

Under the Hood of an Advanced Flex Component

Presented by Josh Tynjala

Flex Maniacs

June 26, 2007

Topics

- Intro to Treemaps
- Item Renderers
- Data
- Optimization
- Styles

Quick Intro to Treemaps

Treemaps

- Ben Shneiderman
- University of Maryland
- Hierarchical Data
- Size and Color

Top 10 Reasons : Why Flex wins against Silverlight	List of Animation Packages for AS3	How to make yourself feel old, instantly	Adobe:Mars beta 2 released on June 25, 2007	TweenLite (AS3) - A Lightweight Yet Powerful Tweening En...	Learn More About Flex and PHP This Thursday	::selection
Flex Code Hinting Component	Flex And/Vs. AJAX - A good breakdown by Anthony Franco	AIR Working with PDF files in Adobe AIR Applications	AIR Windowing Applications	New content on the Flash Developer Center	Useless Trivia Post - ColdFusion and the Gov...	Aftermix, Invites, and Remixer
	AIR Examples	Flex3 - Explicit support for Module base application - performance	Flex and Drupal's Node Delete Service	Have I told you how much I love Illudium PU-36 Code ...	Move vs. Flash FLV	Amazing 3D UI Interaction with 2D Flex Components
Why Silverlight Should Fail	Adobe rocks - Many thanks Mike!	Bus Tour Code Repository is open	The White Zone Is For Loading XML	The Dreamweaver Team Is Blogging	Two ColdFusion 8 Documentation Re...	Adobe Mars Beta 2 Released
Chumby, Flash Lite 3, soon	Two more Reasons : Why Flex wins over Silverlight	Two New AIR Apps and Source code Uploaded	Mad HTML Skillz	Streaming Video - but FONTS! How sick is this...	Flex Maniacs 2007 - Day 2 Keynote	Aptana IDE Supports Adobe AIR
			CFUNITE D Flickr	I'm FLEXing my muscles on RAILS!	Help us help you - but wait, there'...	RichFLV now with mp3 import!
						Consecutive onRelease/onPress bug
						The hiring process...

Demo

Item Renderers and Data

Instantiating Renderers

- `mx.core.IFactory`
- `mx.core.ClassFactory`

```
import mx.controls.listClasses.ListItemRenderer;
```

```
itemRenderer = new ClassFactory( ListItemRenderer );
```

```
itemRenderer.properties = { minWidth: 400 }
```

```
var instance:ILListItemRenderer =
```

```
    itemRenderer.newInstance();
```

Expose Common Properties

- Lists
 - Labels
 - Icons
 - Tooltips
- Trees
 - Same as List
 - Open/Closed Icon
 - Indent
 - Depth

Data Conversion Functions

- Examples from List
 - `itemToLabel()`
 - `itemToIcon()`
 - `itemToDataTip()`
 - `itemToItemRenderer()`
 - `itemRendererToIndex()`

TreeMap.itemToLabel()

```
public function itemToLabel( item:Object ):String
{
    if( this.labelFunction != null )
    {
        return this.labelFunction( item );
    }
    else if( item.hasOwnProperty( this.labelField ) )
    {
        return item[ this.labelField ];
    }
    return null;
}
```

Create Interfaces

- ITreeMapNodeRenderer
- IDropInTreeMapNodeRenderer

Other Important Interfaces

- IEventDispatcher
- IDataRenderer
- ISimpleStyleClient
- IFlexDisplayObject

Drop-in Renderers

- Advantages
 - Promotes encapsulation.
 - Provides a common data structure.
 - Lists natively supported by many components.
- Disadvantages
 - Requires more maintenance.
 - Your component isn't natively supported.

IDropInTreeMapNodeRenderer

```
package com.flextoolbox.controls.treeMapClasses
{
    public interface IDropInTreeMapNodeRenderer
    {
        function get treeMapData():TreeMapNodeData;
        function set treeMapData(value:TreeMapNodeData):void;
    }
}
```

TreeMapNodeData

- label
- weight
- color
- toolTip
- uid
- owner

Working with Hierarchical Data

- `ITreeDataDescriptor`
 - `hasChildren()`
 - `getChildren()`
 - `isBranch()`
 - `getData()`
 - `addChildAt()`
 - `removeChildAt()`
 - `getData()`

Using ITreeDataDescriptor

```
private function getItemWeight( item:Object ):Number
{
    var weight:Number = 0;
    if( this.map.dataDescriptor.isBranch( item ) )
    {
        // loop over the children and sum weights recursively
    }
    else
    {
        weight = treeMap.itemToWeight( item );
    }
    return weight;
}
```

Optimization

Cache your renderers.

The display list will thank you.

Structure of a Caching System

```
override protected function commitProperties():void
{
    super.commitProperties();

    // save the current item renderers in an array.
    this.createCache();

    // reuse or create new renderers to display the data.
    this.renderItems();

    // remove extra cached item renderers we don't need.
    this.clearCache();
}
```

Creating the Cache

```
protected function createCache():void
{
    //copy the Array directly
    this._rendererCache = this._itemRenderers.concat();
    this._itemRenderers = [];
}
```

- Or... loop through each item

Requesting a Renderer

```
protected function renderItem():void
{
    var iterator:IViewCursor =
        this._dataProvider.createCursor();
    while( !iterator.afterLast )
    {
        // request a renderer. it may be new or from
        // the cache. we don't care.
        var renderer:ItemRenderer = this.getRenderer();
        renderer.data = iterator.current;
        iterator.moveToNext();
    }
}
```

Using the Cache

```
protected function getRenderer():void
{
    if( this._rendererCache.length > 0 )
    {
        // unshift() removes the first item
        return this._rendererCache.unshift();
    }
    else
    {
        return this._itemRenderer.create();
    }
}
```

Clearing the Cache

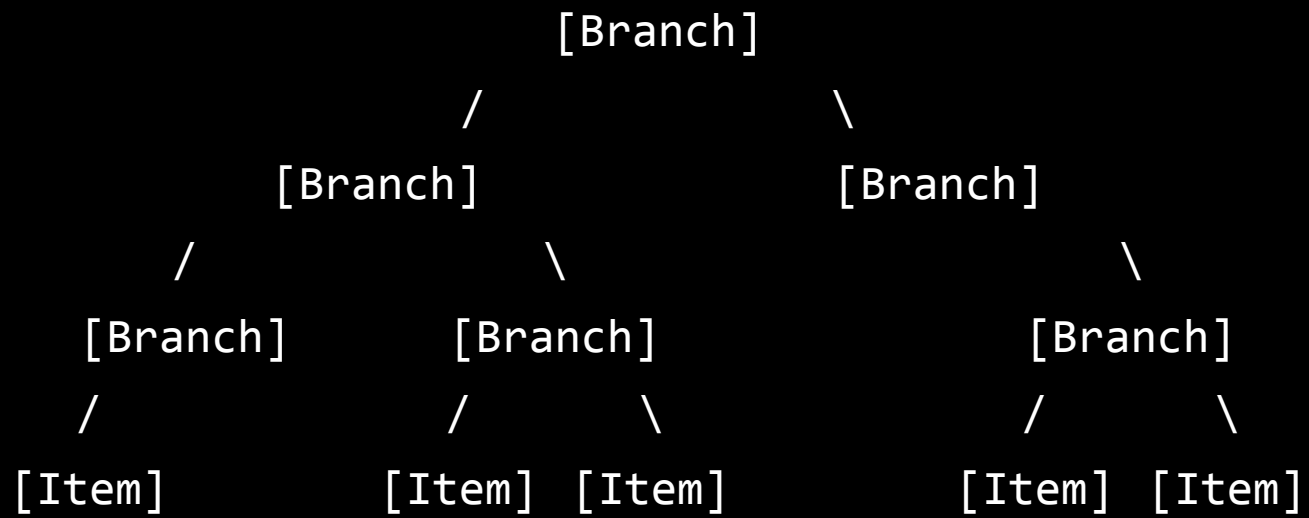
```
protected function clearCache():void
{
    var cacheLength:int = this._rendererCache.length;
    for( var i:int = 0; i < cacheLength; i++ )
    {
        // remove the renderer from the display list
        var renderer:ItemRenderer = this._rendererCache.pop();
        this.removeChild( renderer );
    }
}
```

Cache common calculations.

Give the CPU a break.

Remember this Function?

```
private function getItemWeight( item:Object ):Number
{
    var weight:Number = 0;
    if( this.map.dataDescriptor.isBranch( item ) )
    {
        // this operation could be slow!
    }
    else
    {
        weight = treeMap.itemToWeight( item );
    }
    return weight;
}
```



TreeMap.itemToWeight()

```
public function itemToWeight( item:Object ):Number
{
    var weight:Number = this._cachedWeights[ item ];
    if( isNaN( weight ) )
    {
        if( this.weightFunction != null )
            weight = this.weightFunction( item );
        else if( item.hasOwnProperty( this.weightField ) )
            weight = item[ this.weightField ];
        this._cachedWeights[ item ] = weight;
    }
    return weight;
}
```

Styles

Don't forget style metadata!

- Needed for use in MXML.

```
/**  
 * Documentation is important too.  
 */  
[Style(name="myStyle", type="com.mypackage.MyClass")]
```

Default Styles: A CSS File

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"  
  layout="absolute">
```

```
  <!-- components and stuff here -->
```

```
  <!-- treemap_styles is included in the SWC -->
```

```
  <mx:Style source="treemap_styles.css"/>
```

```
</mx:Application>
```

Default Styles: A Static_INITIALIZER

```
private static function initializeStyles():void
{
    var selector:CSSStyleDeclaration =
        StyleManager.getStyleDeclaration("TreeMap");
    if(!selector) selector = new CSSStyleDeclaration();
    selector.defaultFactory = function():void
    {
        this.backgroundColor = 0xffffffff;
    }
    StyleManager.setStyleDeclaration("TreeMap", selector,
    false);
}
initializeStyles(); //call it immediately
```

Presented by Josh Tynjala

- Email:

josh@zeuslabs.us

- Blog:

www.zeuslabs.us

- TreeMap Component:

code.google.com/p/flex2treemap