

Getting the most out of



Michael VanDaniker



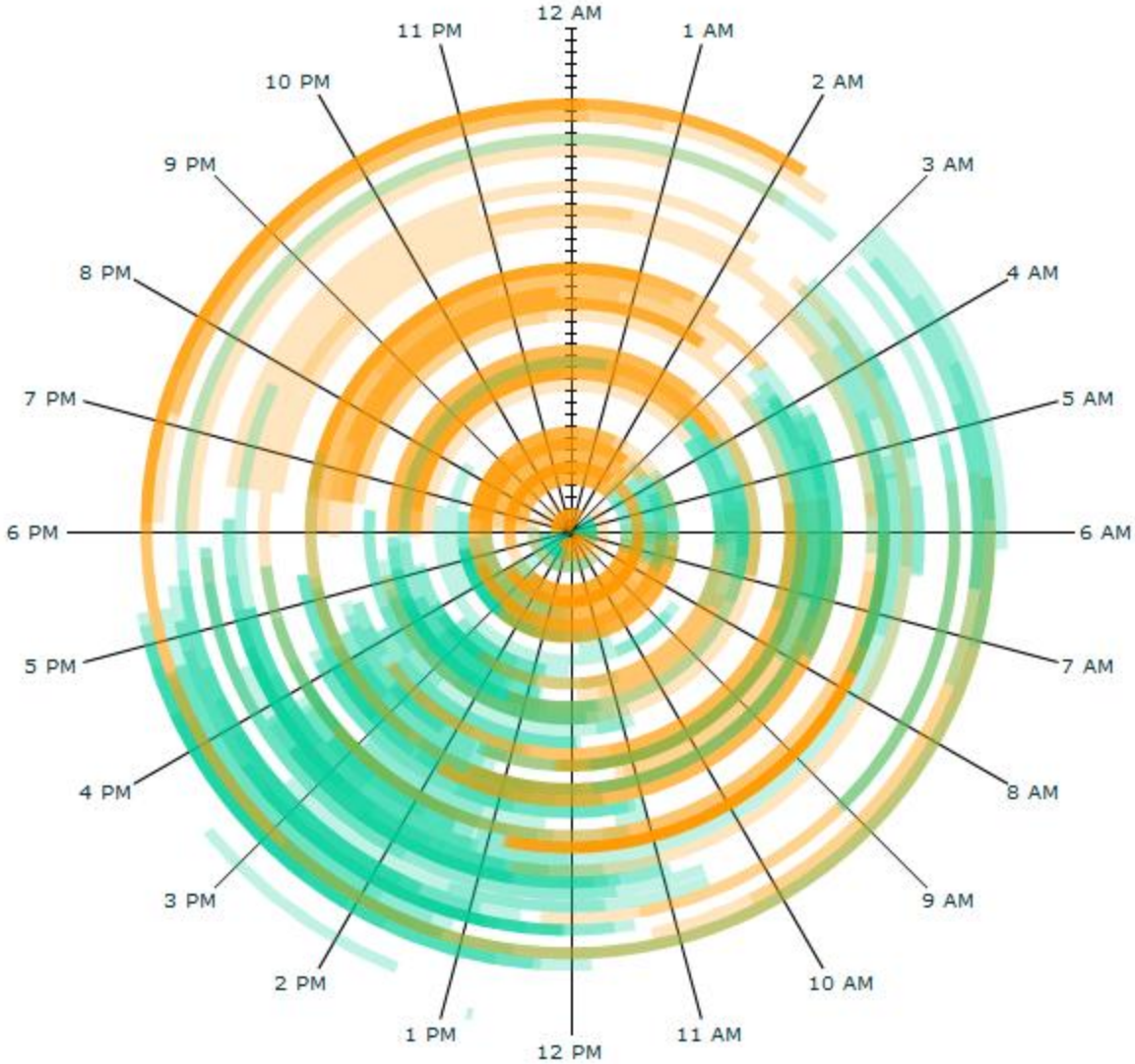
Who am I?

- Data Visualization Manager @ University of Maryland Center for Advanced Transportation Technology Laboratory

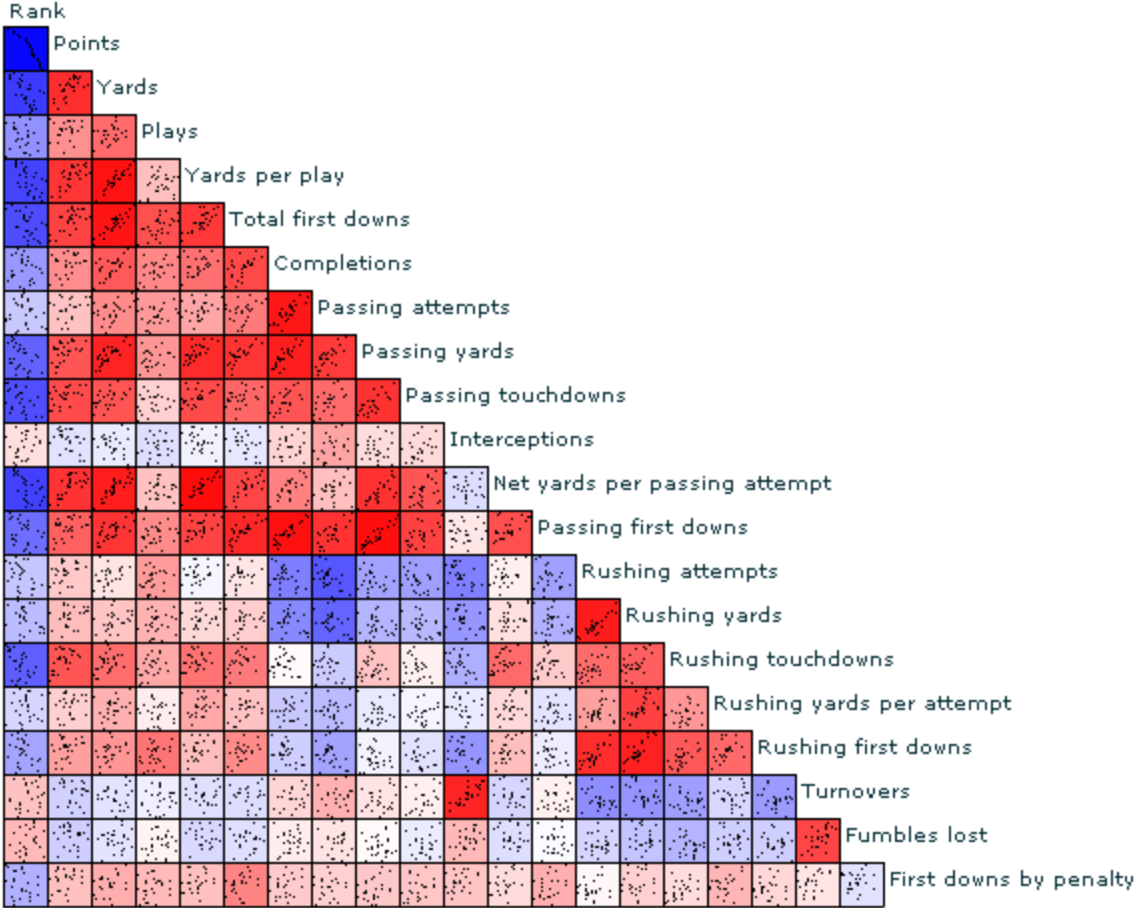
Where we're going today

- Defining Axiis
- How Axiis works
- Writing custom, data-driven layouts
- Adding interactivity
- Using the pre-built visualizations parts
- Tips and tricks

Some visualizations to think about



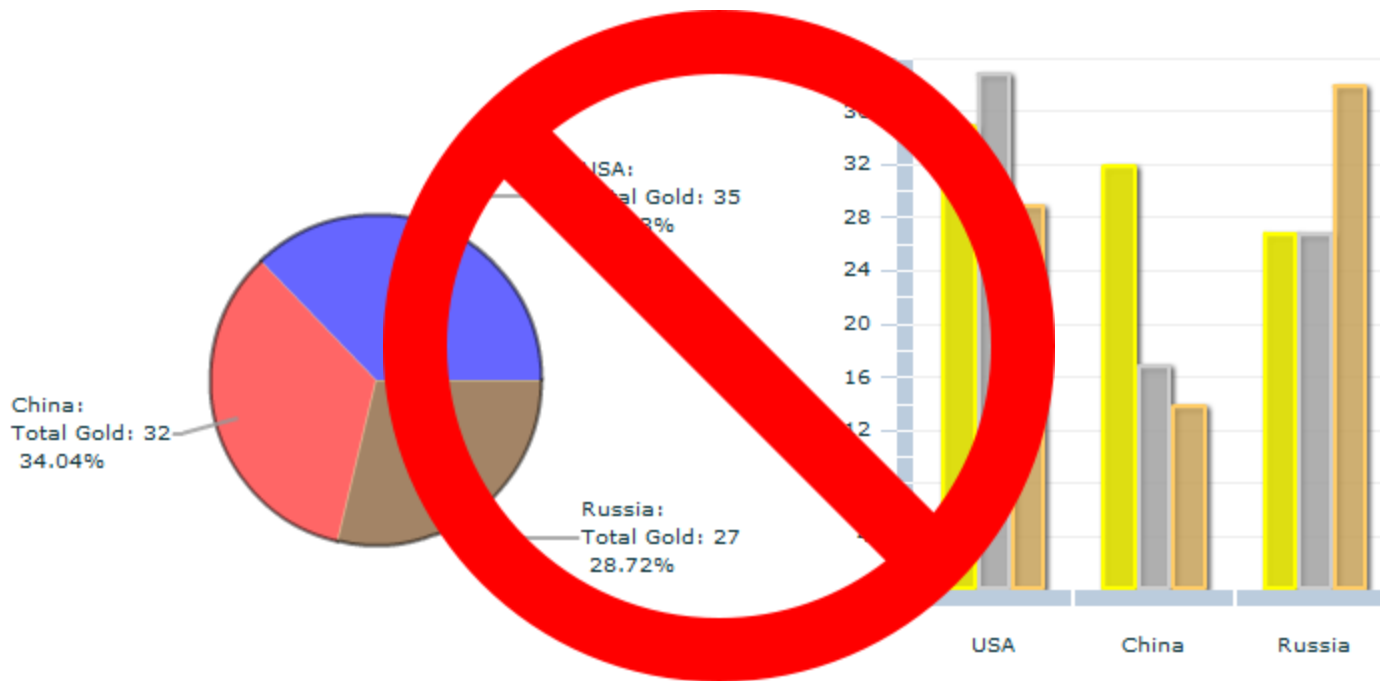
Some visualizations to think about



What is Axiis?

According to almost any resource...

Markup-based data visualization framework



It's not a re-implmentation of the Flex Data Visualization Components and it's not a component library

What is Axiis?

Data-driven Degrafa manipulation engine

```
<axiis:BaseLayout id="hLayout"
  dataProvider="{dataProvider}">
  <axiis:referenceRepeater>
    <axiis:GeometryRepeater>
      <axiis:geometry>
        <degrafa:RegularRectangle
          width="{(hLayout.width / hLayout.itemCount)}"
          height="{hLayout.height}"/>
        </axiis:geometry>
        <axiis:modifiers>
          <axiis:PropertyModifier property="x"
            modifier="{hLayout.width / hLayout.itemCount}"/>
        </axiis:modifiers>
      </axiis:GeometryRepeater>
    </axiis:referenceRepeater>
    <axiis:drawingGeometries>
      <degrafa:RegularRectangle x="{hLayout.currentReference.x}"
        y="{hLayout.currentReference.y}"
        width="{hLayout.currentReference.width}"
        height="{hLayout.currentReference.height}">
        <degrafa:stroke>
          <degrafa:SolidStroke color="0"/>
        </degrafa:stroke>
      </degrafa:RegularRectangle>
```

Degrafa

- Declarative graphics framework

```
graphics.clear();  
graphics.strokeStyle(1, 0, .5);  
graphics.beginFill(0xff0000, .5);  
graphics.drawRect(0, 0, 100, 100);  
graphics.endFill();
```

```
<degrafa:RegularRectangle x="0" y="0" width="100" height="100">  
  <degrafa:stroke>  
    <degrafa:SolidStroke color="0" alpha=".5"/>  
  </degrafa:stroke>  
  <degrafa:fill>  
    <degrafa:SolidFill color="0xff0000" alpha=".5"/>  
  </degrafa:fill>  
</degrafa:RegularRectangle>
```

Axiis in practice

- [Running map](#)
- [Smith chart](#)
- [Buoy monitor](#)
- [Browser statistics visualization](#)

How Axis works

- Many visualizations are based on two rule sets
 - “Layout rules”
 - “Data rules”

How Axis works

- (Get your data ready)
- Drop in your data
- Describe a layout pattern in MXML
- Draw a picture using the data and the rules of the pattern

How Axis works

- Get your data ready
 - Arrays and ArrayCollections
 - Support for transforming XML and CSV

How Axis works

- Drop in your data
 - currentDatum
 - currentIndex
 - currentLabel
 - currentValue

How Axis works

- Describe layout pattern in MXML
 - GeometryRepeater
 - PropertyModifier
- currentReference

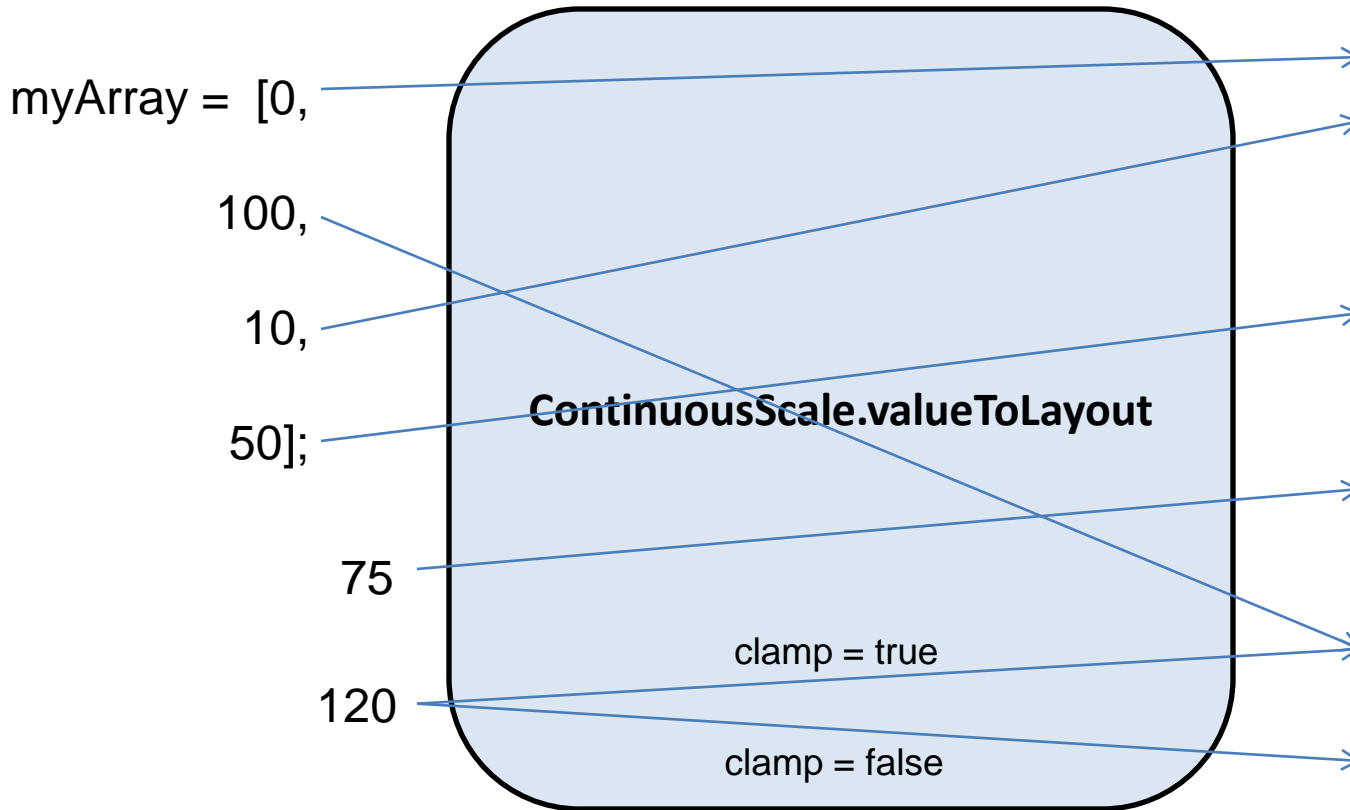
How Axis works

- Draw a picture using the data and the rules of the pattern
 - currentReference
 - currentDatum
 - currentLabel
 - currentIndex
 - currentValue

Scales

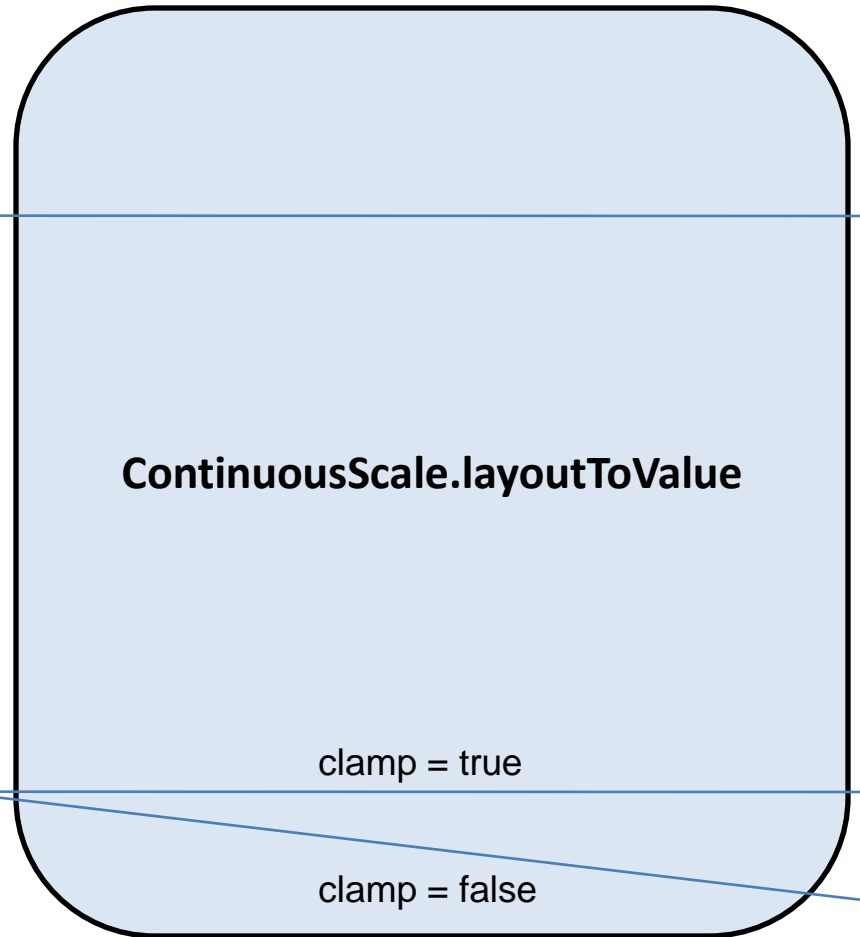
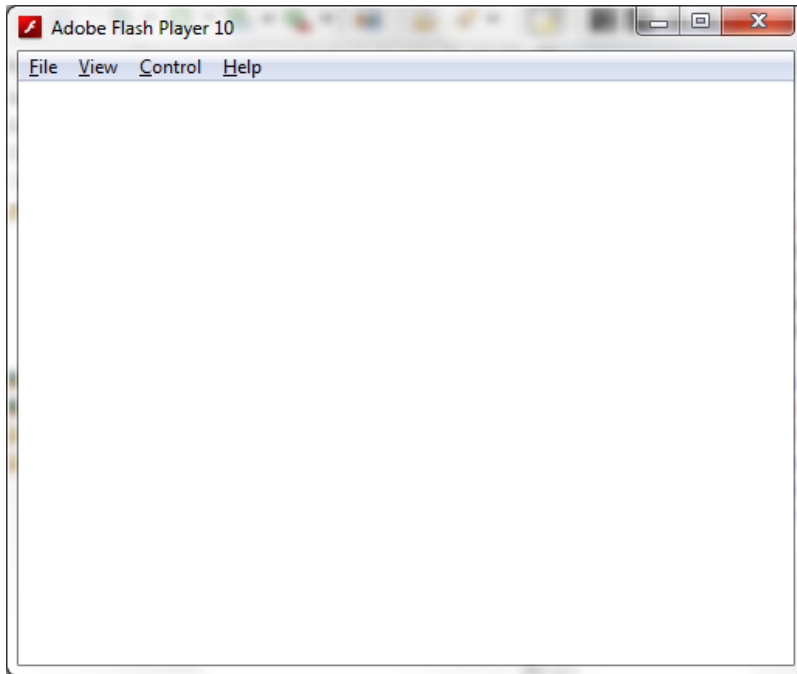
- Mapping a data value to a screen position
 - `valueToLayout(value, clamp, invert) : Number`
- And reverse
 - `layoutToValue(layout, clamp, invert) : *`
- Types of scales
 - ContinuousScale
 - LinearScale
 - LogScale
 - DateTimeScale
 - CategoricalScale

ContinuousScale



ContinuousScale

myArray = [1..100]



25

ContinuousScale.layoutToValue

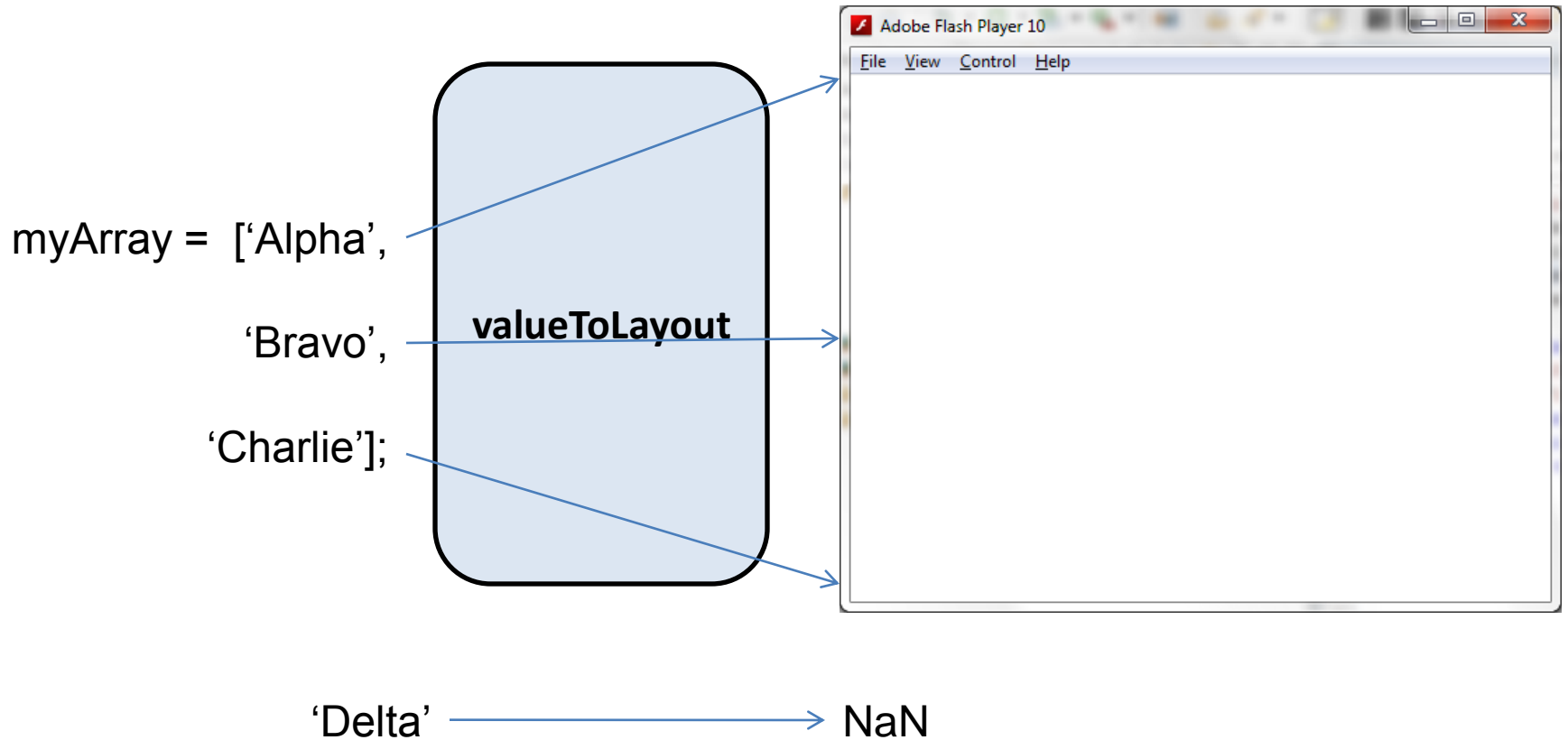
clamp = true

100

clamp = false

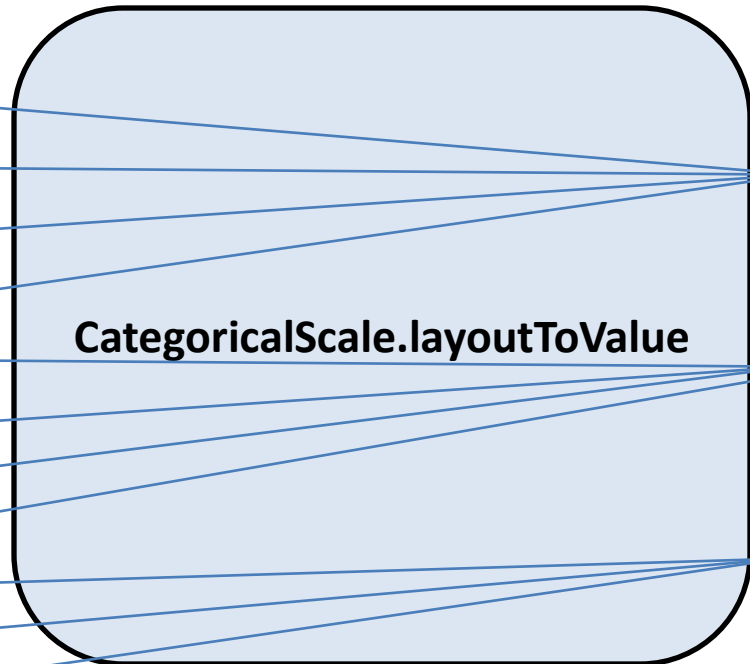
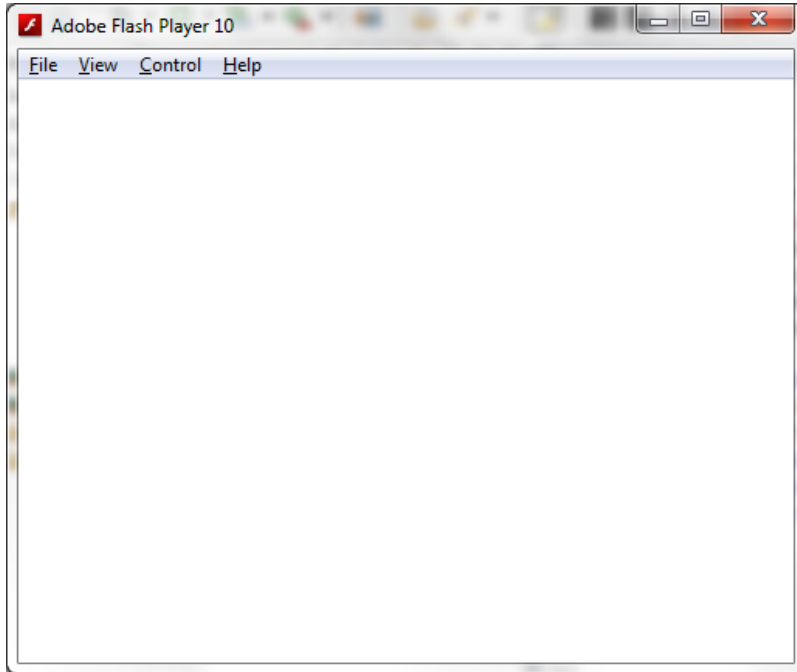
120

CategoricalScale



CategoricalScale

```
myArray = ['Alpha', 'Bravo', 'Charlie'];
```



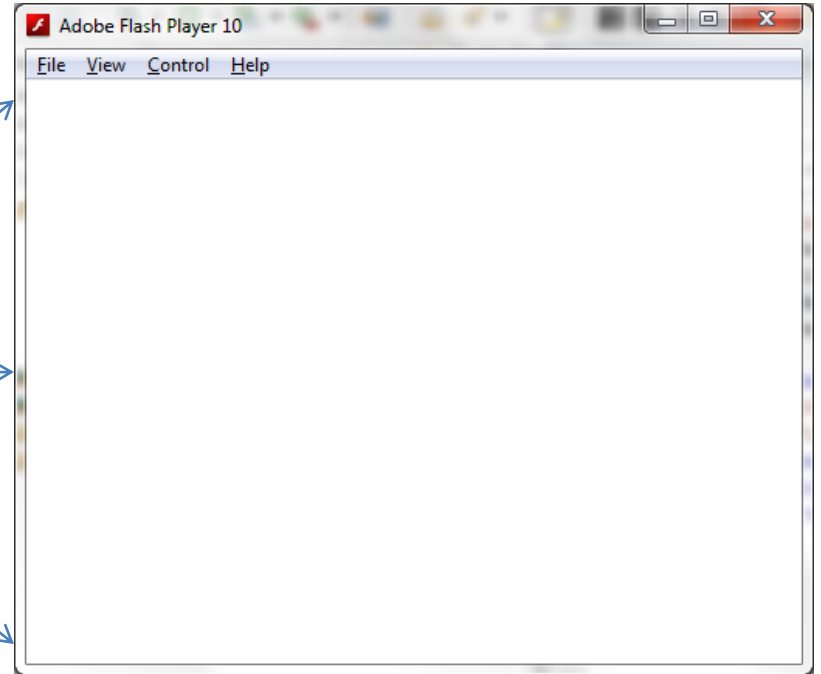
'Alpha'

'Bravo'

'Charlie'

Inverted scales

```
myArray = ['Alpha',  
           'Bravo',  
           'Charlie'];
```



Scales

- Custom ranges
 - min**Value**, max**Value**, min**Layout**, max**Layout**
- computedMinimum
- computedMaximum
- computedAverage

Hierarchical data visualizations

- Child layouts

Adding interactivity

- States
- Data tips
- Per item MouseEvents
 - Debugging with events
- Per layout data filtering
- Conditional geometry properties

Built in visualizations

- Axis
- ConcentricWedgeLayout
- WedgeStack

Data binding shortcuts

- You wouldn't do this:

```
var xyOver2:Number = x * y / 2;  
var xyOver3:Number = x * y / 3;
```

- So don't do this:

```
<mx:Label text="{x * y / 2}"/>  
<mx:Label text="{x * y / 3}"/>
```

Data binding shortcuts

- Use expression classes!
 - BooleanExpression
 - NumericExpression
 - UntypedExpression

```
<axiis:NumericExpression id="product" value="{x * y}"/>
```

```
<mx:Label text="{product.value / 2}"/>
```

```
<mx:Label text="{product.value / 3}"/>
```

Where does Axiis stand?

- Beta 1.1
- Compiled with Flex 3.5
- Comes with a build of Degrafa's Origin branch.
- Currently not 100% compatible with Flex 4

Get Axiis: <http://www.axiis.org/>

Where is Axiis going?

- Revamped pipeline - item renderer pattern
- Animations
- More data processing functionality
- Flex 4 compatibility
- Flex 4 state syntax?
- API likely to change, but you can still use it to do something useful now.

Questions about that?