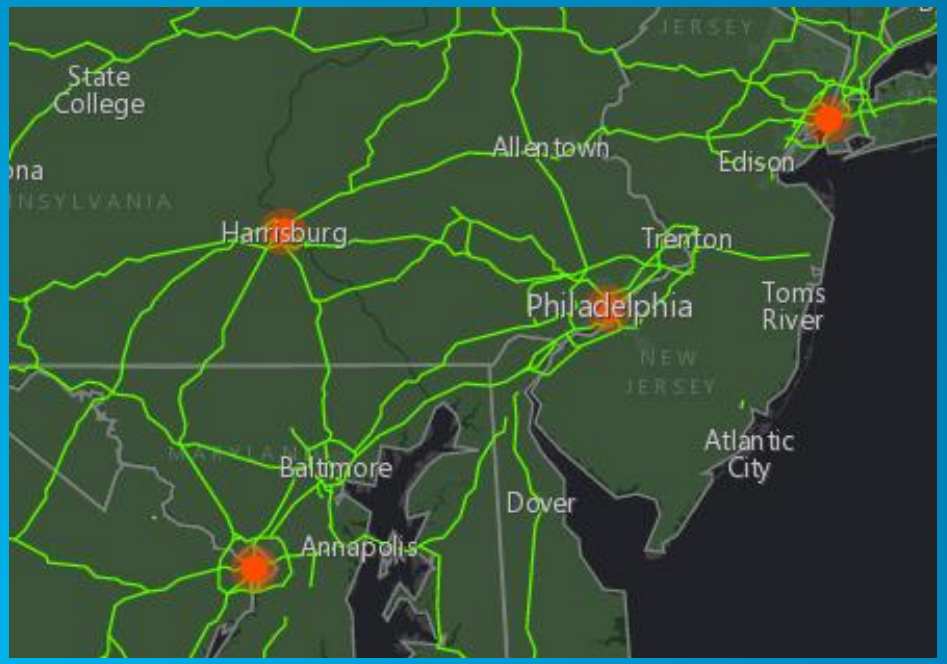
- What you can visualize in 3D
- How to do it
- Considerations and pitfalls

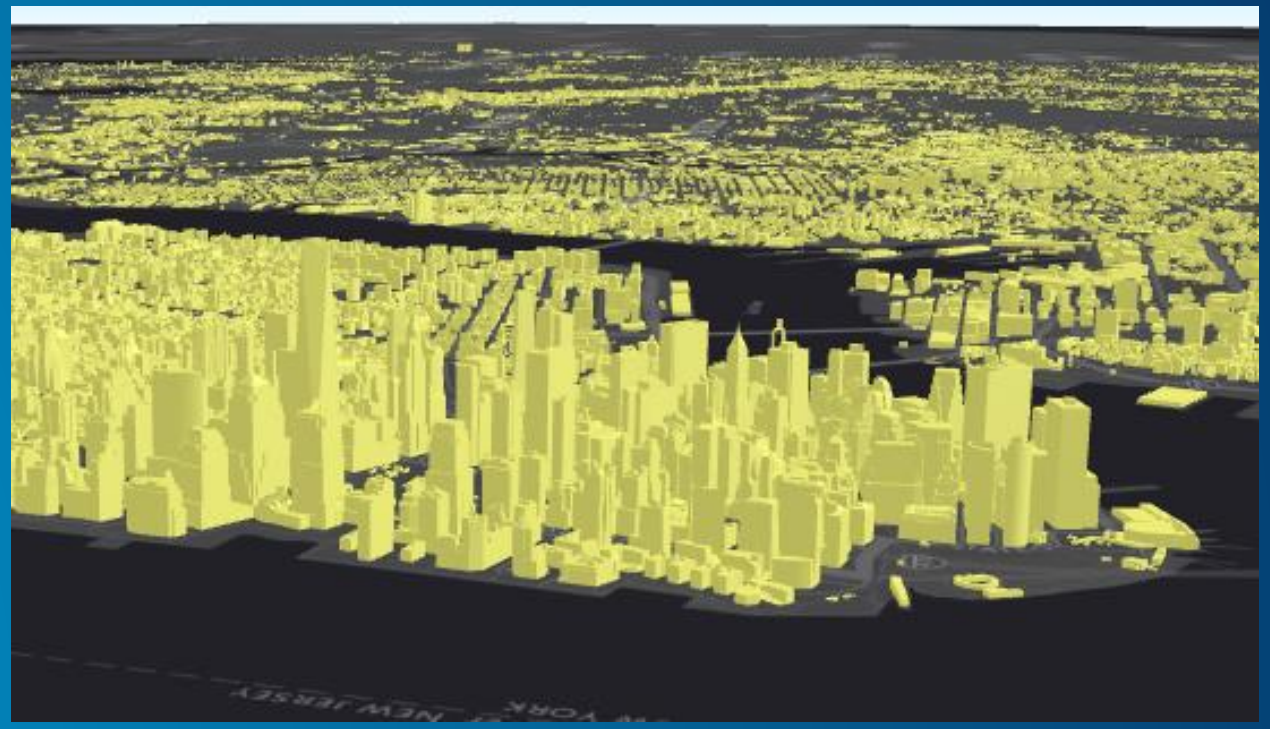
What can we visualize?

- Location
- Types
- Data (numbers)
- Combinations of the above

Location

All features in a layer have same symbol

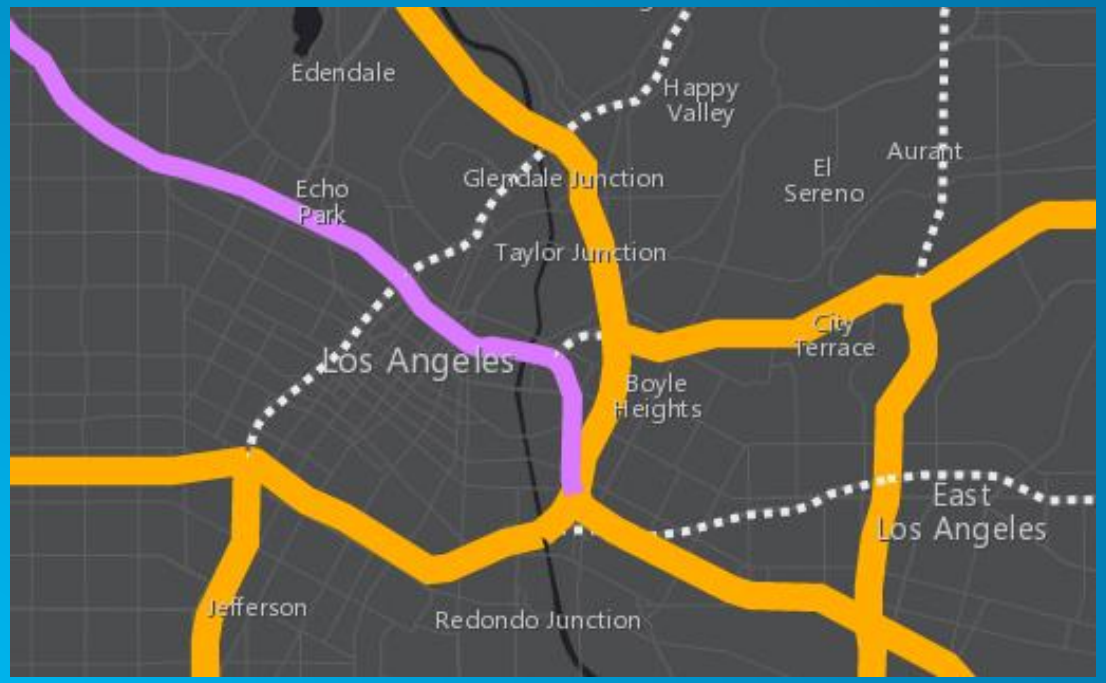




Types

Based on unique (usually text) values

Interstates, highways, major roads, …



Residential, commercial, mixed, …

Numbers

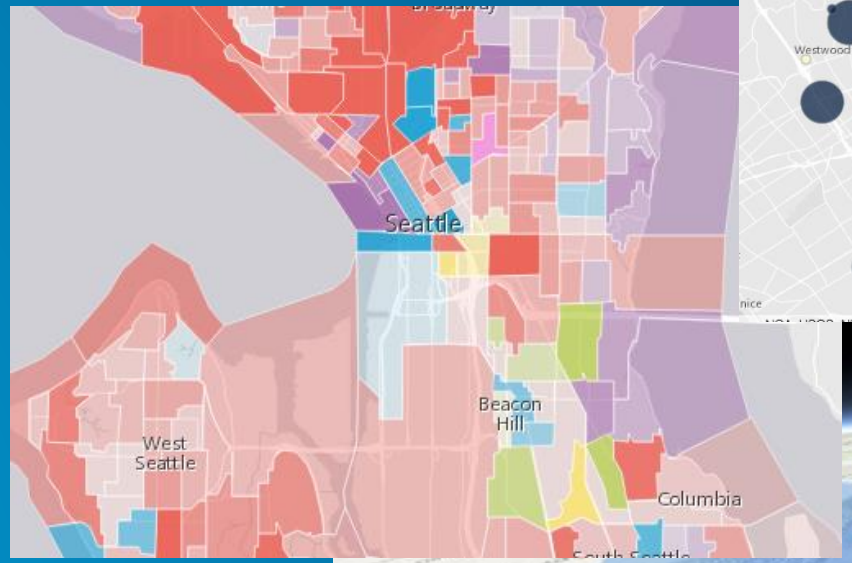
Based on field values or functions

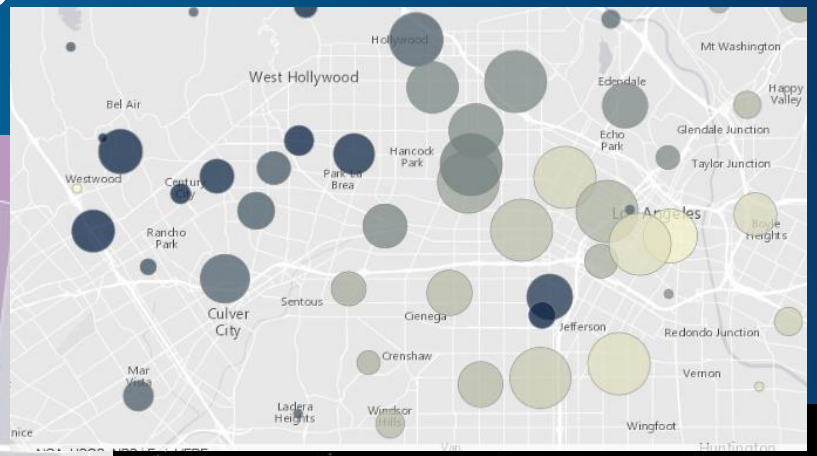
- Color
- Size/Extrusion
- Opacity

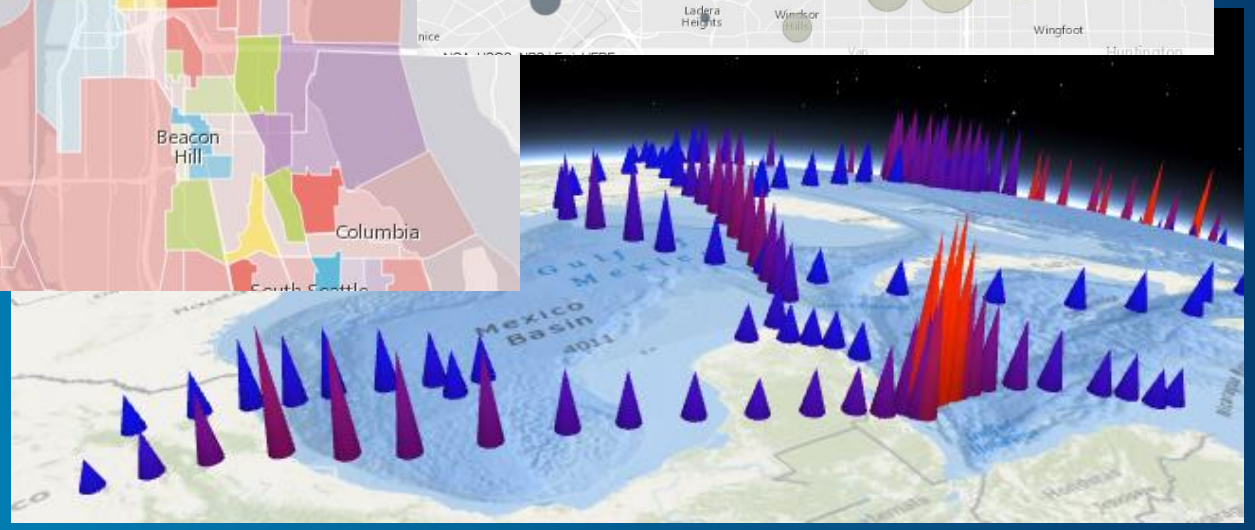
Multivariate viz

Based two or more field values

- Color + Opacity
- Color + Size
- Size + Opacity
- Size + Size
  - (Height + width)







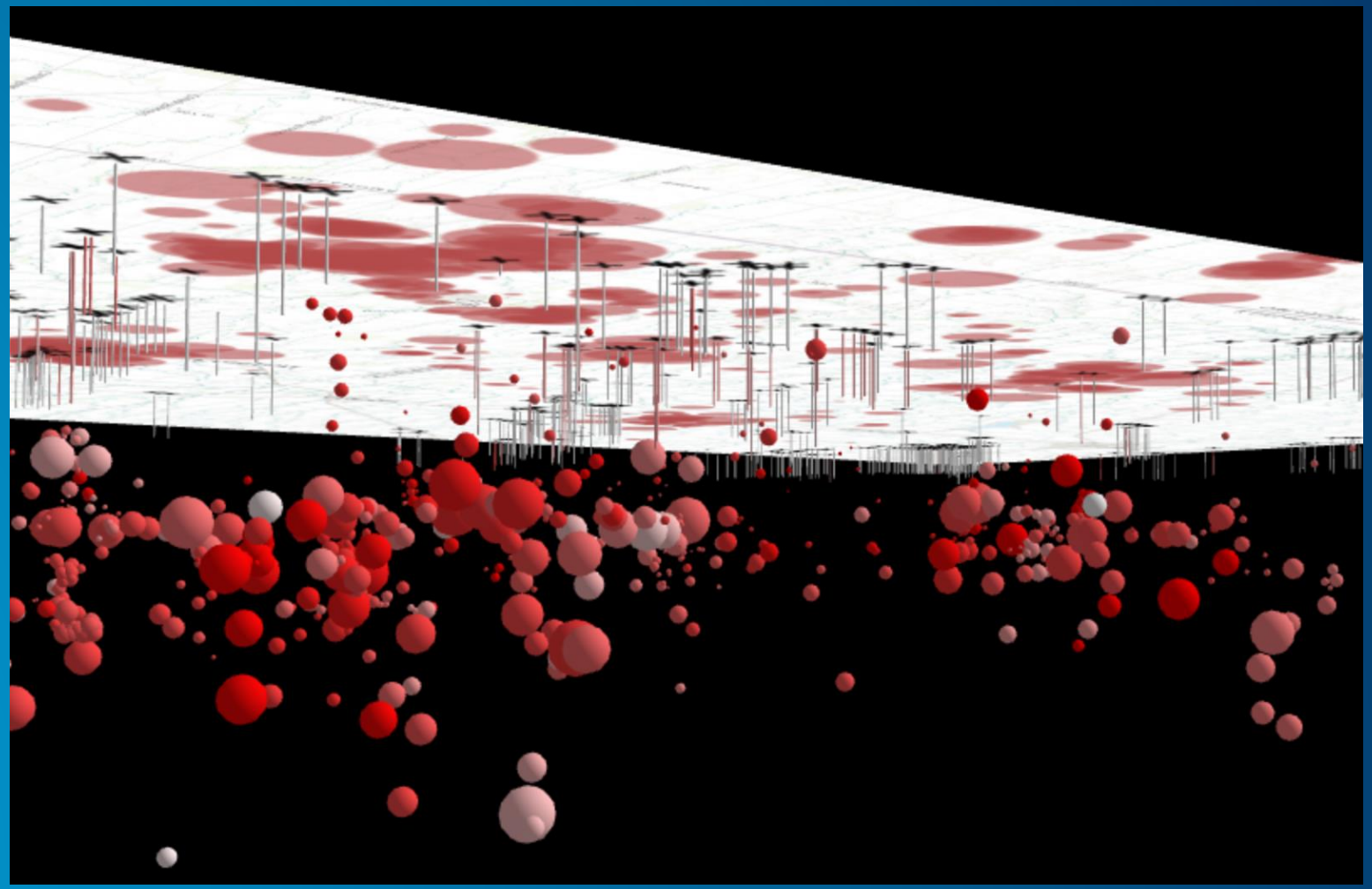
Hybrid thematic maps

- Size and shape based on real-world measurements
- Color or opacity driven by thematic data

Simplifies or
removes the need for a legend!

Subsurface mapping

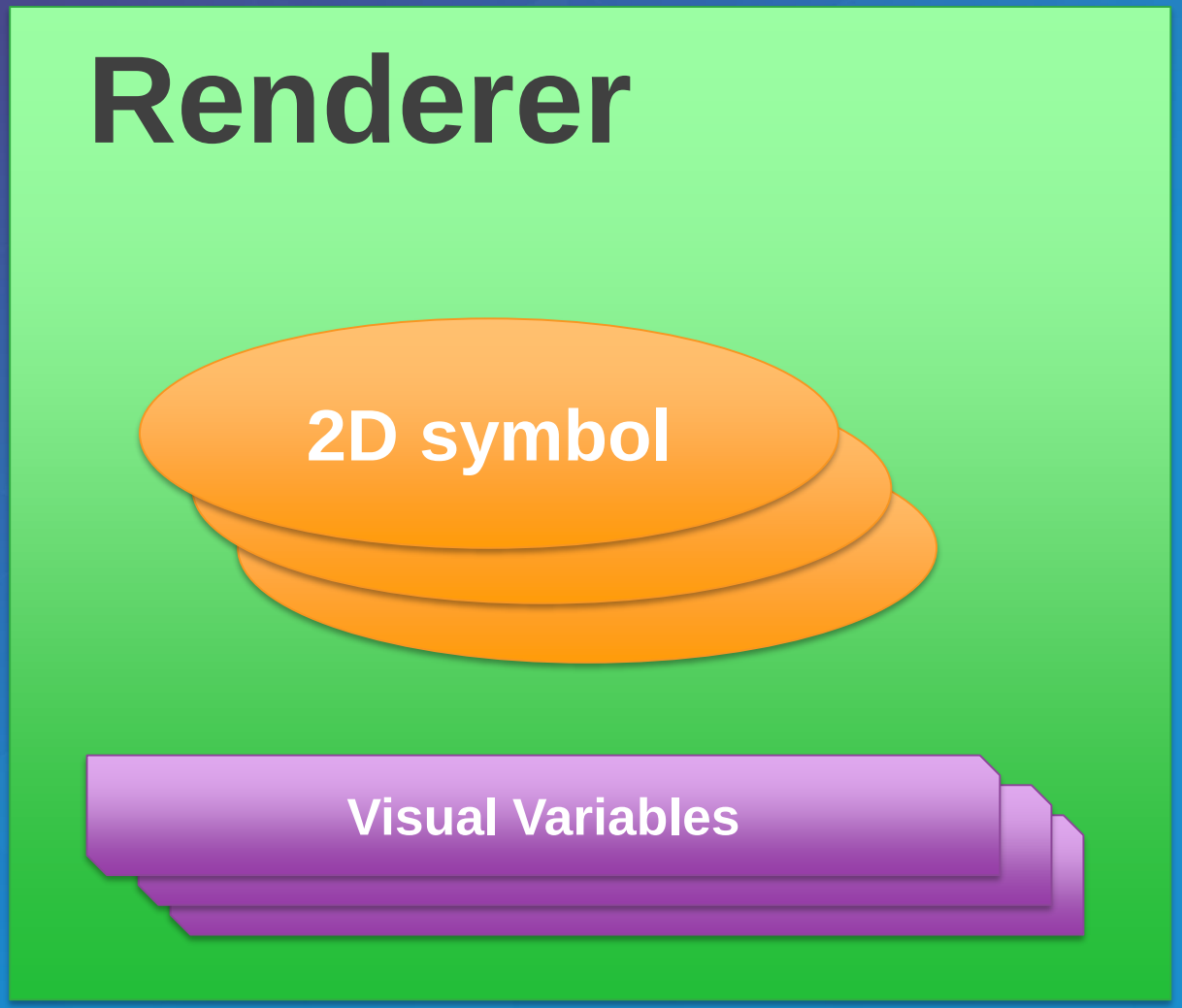
- Features with negative z values
- Negative extrusions
- Only in local scenes

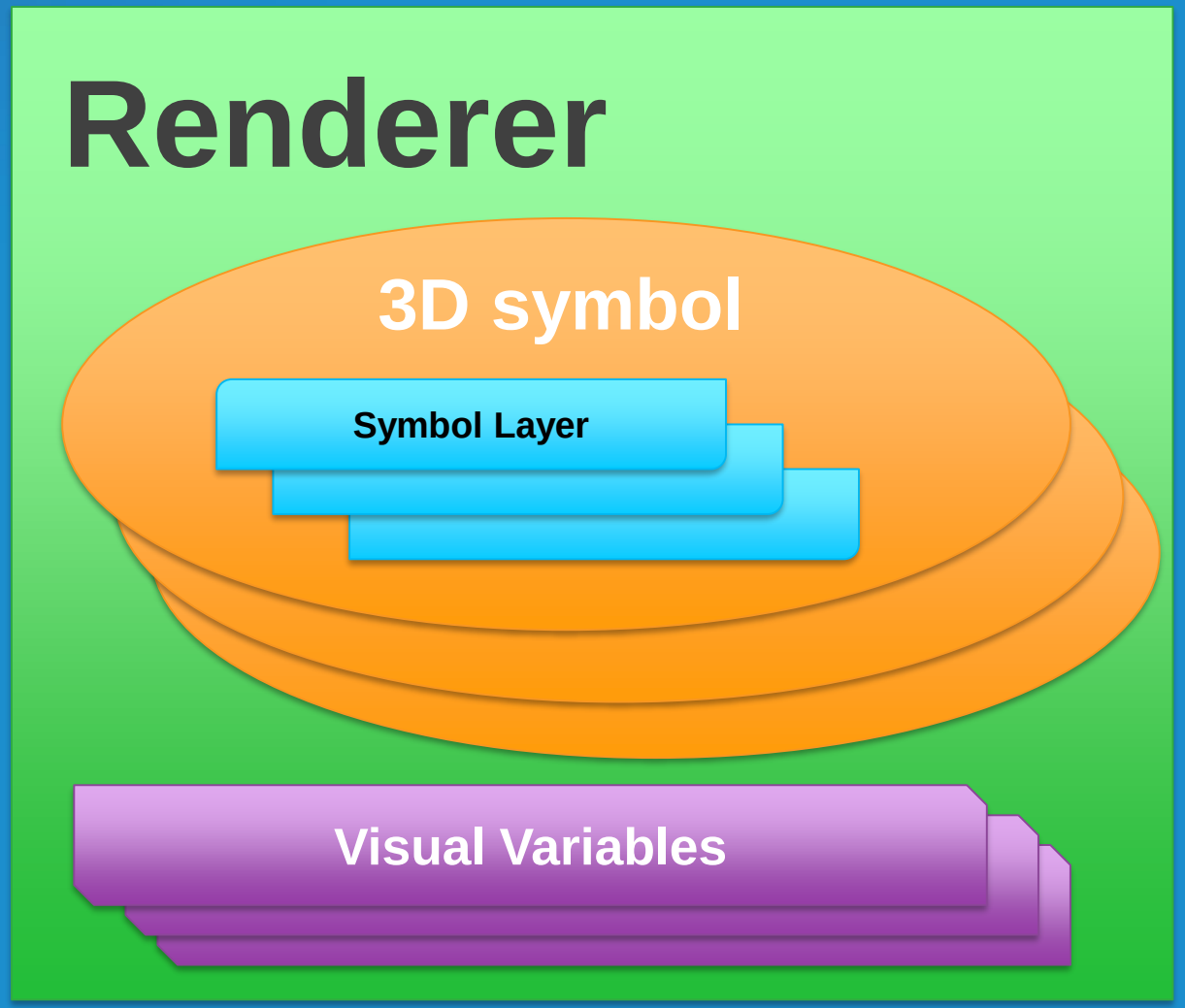


Overview

- What you can visualize in 3D
- How to do it
- Considerations and pitfalls

2D visualization vs. 3D visualization



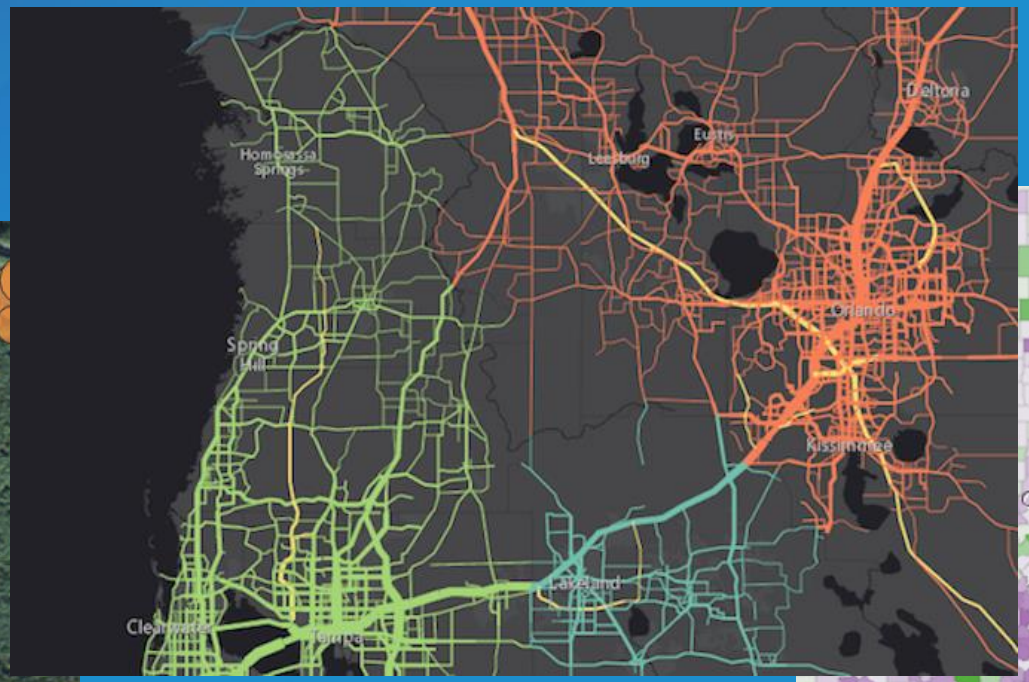


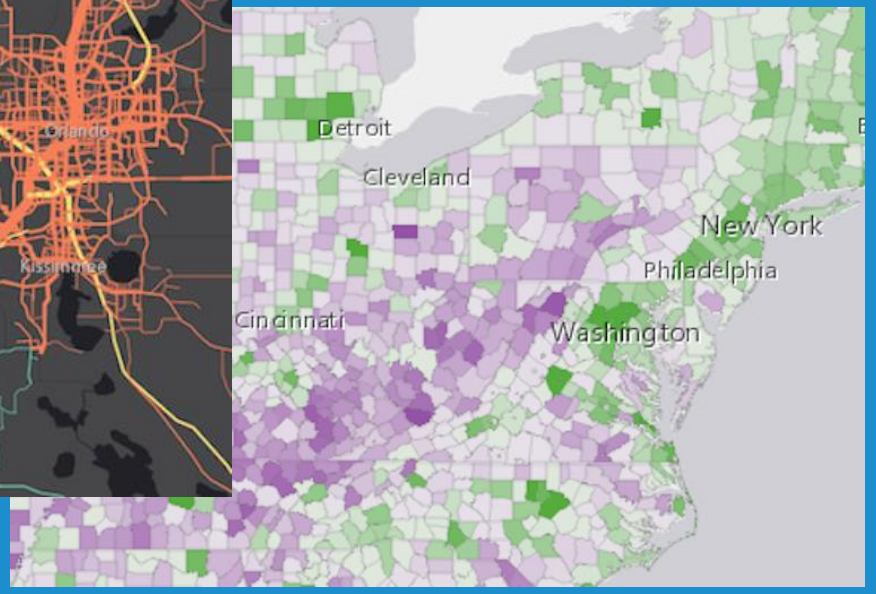
Symbols

2D Symbols

Not recommended!

- All 2D symbols (except PictureFillSymbol) are supported in 3D





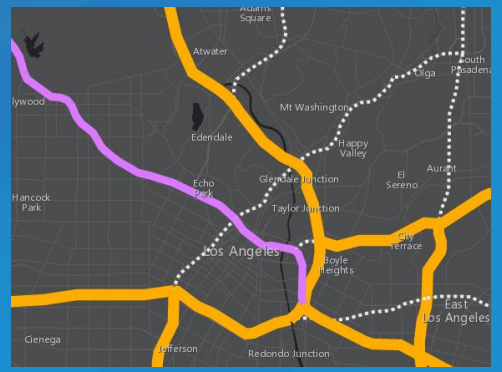
3D Symbols

based on geometry type

Points

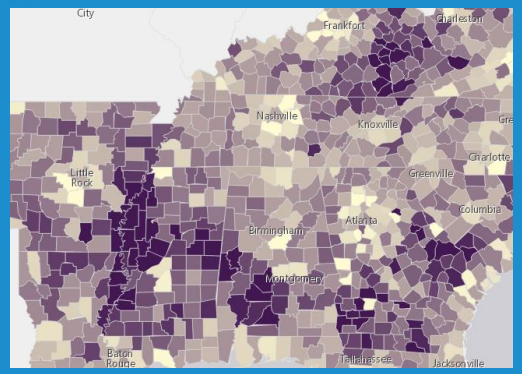
PointSymbol3D

Lines



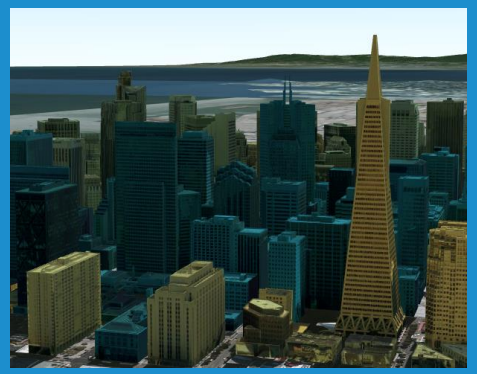
LineSymbol3D

Polygons



PolygonSymbol3D

Mesh



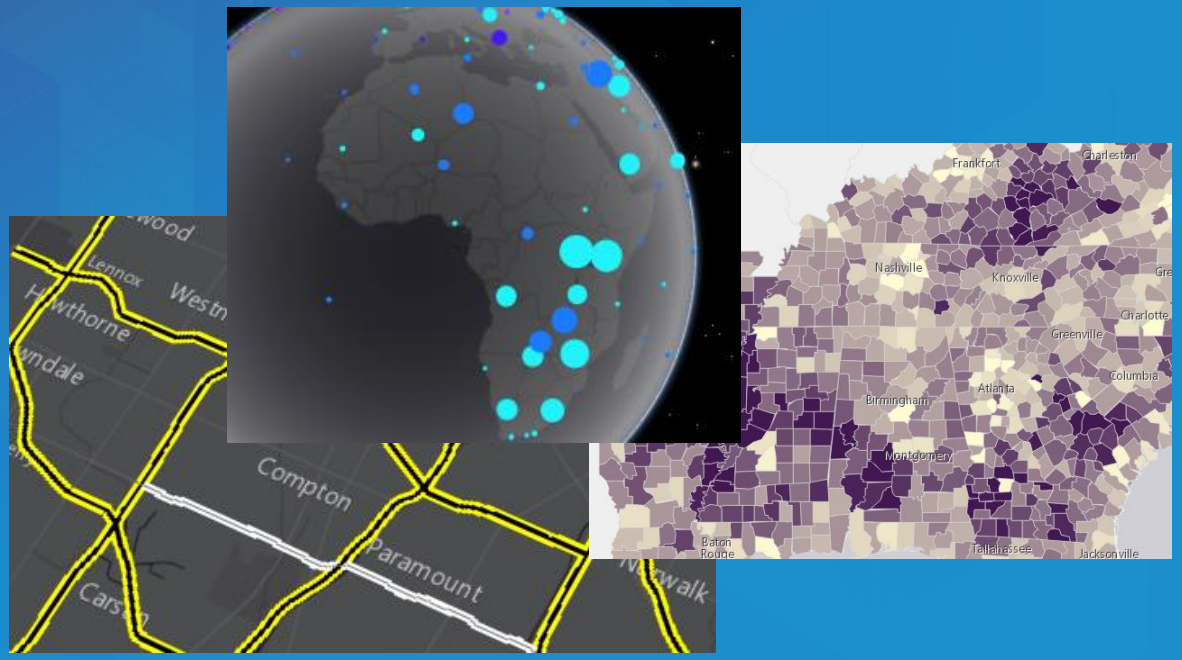
MeshSymbol3D

LabelSymbol3D

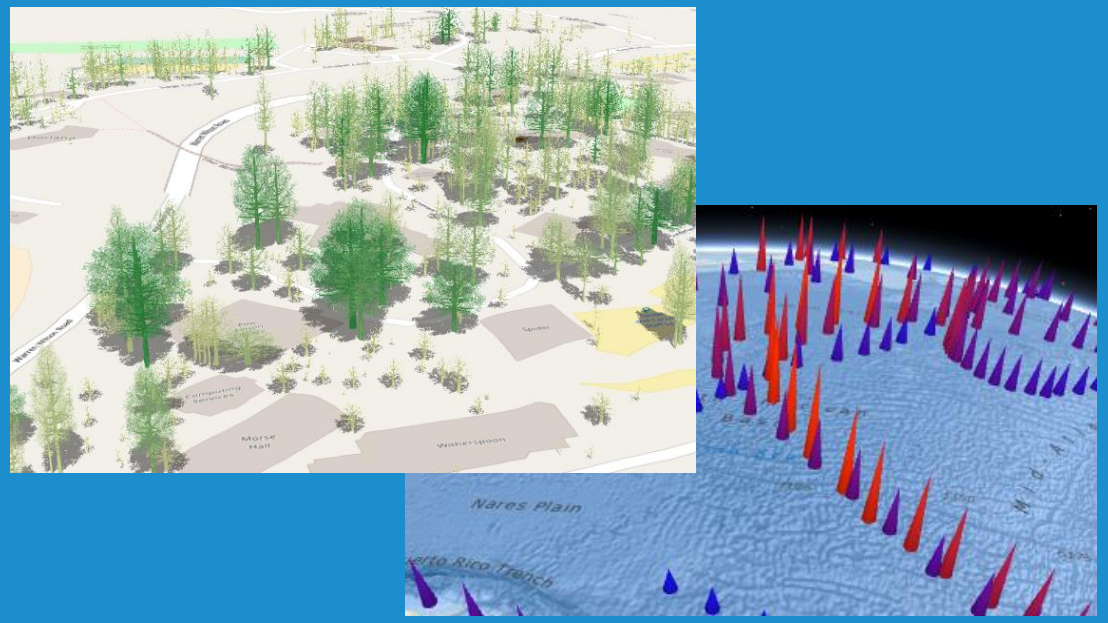
3D Symbol Layers

Each symbol must be composed of one or more symbol layers

Flat



Volumetric

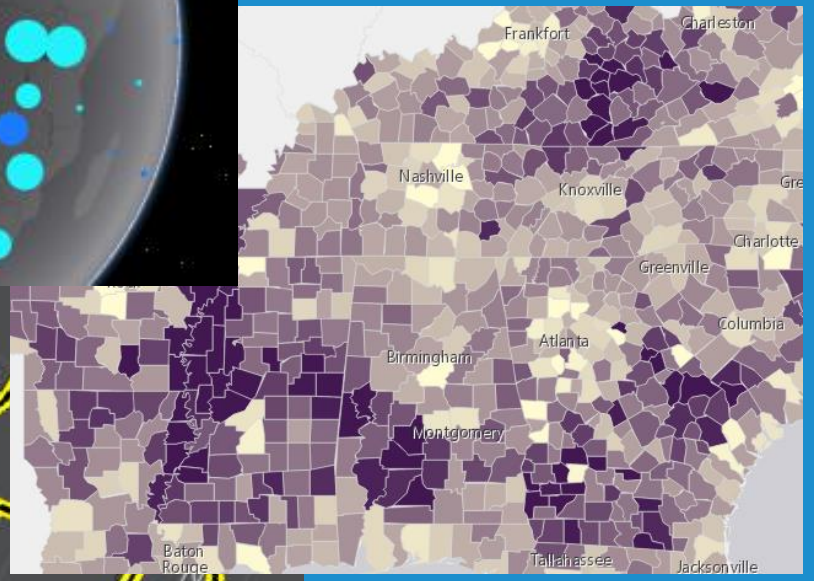


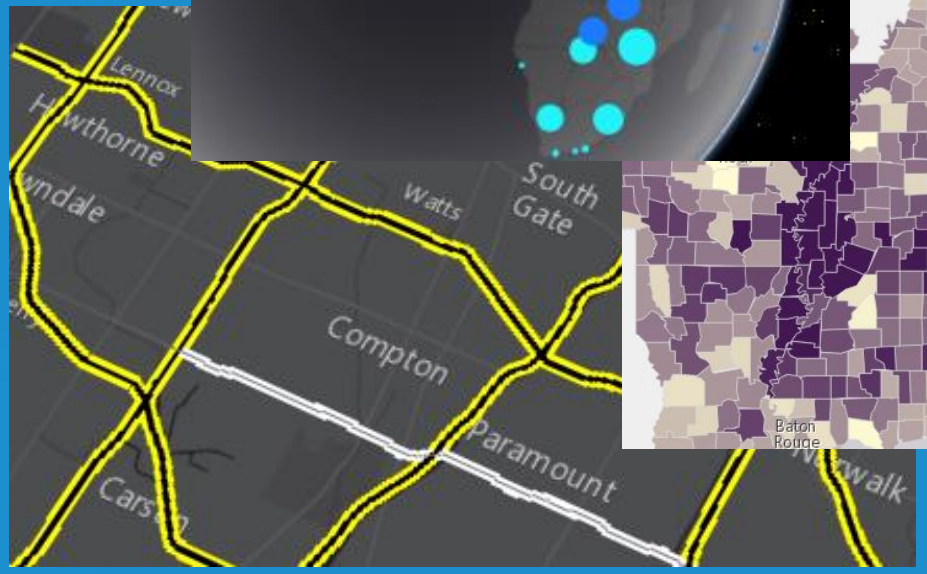
Flat Symbol Layers

- Occupy screen space
- Size is expressed in screen units
  - pixels or points

IconSymbol3DLayer

FillSymbol3DLayer



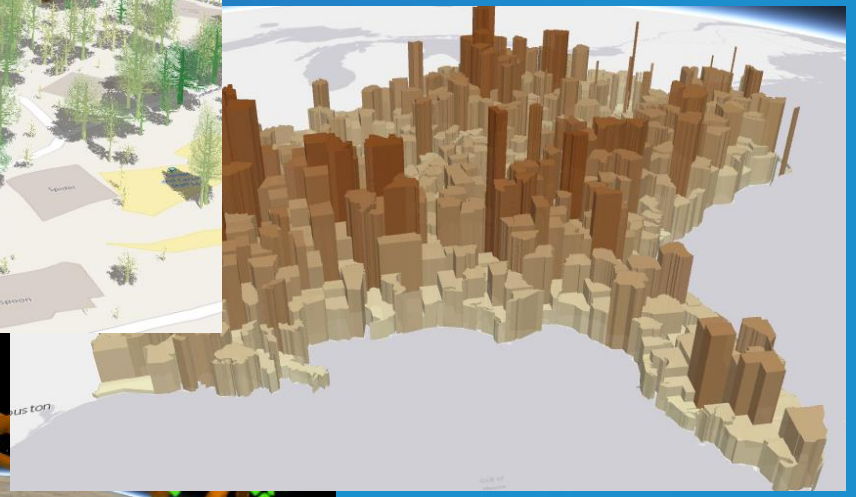


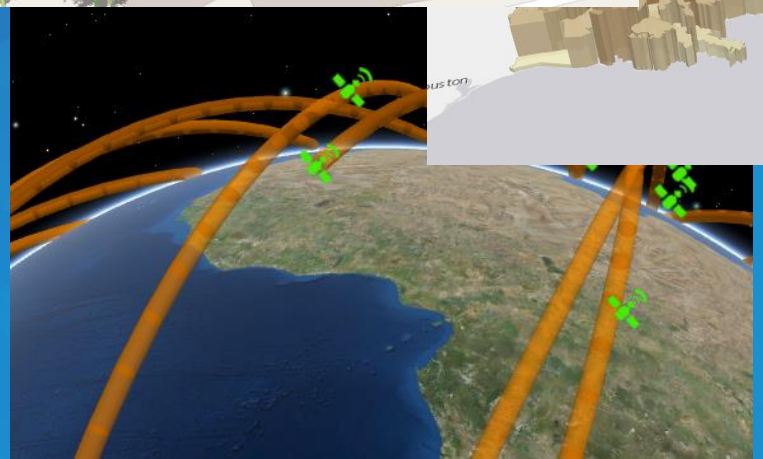
LineSymbol3DLayer

Volumetric Symbol Layers

ObjectSymbol3DLayer

ExtrudeSymbol3DLayer

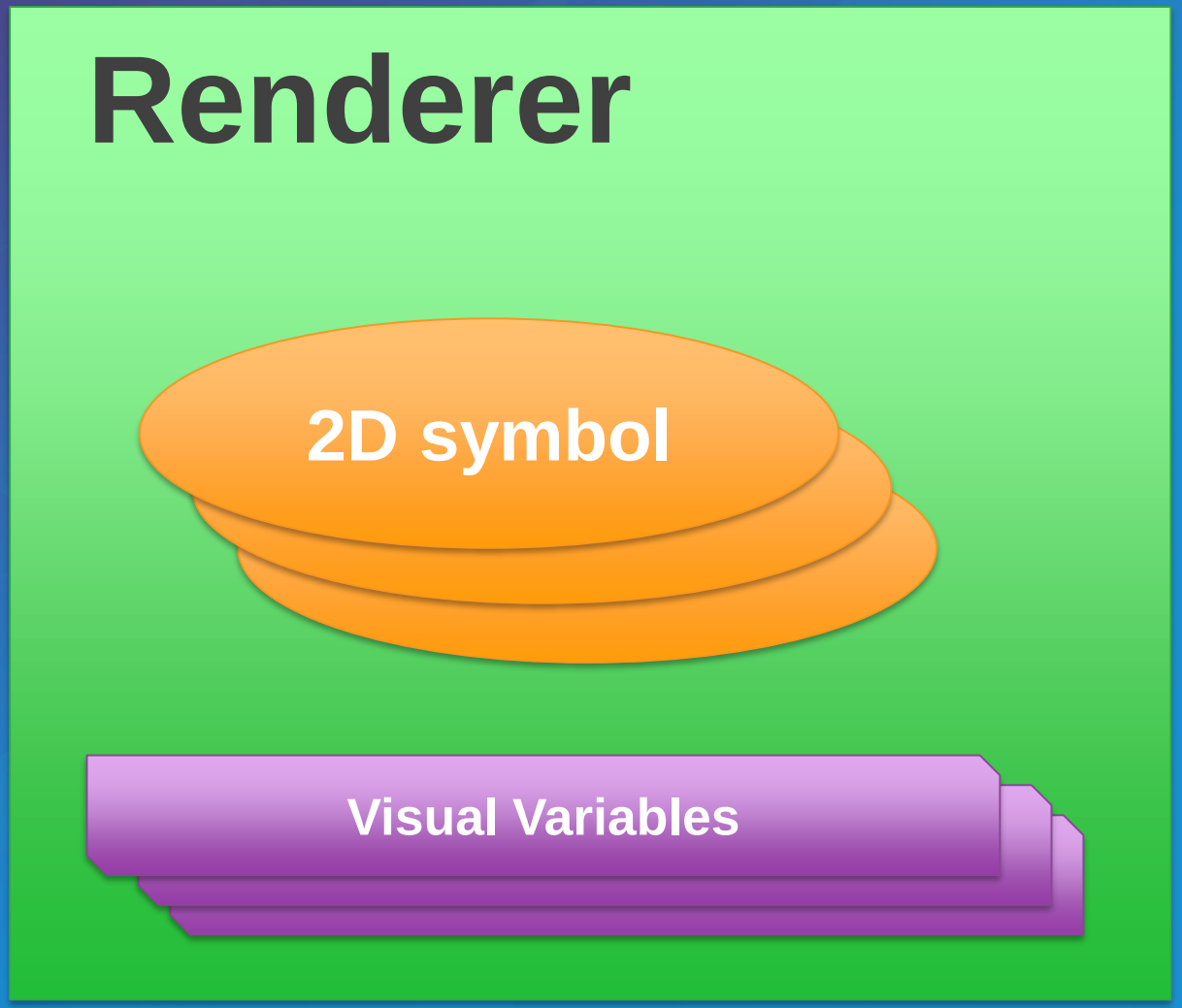


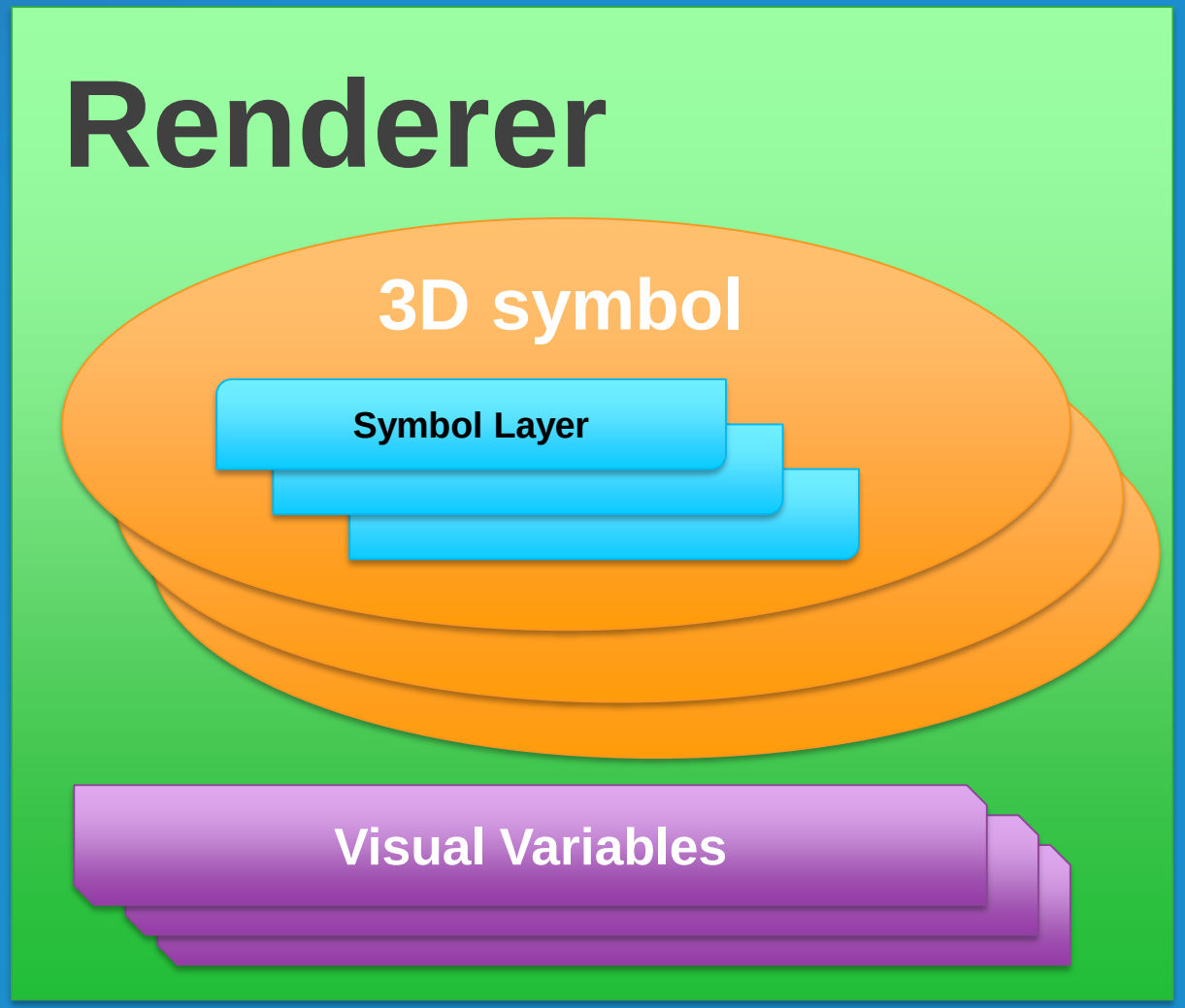


PathSymbol3DLayer

- Occupy space in virtual world
- Size is expressed in meters
- Object symbol layers (point features) have an axis to size height, width, and depth

2D visualization vs. 3D visualization





PointSymbol3D

Only for point features

```
objectSymbol = new PointSymbol3D({
  symbolLayers: [new ObjectSymbol3DLayer({
    width: 70000,
    height: 100000,
    resource: {
      primitive: "cone"
    },
    material: {
      color: "#FFD700"
    }
  })]
});
```

```
ıSymbol = new PointSymbol3D({
ymbolLayers: [new IconSymbol3DLayer({
 size: 12,
 resource: {
   primitive: "square"
 },
 material: {
   color: "orange"
 },
 outline: {
   color: "white",
   size: 1
 }
})]
});
```

Flat (size in points/pixels)

PointSymbol3D

ObjectSymbol3DLayer – resource types

cone

sphere

tetrahedron

cylinder

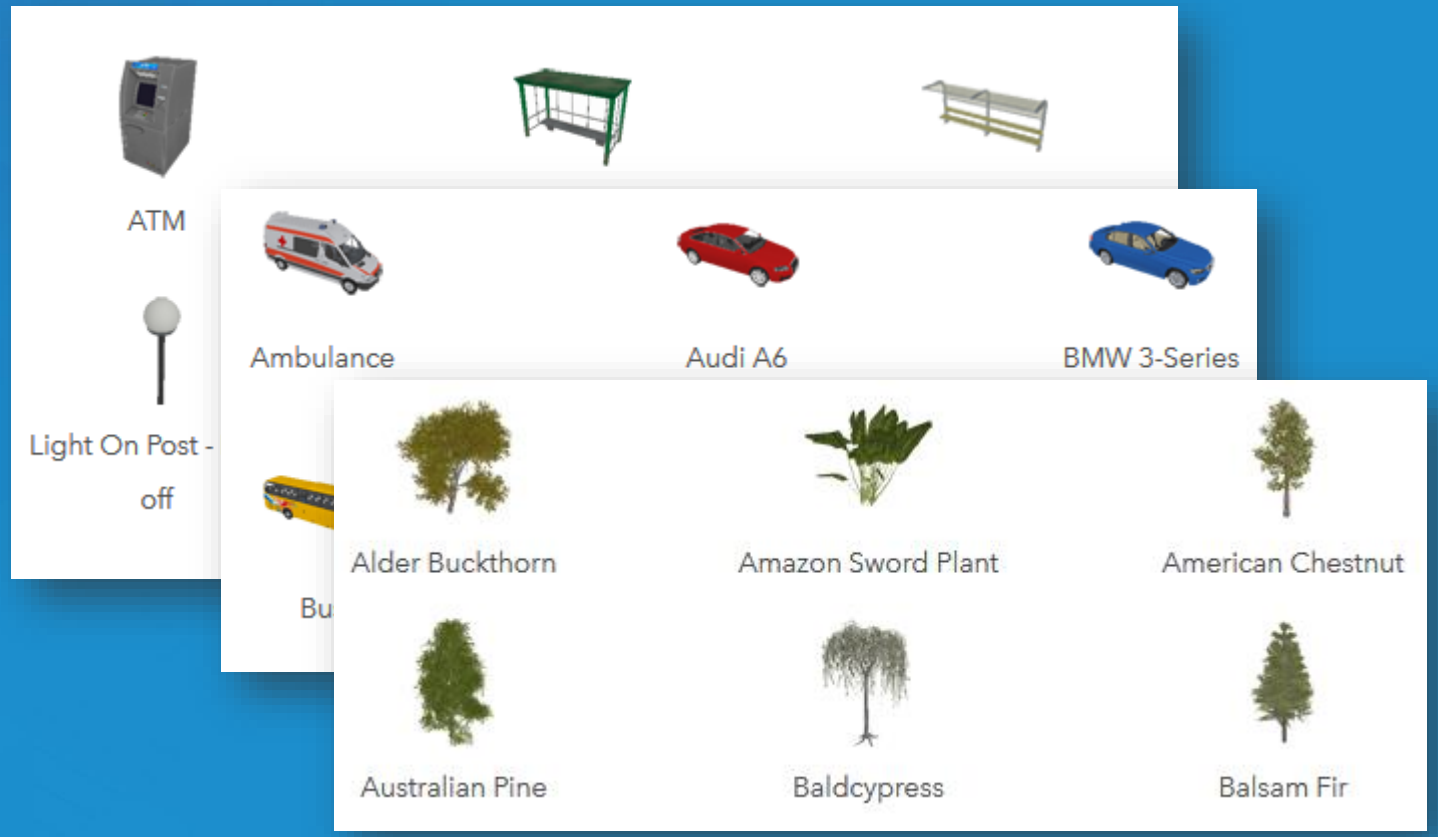
diamond

cube

PointSymbol3D

ObjectSymbol3DLayer – web styles

- Street assets
- Vehicles
- Vegetation
- …



PointSymbol3D

ObjectSymbol3DLayer – web styles

- **WebStyleSymbol**

```
var webStyleSymbol = new WebStyleSymbol({
  name: "Bus_Stop_1",
  portal: {
    url: "https://www.arcgis.com"
  },
  styleName: "EsriRealisticStreetSceneStyle"
});
```

- https://developers.arcgis.com/javascript/latest/guide/visualizing-points-3d/index.html
- https://developers.arcgis.com/javascript/latest/guide/esri-web-style-symbols/index.html

PointSymbol3D

IconSymbol3DLayer – resource types

circle

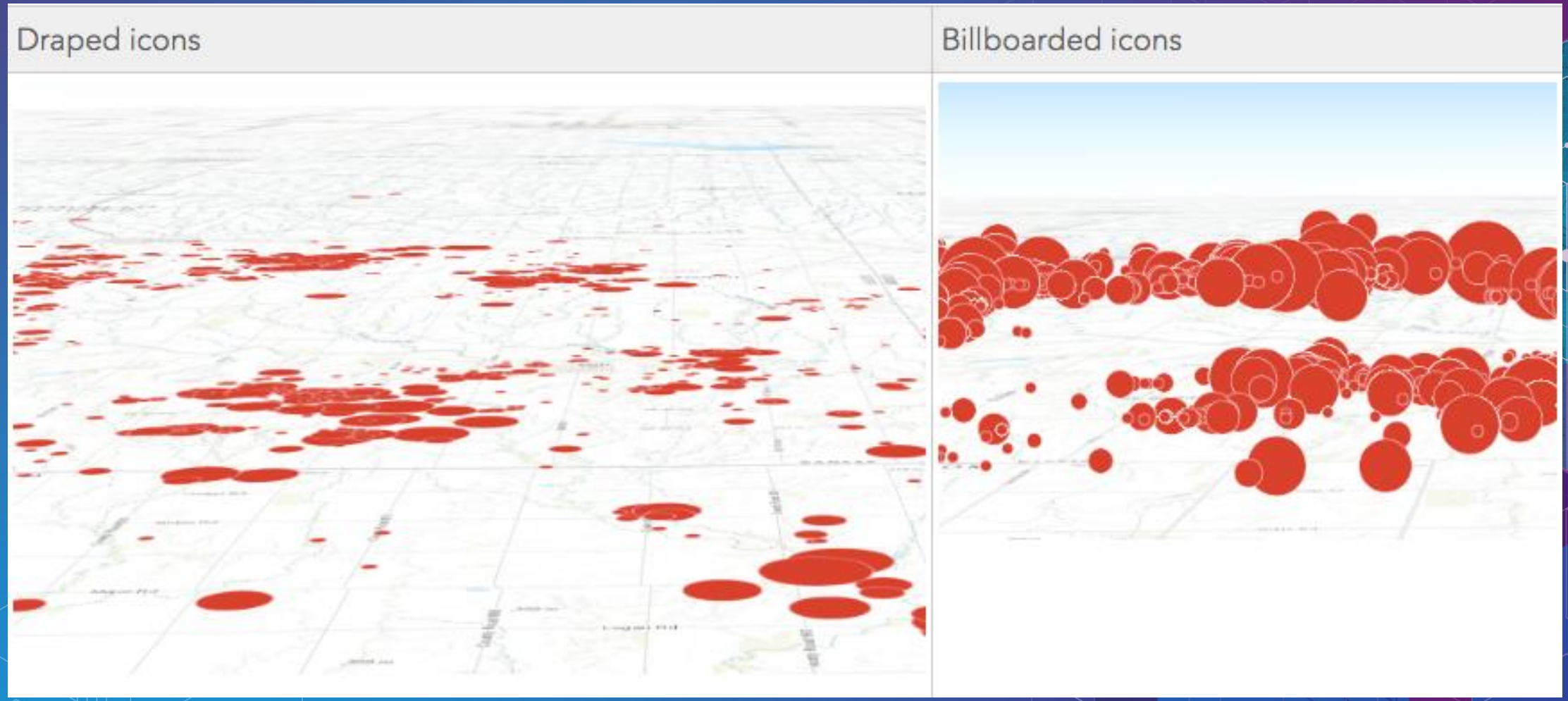
square

kite

cross

x

Keep in mind…



Sample comparison: When is each appropriate?

LineSymbol3D

Only for polyline features

LineSymbol3DLayer

```
symbol: new LineSymbol3D({
  symbolLayers: [ new LineSymbol3DLayer({
    material: { color: [192,192,192,0.5] },
    size: 3
  }) ]
})
```

Flat (size in points/pixels)

PathSymbol3DLayer

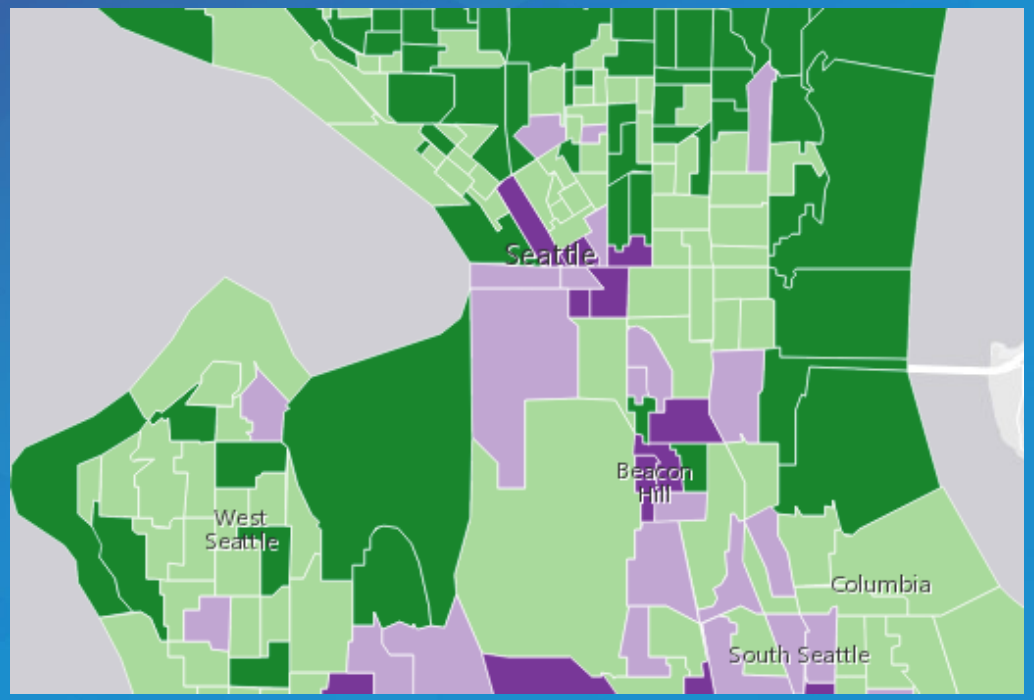
```
symbol: new LineSymbol3D({
  symbolLayers: [ new PathSymbol3DLayer({
    material: { color: [192,192,192,0.5] },
    size: 5000 // meters
  }) ]
})
```

Volumetric (size in meters)

PolygonSymbol3D

Only for polygon features

FillSymbol3DLayer



Flat (no size)

ExtrudeSymbol3DLayer

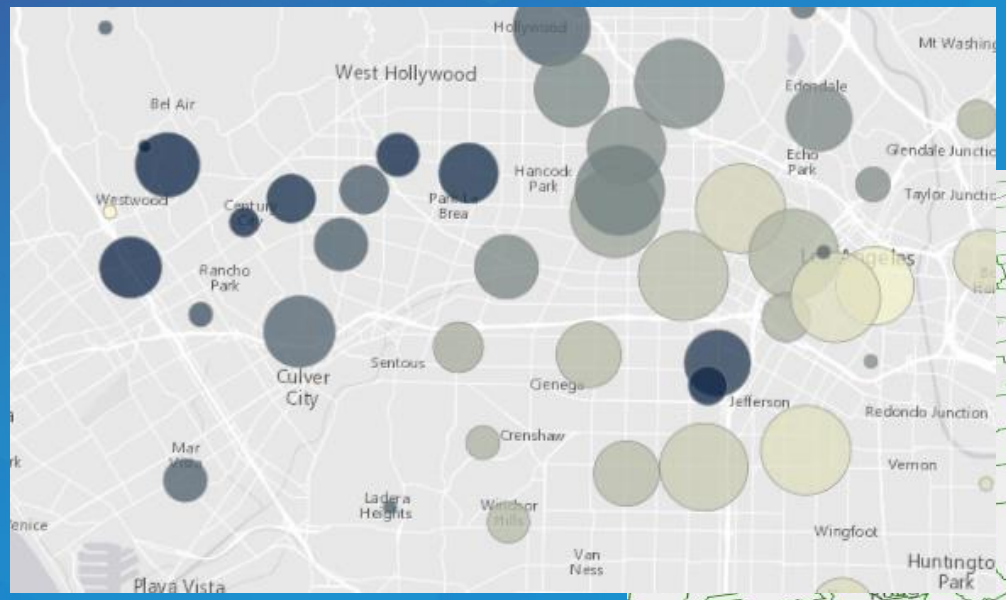
```
var less35 = new PolygonSymbol3D({
  symbolLayers: [
    new FillSymbol3DLayer({
      material: { color: "#7B3294" },
      outline: {
        size: 0.5,
        color: "white"
      }
symbol: new PolygonSymbol3D({
  symbolLayers: [ new ExtrudeSymbol3DLayer() ]
}),
```

Volumetric (size in meters)

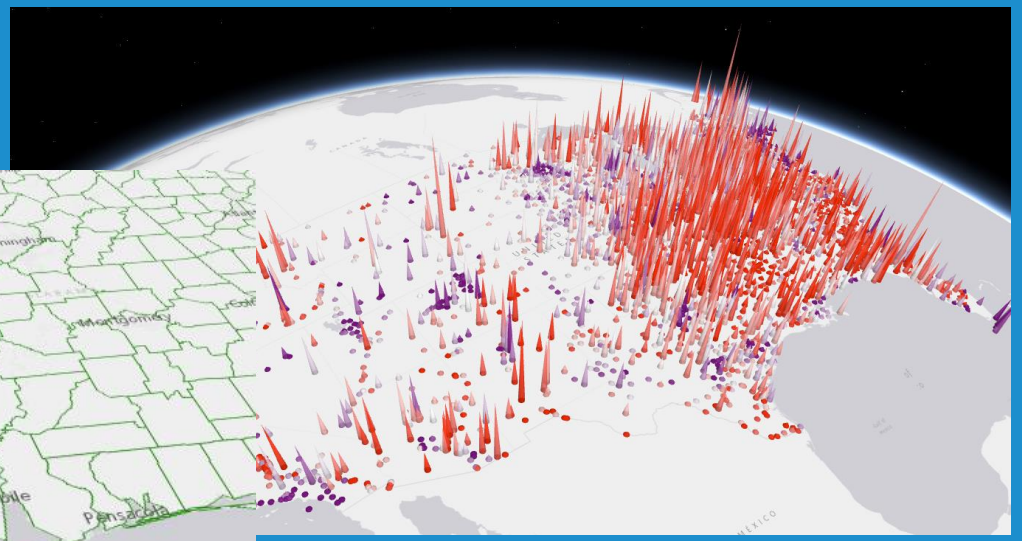
PolygonSymbol3D

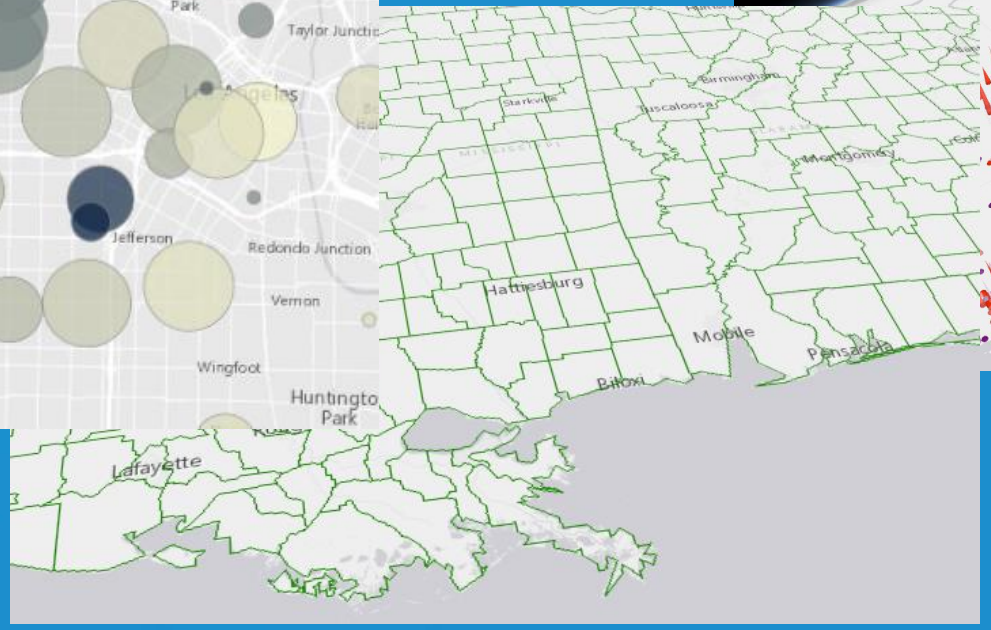
Also supports other symbol layers

IconSymbol3DLayer



ObjectSymbol3DLayer



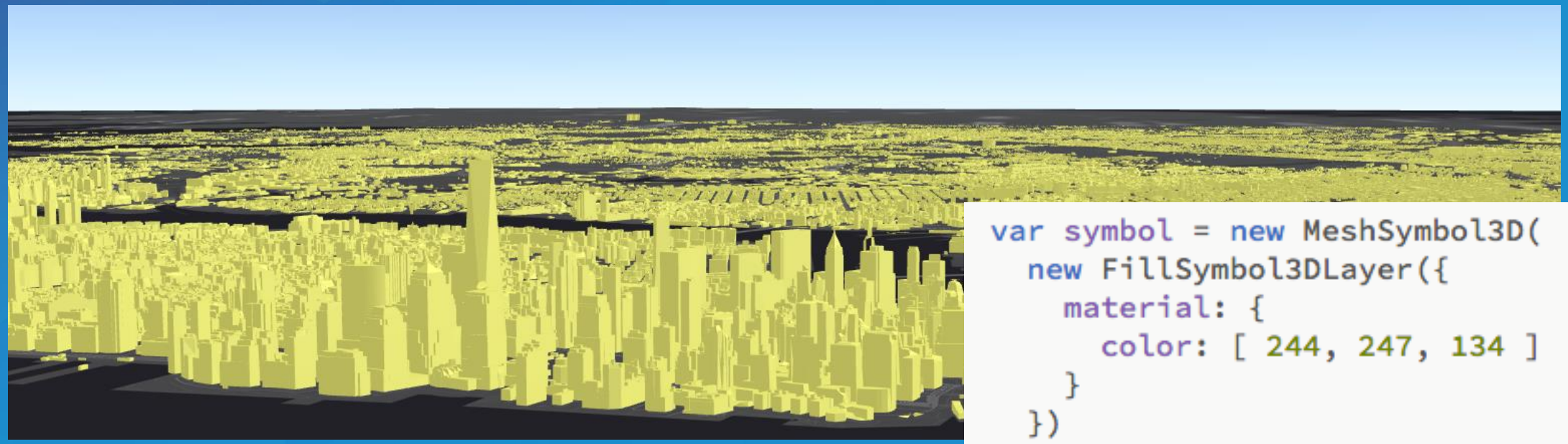


LineSymbol3DLayer

MeshSymbol3D

Only for mesh features (in SceneLayers)

FillSymbol3DLayer



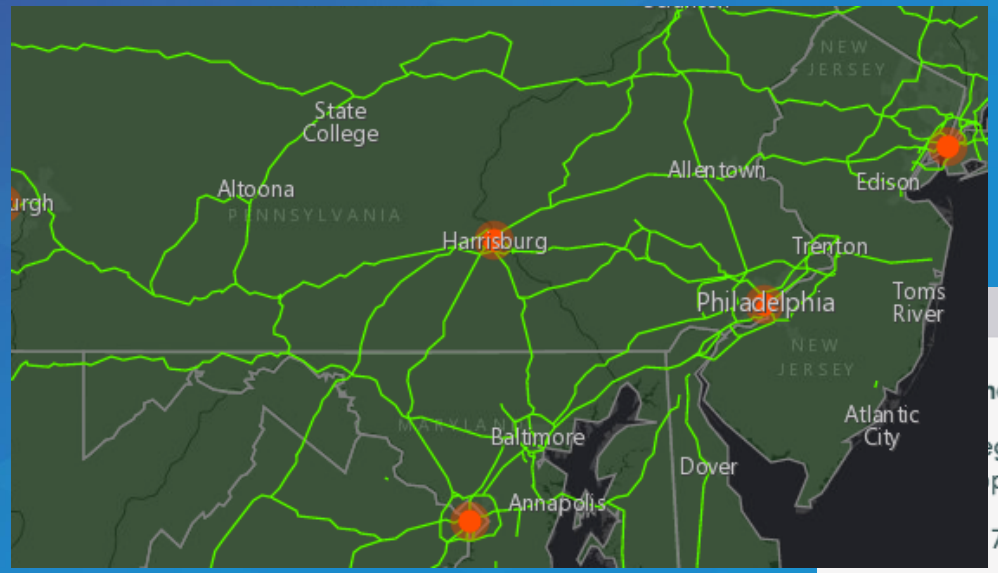
```
var symbol = new MeshSymbol3D(
  new FillSymbol3DLayer({
    material: {
      color: [ 244, 247, 134 ]
    }
  })
);
```

Renderers

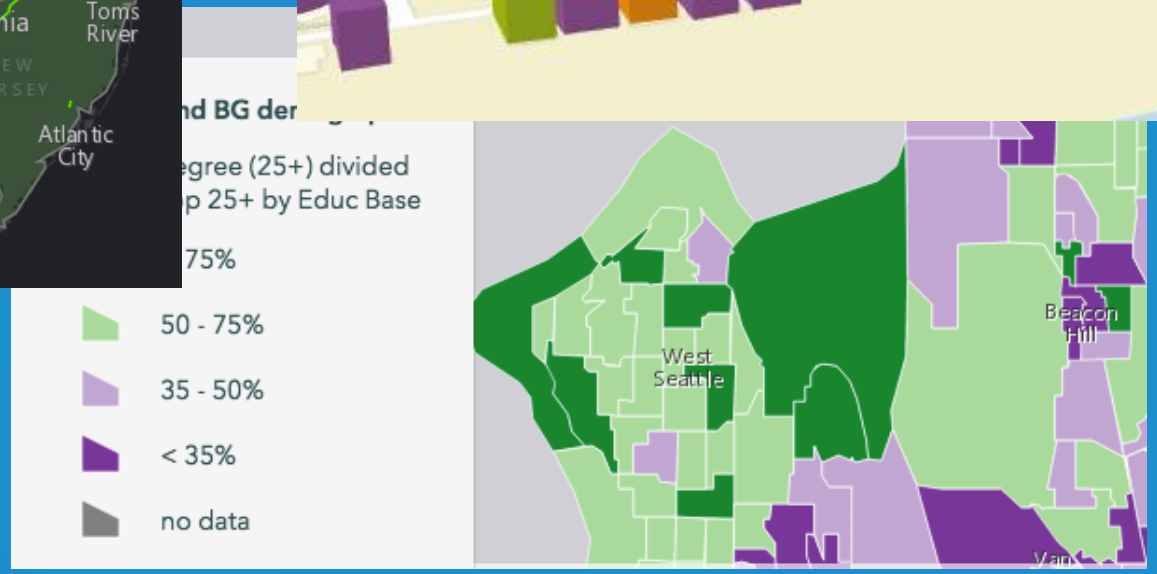
Renderers

UniqueValueRenderer

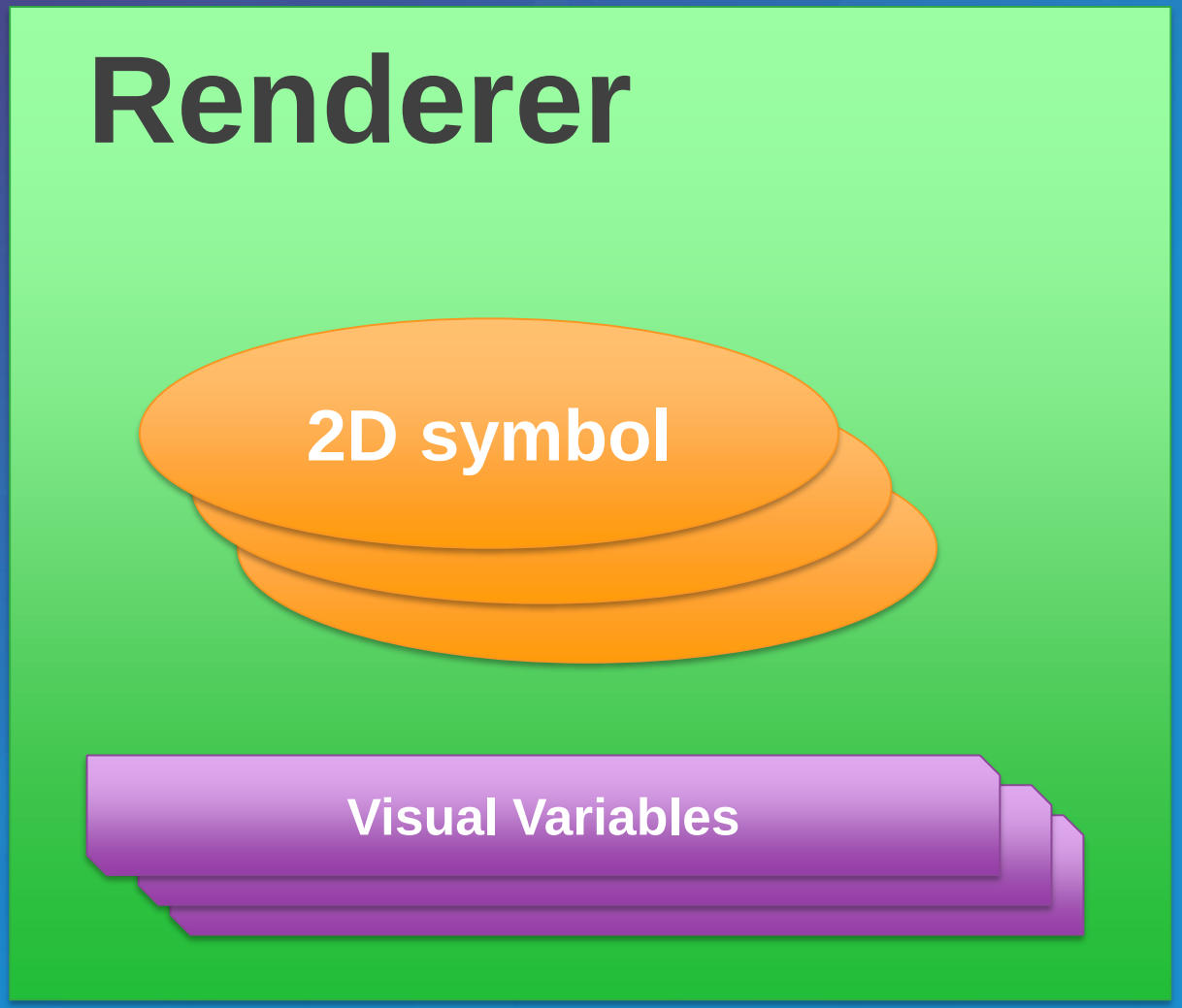
SimpleRenderer

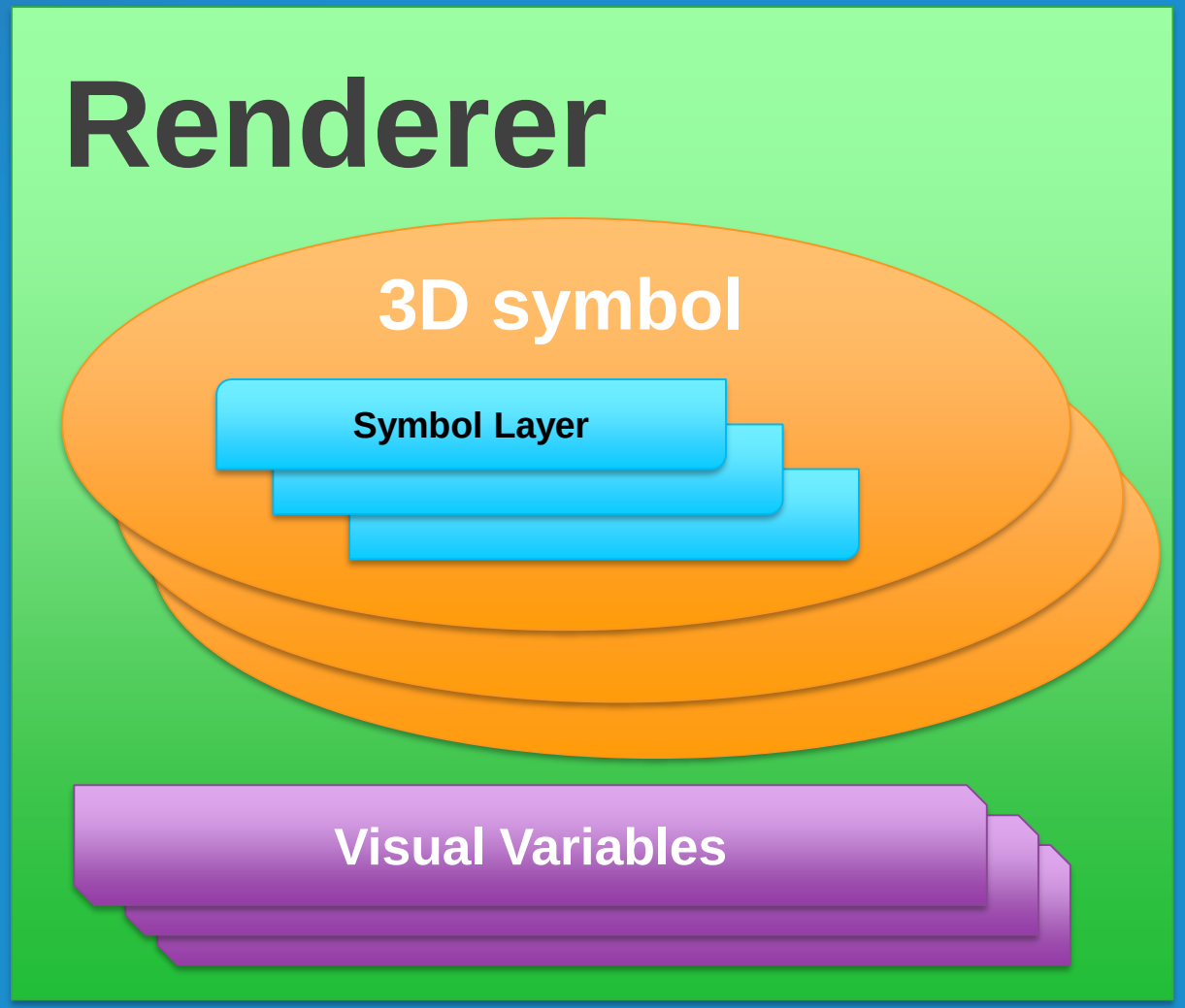


ClassBreaksRenderer



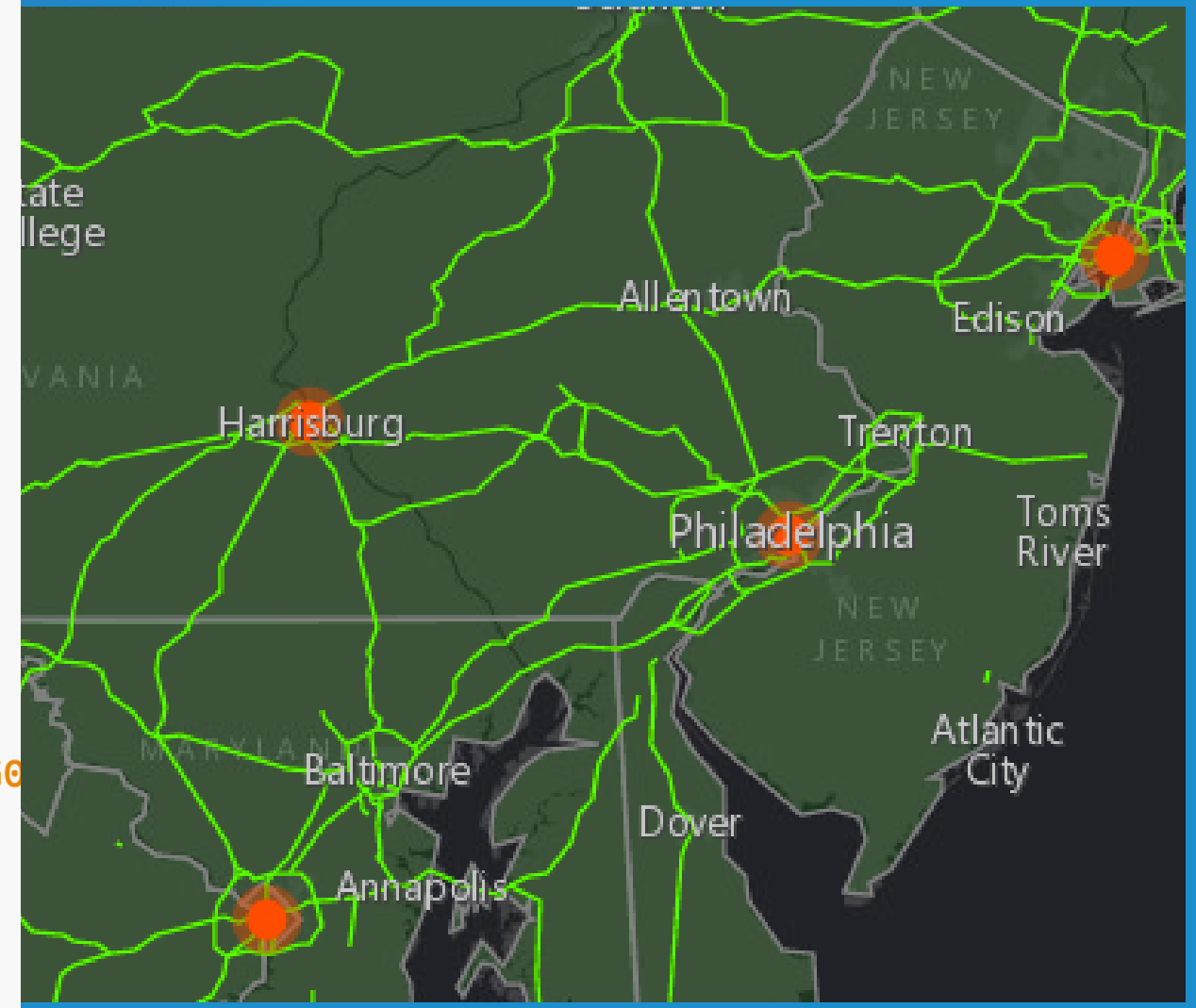
2D visualization vs. 3D visualization





SimpleRenderer

```
var citiesRenderer = new SimpleRenderer({
  symbol: new PointSymbol3D({
    symbolLayers: [ new PointSymbol3DLayer({
      size: 10,
      material: { color: "#FF4000" },
      outline: {
        color: [ 255, 64, 0, 0.4 ],
        size: 7
      }
    })]
  })
});

var citiesLyr = new FeatureLayer({
  url:
  "https://services.arcgis.com/V6ZHFr6zdgNZuVG0
  /WorldCities/FeatureServer/0",
  renderer: citiesRenderer,
  popupTemplate: { content: "{*}" }
});
```



UniqueValueRenderer

Buildings with real-world height

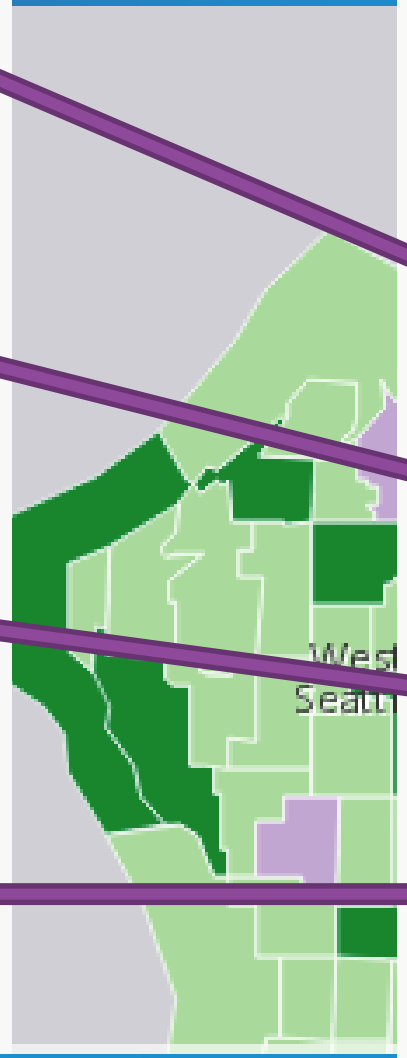
Buildings with real-world height

```
var resSym = new PolygonSymbol3D({
  symbolLayers: [
    new ExtrudeSymbol3DLayer({
      material: {
        color: "#FC921F"
      }
    })
  ]
});

var condoSym = new PolygonSymbol3D({
  symbolLayers: [
    new ExtrudeSymbol3DLayer({
      material: {
        color: "#9E559C"
      }
    })
  ]
});
```

```
var renderer = new UniqueValueRenderer({
  defaultSymbol: new PolygonSymbol3D({
    symbolLayers: [new ExtrudeSymbol3DLayer({
      material: {
        color: "#A7C636"
      }
    })]
  }),
  defaultLabel: "Other",
  field: "DESCLU",
  uniqueValueInfos: [
  {
    value: "Residential",
    symbol: resSym,
    label: "Residential"
  }, {
    value: "Residential Condominium",
    symbol: condoSym,
    label: "Condominium"
  }],
  visualVariables: [{
    type: "size",
    field: "ELEVATION",
    valueUnit: "feet" // Converts and extrudes all
                      // data values in feet
  }]
});
```

ClassBreaksRenderer

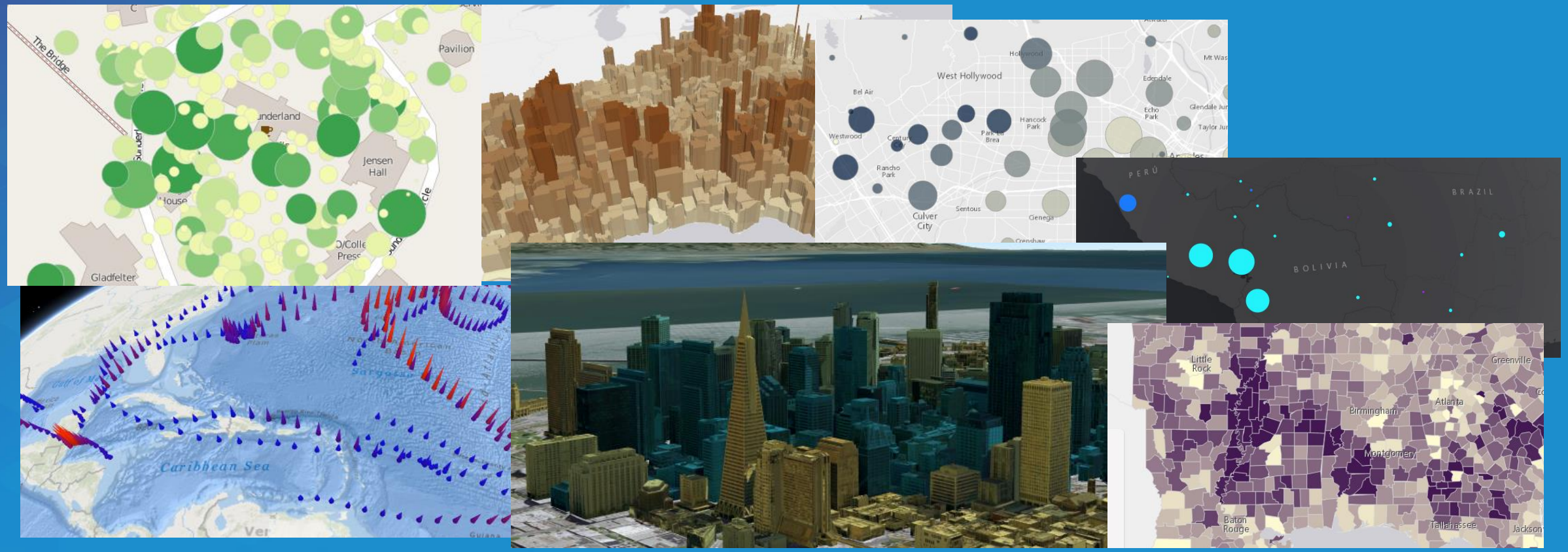
```
var less35 = new PolygonSymbol3D({
  symbolLayers: [ new FillSymbol3DLayer({
    ...
  })]
});

var less50 = new PolygonSymbol3D({
  symbolLayers: [ new FillSymbol3DLayer({
    ...
  })]
});

var more50 = new PolygonSymbol3D({
  symbolLayers: [ new FillSymbol3DLayer({
    ...
  })]
});

var more75 = new PolygonSymbol3D({
  symbolLayers: [ new FillSymbol3DLayer({
    ...
  })]
});
```



```
var renderer = new ClassBreaksRenderer({
  field: "COL_DEG",
  normalizationField: "EDUCBASECY",
  defaultSymbol: new SimpleFillSymbol({
    color: "gray",
    outline: {
      width: 0.5,
      color: "white"
    }
  }),
  defaultLabel: "no data",
  classBreakInfos: [
    {
      minValue: 0,
      maxValue: 0.3499,
      symbol: less35,
      label: "< 35%"
    }, {
      minValue: 0.35,
      maxValue: 0.4999,
      symbol: less50,
      label: "35 - 50%"
    }, {
      minValue: 0.50,
      maxValue: 0.7499,
      symbol: more50,
      label: "50 - 75%"
    }, {
      minValue: 0.75,
      maxValue: 1.00,
      symbol: more75,
      label: "> 75%"
    }
  ]
});
```

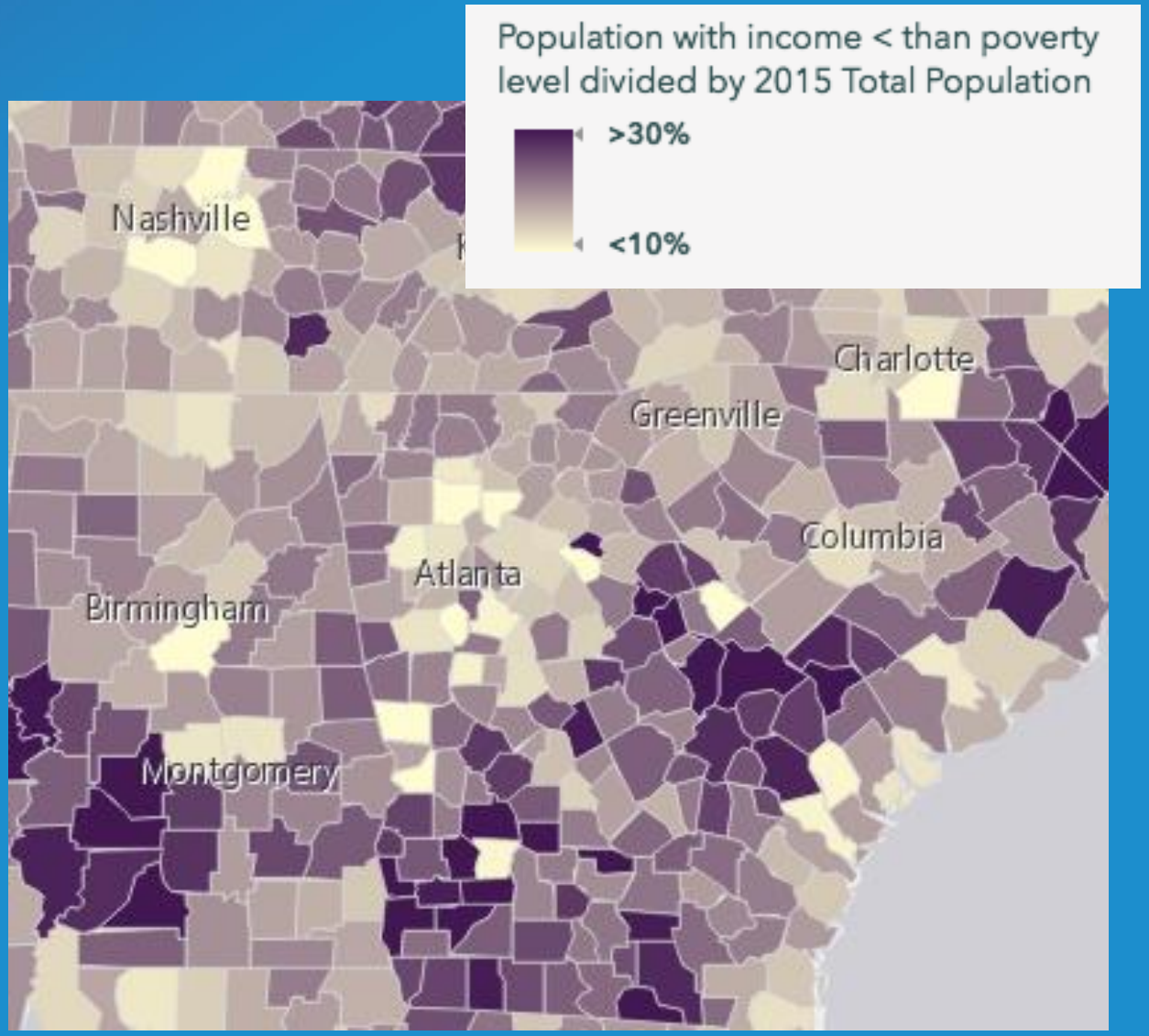
Visual Variables

- A property of the renderer
- For numeric data-driven continuous visualizations



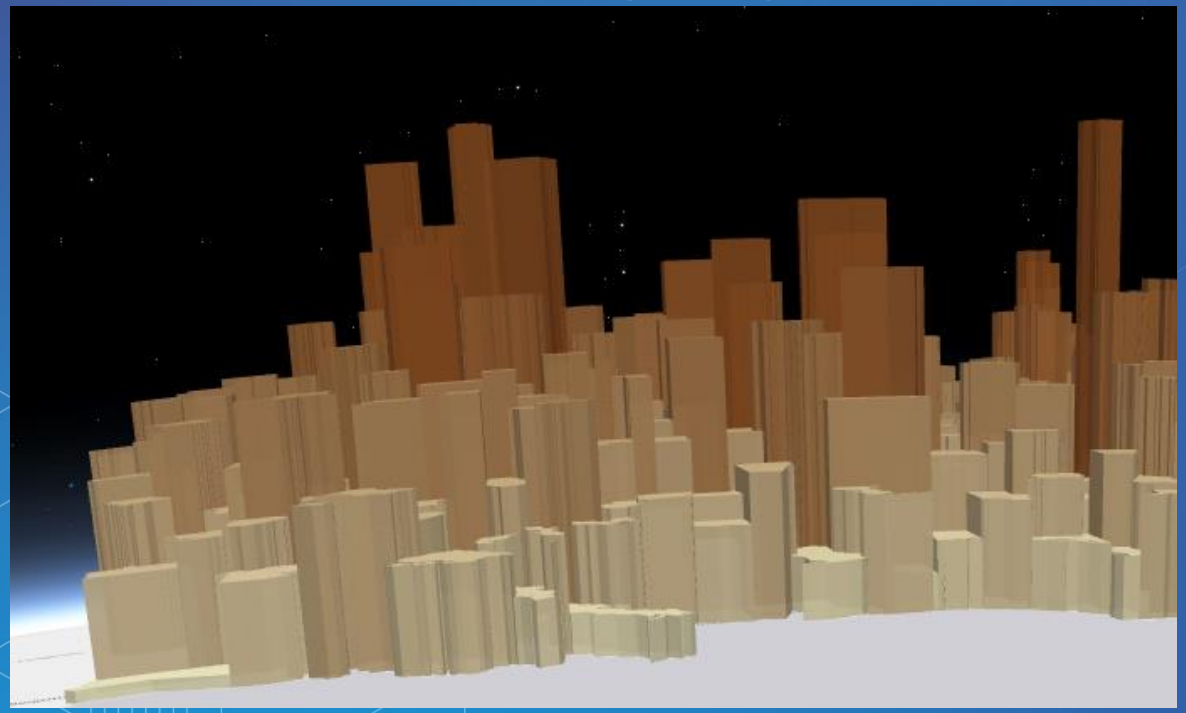
Visual Variables: Color

```
var colorVV = {
  type: “color”,
  field: “POP_POVERTY”,
  normalizationField: “TOTPOP_CY”,
  stops: [
    {  value: 0.1,  color: “#FFFCD4”  },
    {  value: 0.3,  color: “#350242”  }
  ]
};
```

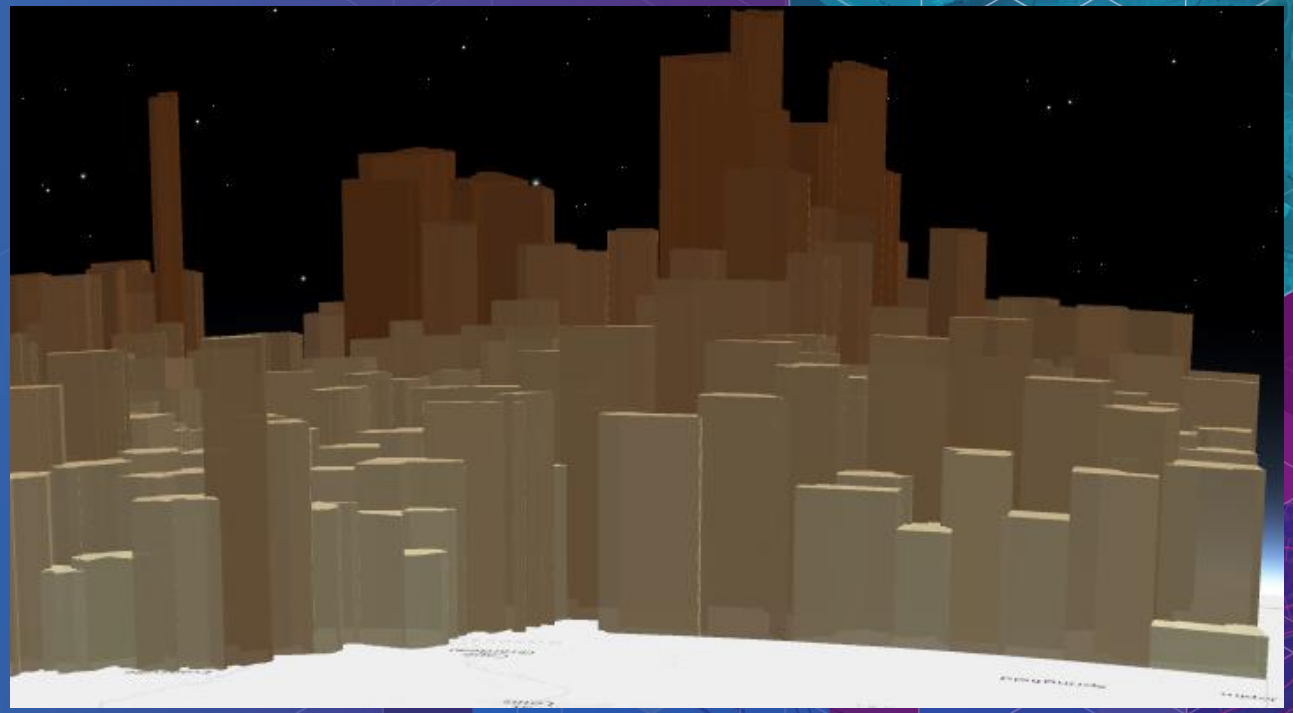


Keep in mind…

Direct lighting

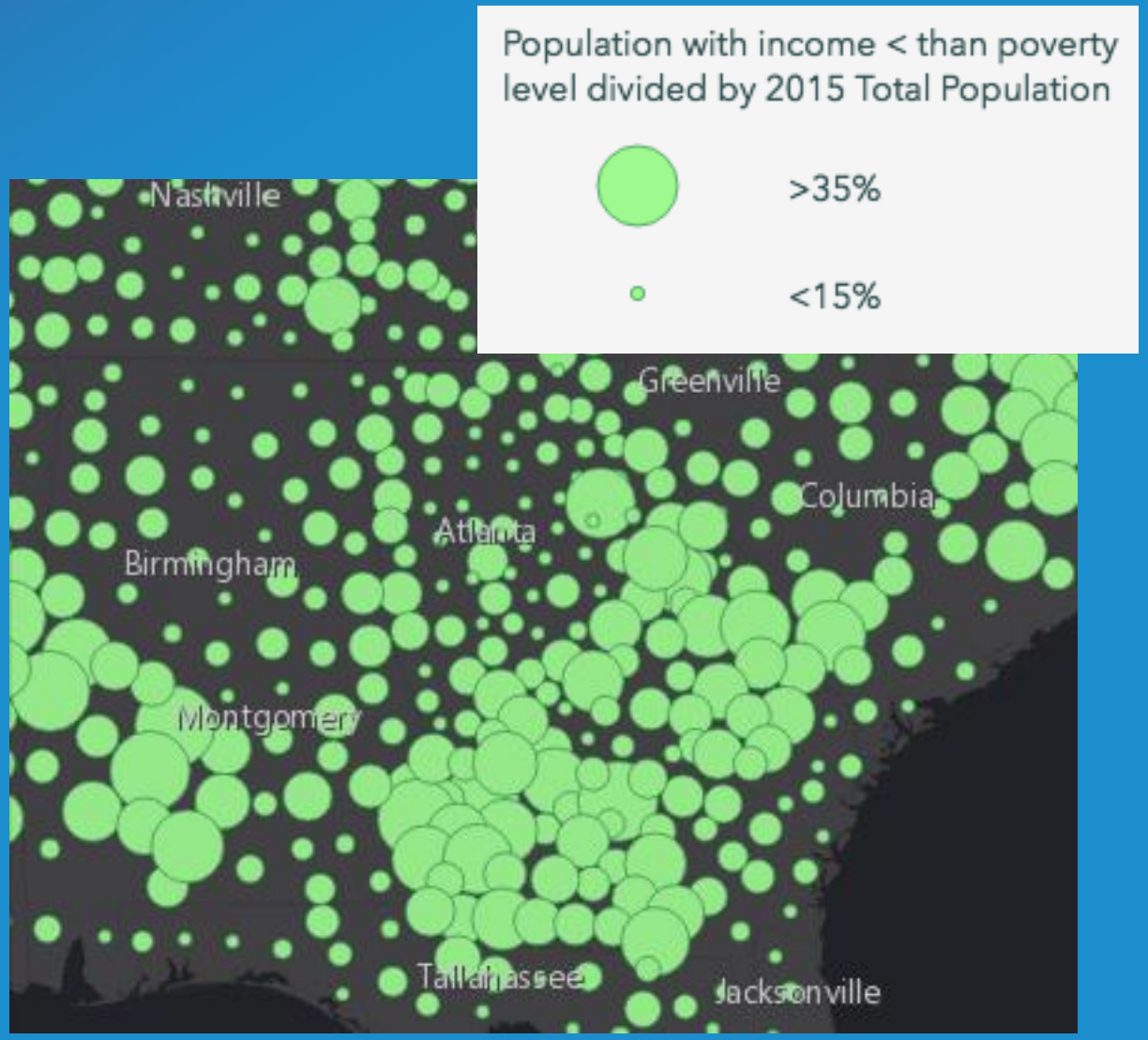


Shade



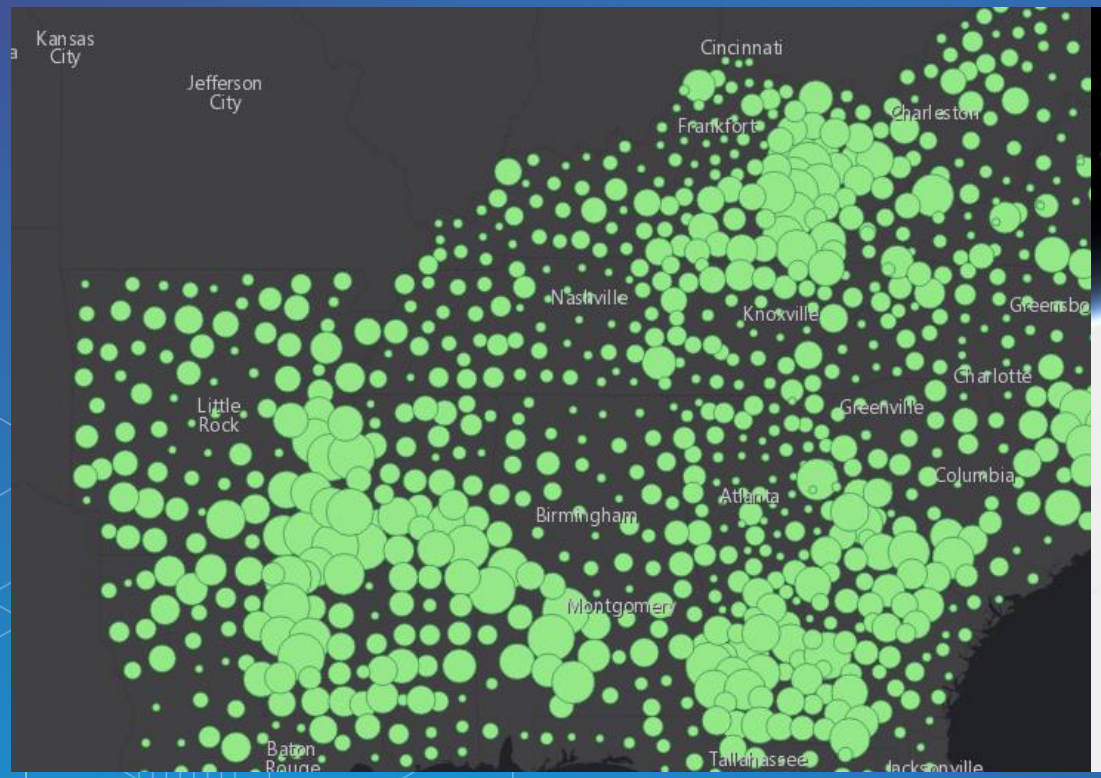
Visual Variables: Size

```
var sizeVV = {
  type: “size”,
  field: “POP_POVERTY”,
  normalizationField: “TOTPOP_CY”,
  stops: [
    {  value: 0.20,  size: “4px”  },
    {  value: 0.65,  size: “60px”  }
  ]
};
```



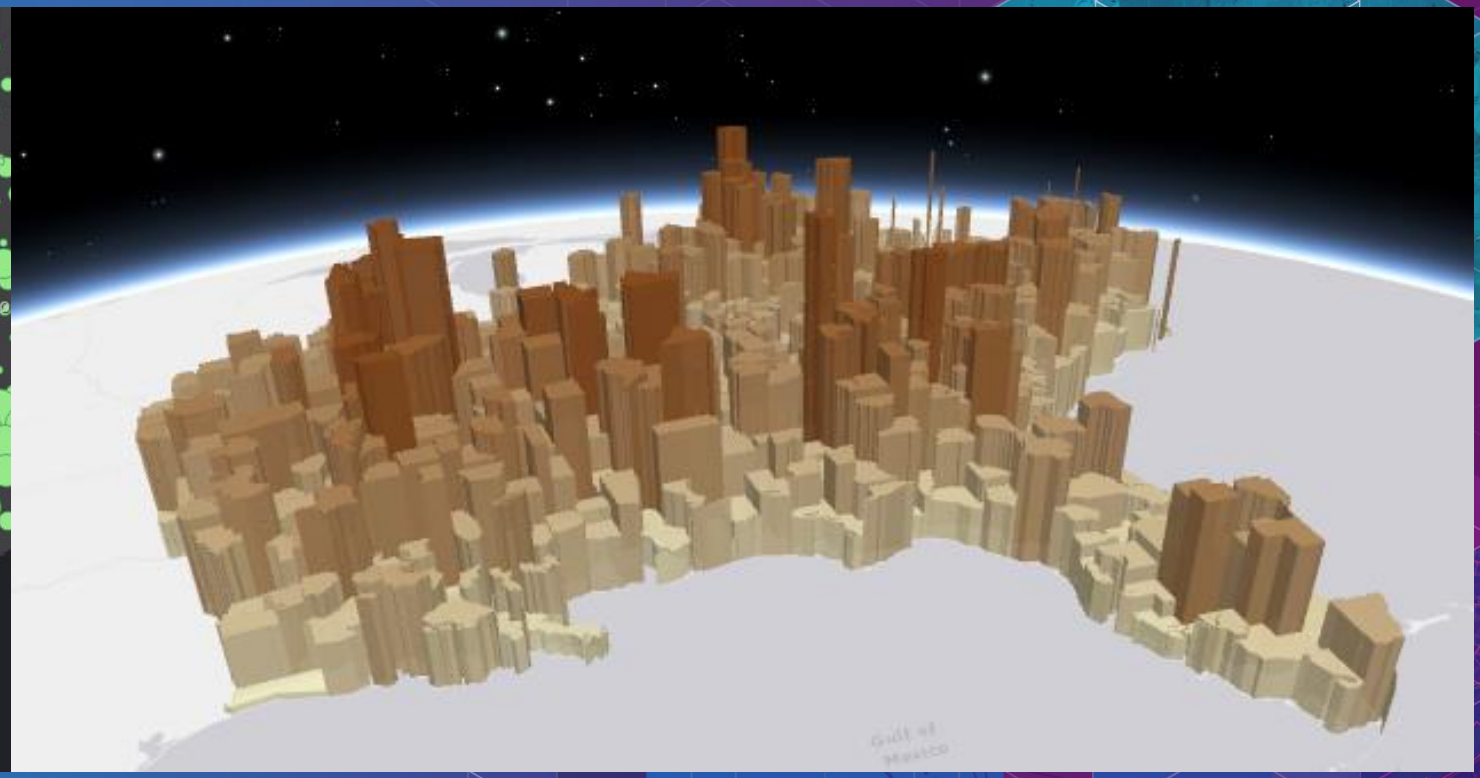
Keep in mind…

Flat symbols (pixels or points)



Good for multiple scale levels

Volumetric symbols (meters)



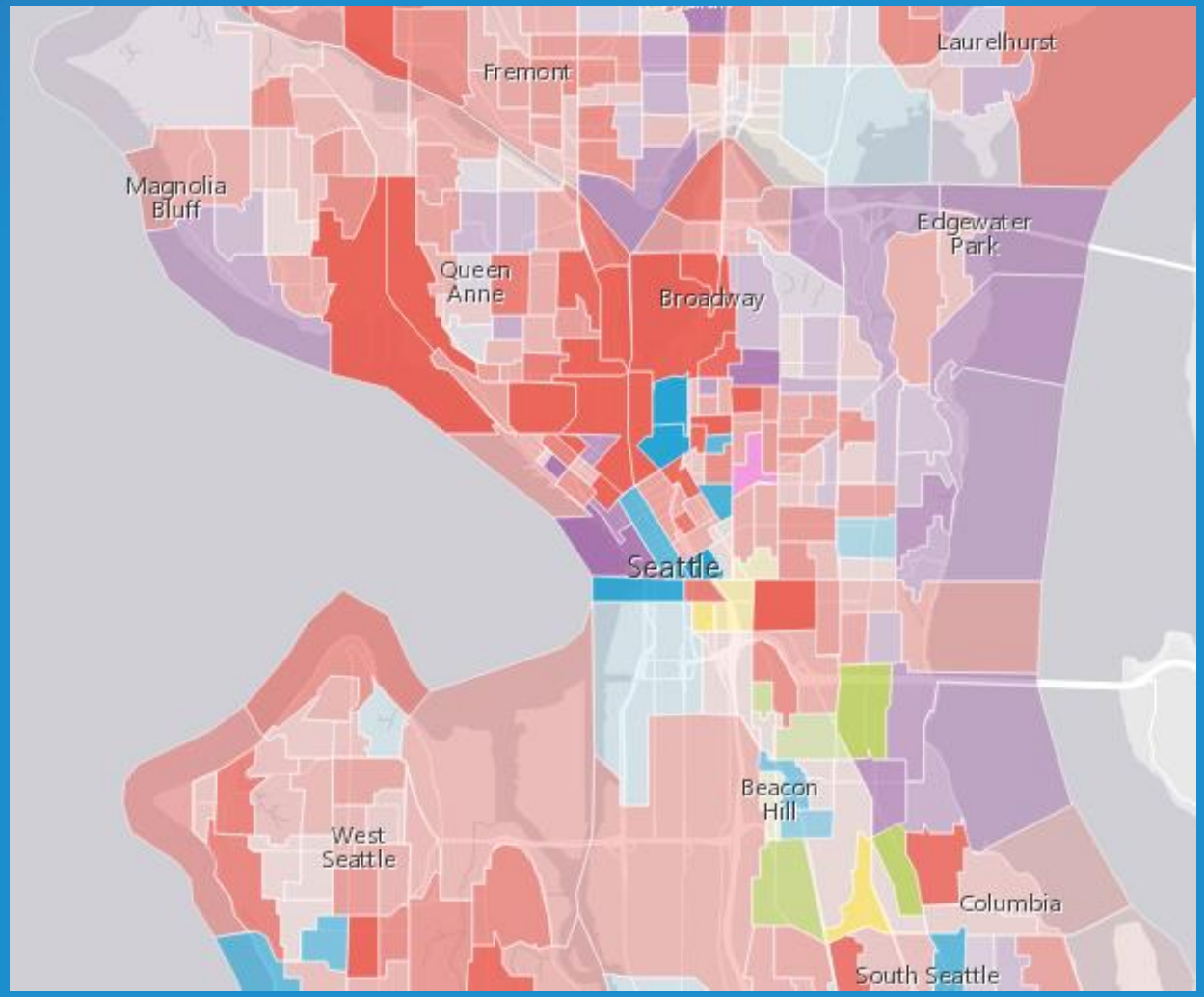
Good for one or two scale levels

Visual Variables: Size (real-world units)

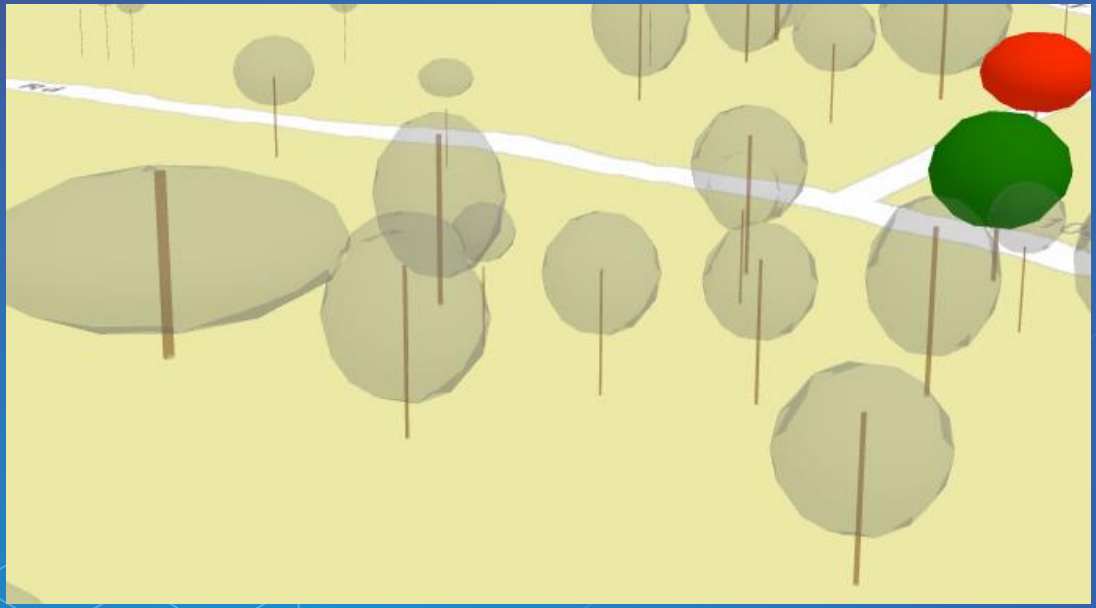
```
var heightVV = {
   type: “size”,
   field: “HEIGHT”,
   valueUnit: “feet”,
   axis: “height”
   // axis is only relevant for
  // ObjectSymbol3DLayer
};

var widthVV = {
   type: “size”,
   axis: “width-and-depth”,
   valueUnit: “inches”
};
```

Visual Variables: Opacity

```
var opacityVV = {
  type: “opacity”,
  field: “EDUCBASECY”,
  stops: [
    {  value: 700,  opacity: 0.1 },
    {  value: 1500,  opacity: 0.9 }
  ]
};
```

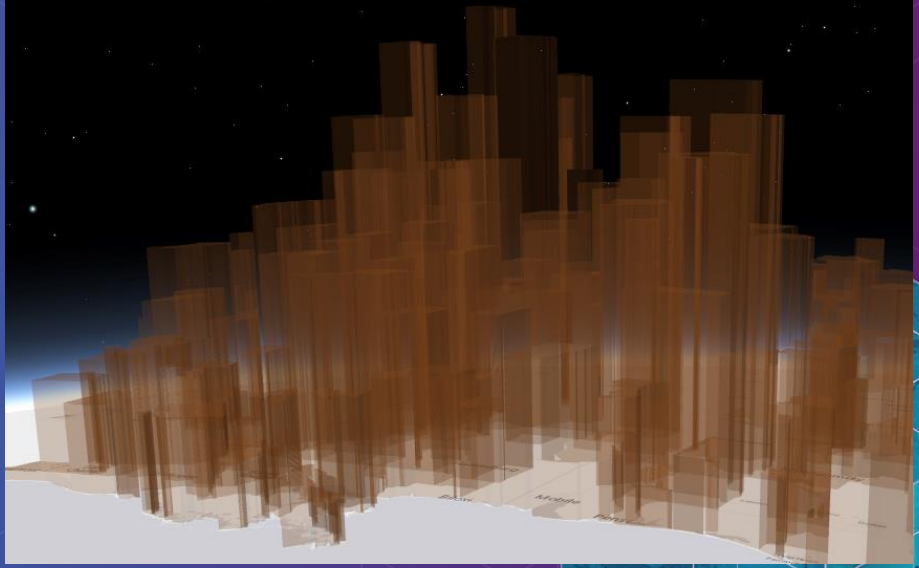


Keep in mind…

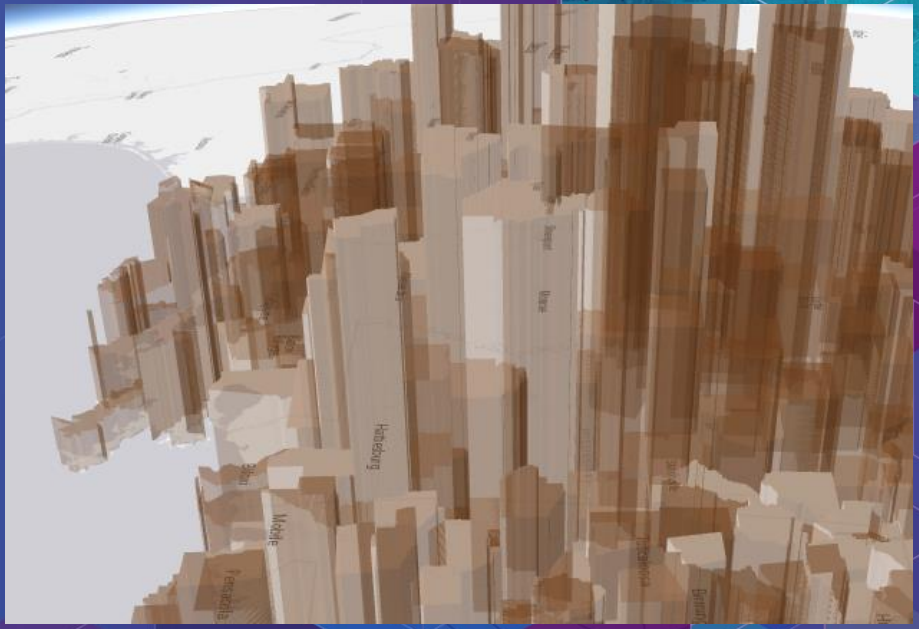


Opacity tends to look better in 3D space when there are relatively few features spread out

Features behind are visible

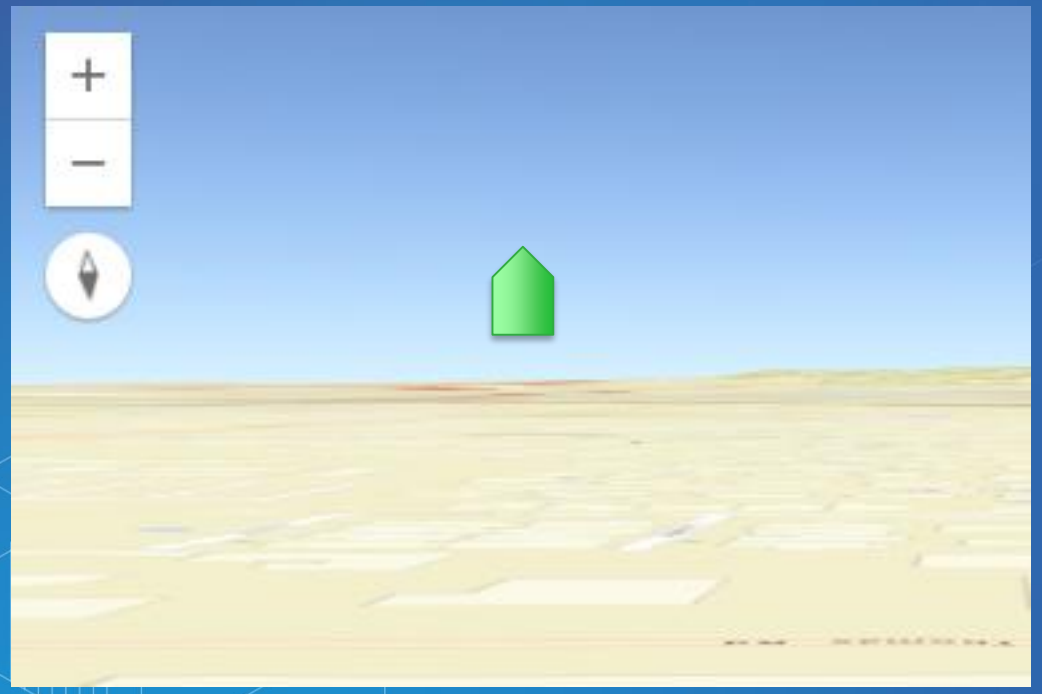


Features behind are not visible



Rotation

Which direction is this symbol pointing?



Easier to interpret in 2D where tilt doesn't interfere

No support for rotation in IconSymbolLayer3D at the moment

Rotation

Object3DSymbolLayer

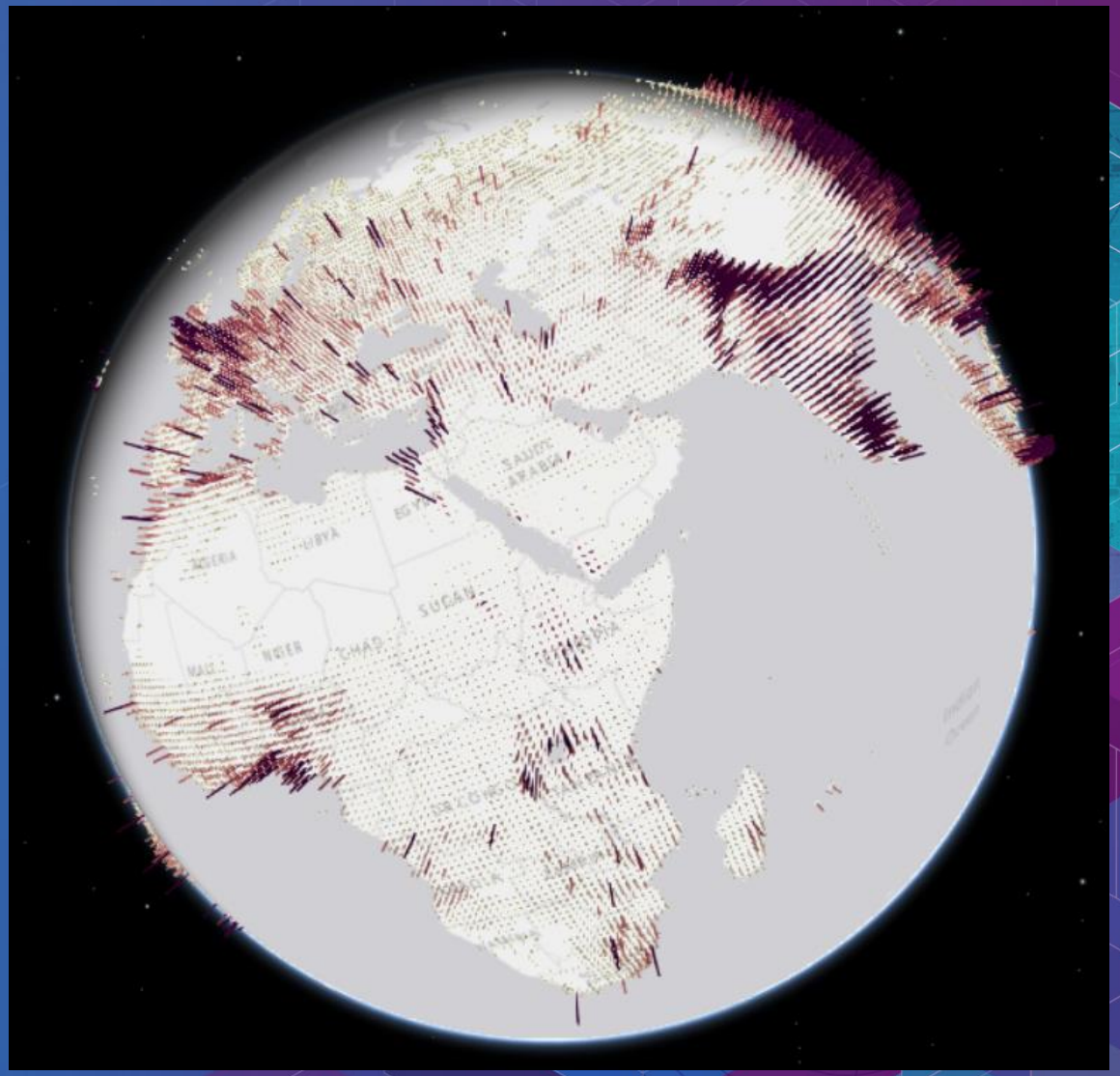
Advanced cases

- Can extend the WebGL engine for complex use cases
- Experimental, not stable

http://richiecarmichael.github.io/ocean/index.html#
https://jkieboom.github.io/devsummit-palm-springs-2016/demos/apps/external-renderer/

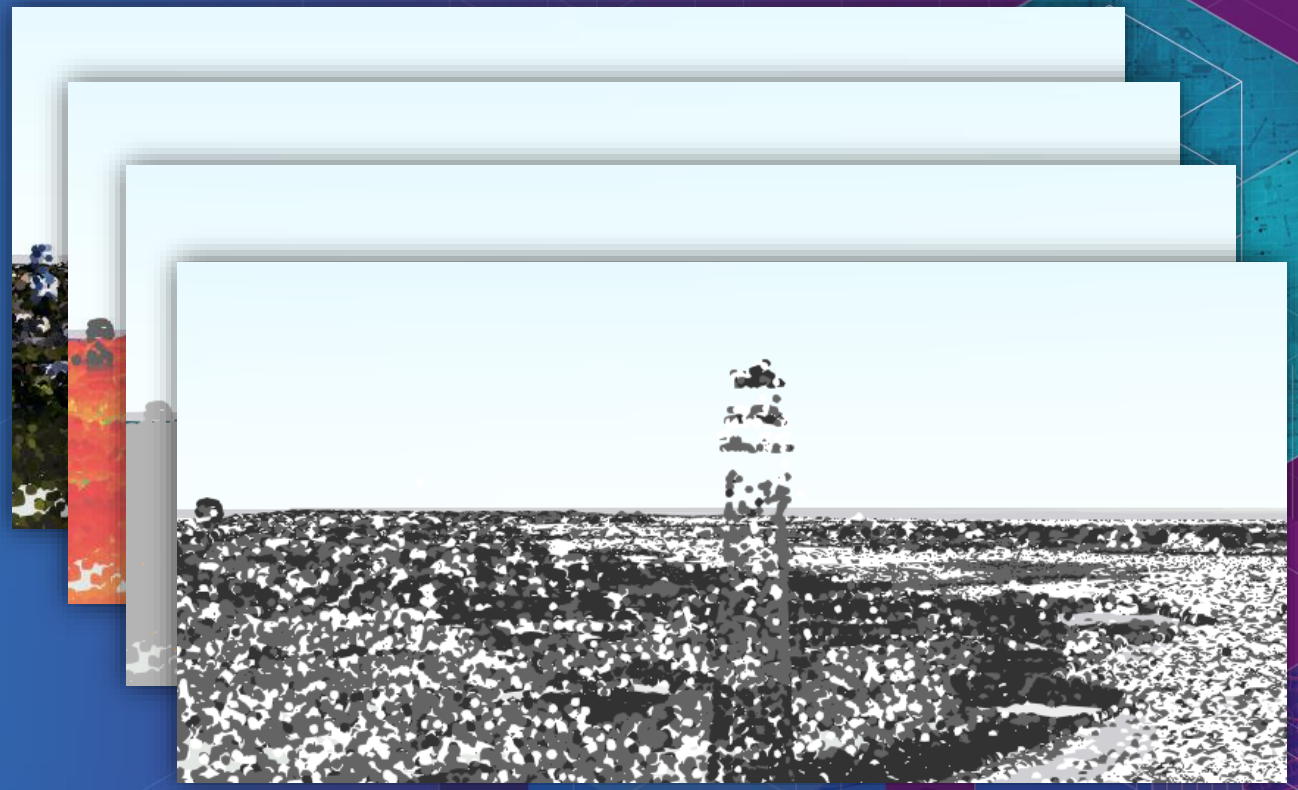
3D Smart Mapping

- No need to guess “good” values.
- Generates visually appealing defaults to thematic visualizations in the Scene View



Point Cloud Layers

- No symbols, but renderers
  - PointCloudRGBRenderer
  - PointCloudStretchRenderer
  - PointCloudUniqueValueRenderer
  - PointCloudClassBreaksRenderer



Overview

- What you can visualize in 3D
- How to do it
- Considerations and pitfalls

Pitfalls

- Too many variables = confusion
- Be aware of scale (generalize when you need to)
- SceneLayer vs. FeatureLayer

Helpful Resources

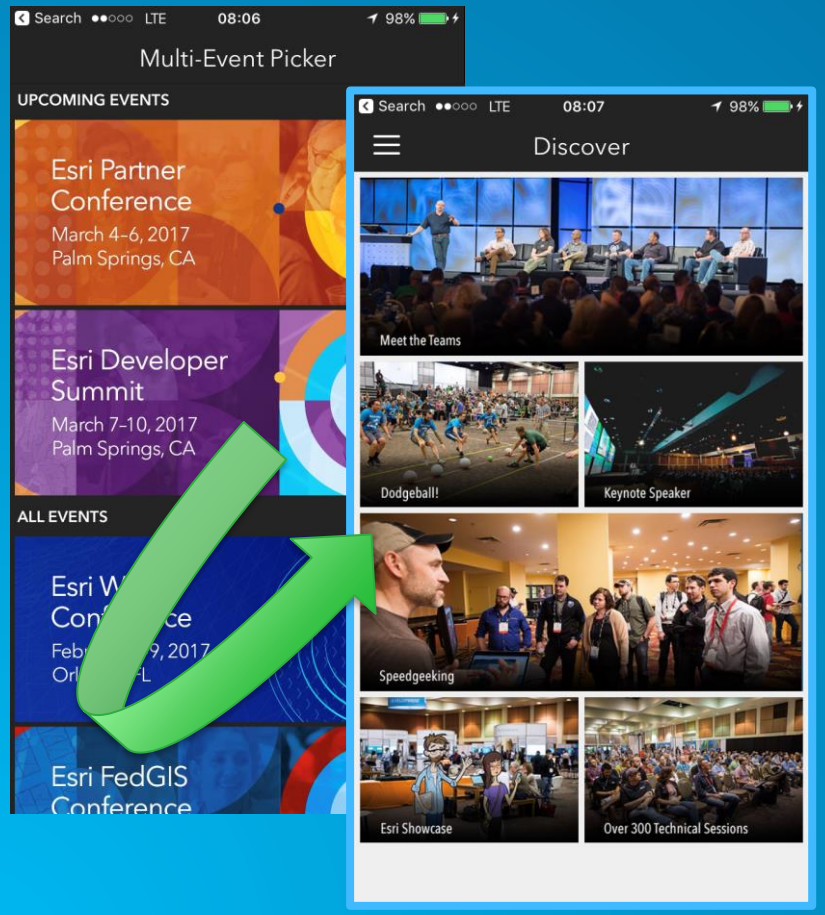
- Get started with visualization
- Symbol Playground

- ArcGIS Blog
  - Icons, lines, fills
  - Objects, paths, extrusion
  - Real world sizes

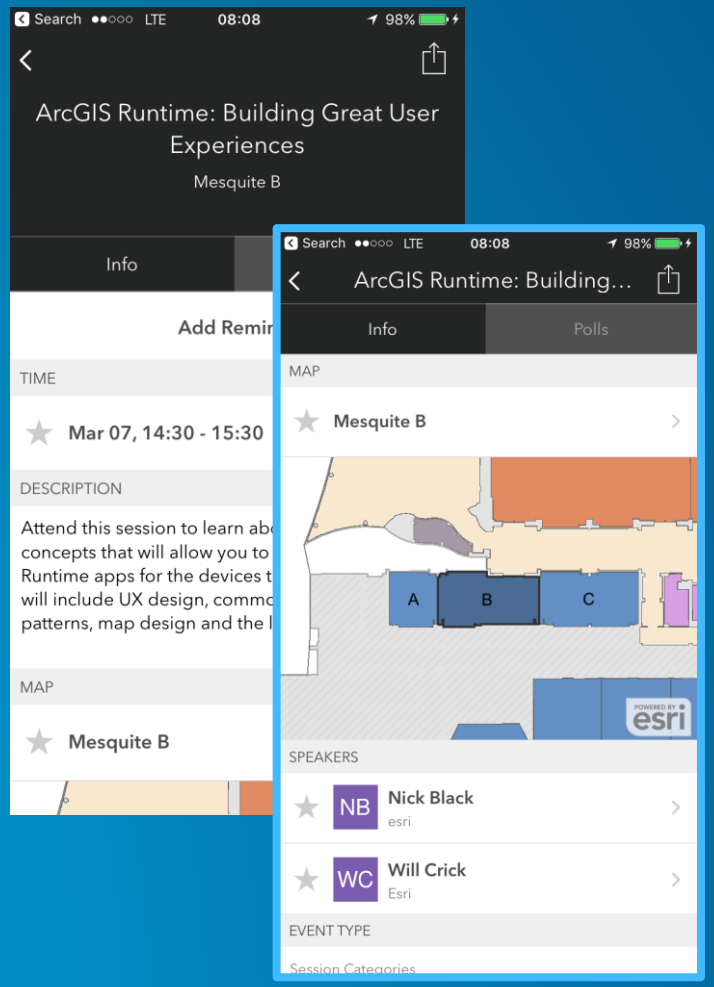
- Documentation
  - Renderer
  - Symbol3D
  - Symbol3DLayer

Please Take Our Survey!

Download the Esri Events app and go to DevSummit

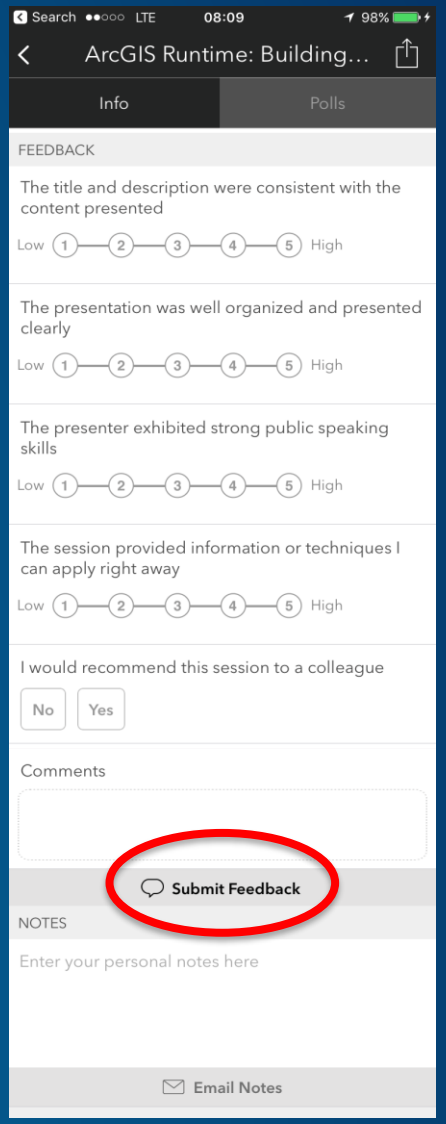


Select the session you attended



Scroll down to the "Feedback" section

Complete Answers, add a Comment, and Select "Submit"



esri

THE SCIENCE OF WHERE