Tutorial. How to implement ActionScript from a Umap in a Flex application and adapt the script as required.

At umapper.com you can create a map with pointers and export the code in actionScript format. The options in Umapper are limited to very basic layout and appearance functionalities. This tutorial will mention certain possibilities for extending the functionality by describing:
- possibiblities with Umap
- how to get the code in a richer environment
- where and how to change the actionScript

1. Umap
To create a map is quite easy; Click ‘create’, fill in the form and a map appears where you can place several objects. Choose an object from the menu on the right and place it on the map. Edit the object’s information in the rollover pane at the object.
This map can be exported to ActionScript by using the… ‘export to actionScript’ button!

2. Flex
Flex is a rich internet application creator and can be downloaded from adobe.com. The free command line version is Flex SDK. Flex Builder comes with a whole development environment.
Like Flash it’s built on the Flash Player technology. The difference lies in the models for design and development. While Flash is mainly meant for media and animation, Flex is meant for building complex applications allowing interaction with large amounts of data.
Flex uses .mxml files, which looks like xml, but with predefined tags. ActionScript can be embedded in a mxml file by using the tag <mx:script> or included/imported in a mxml file as a separate file. A Flex project will be compiled to a swf file.
More on Flex at Adobe help:
http://livedocs.adobe.com/flex/3/html/help.html?content=mxml_2.html#179902
This is a nice place to start, it describes how to make a simple application.

A few tips for a quick start with Flex SDK:
Run examples/explorer/build.bat. After this you can open explorer.html to view Flex standard components + codes.

Umap in Flex

Step 1, Umap component.

First of all, it’s necessary to get the Umap component in your Flex application. This component is a .swc file, which is an archive file, consisting of several other files (swf, as, xml) that can be used by Flex.

In Flex SDK you’ll manually need to link the component to the application. In general you create a yourapp-config.xml file in the dir of yourapp where you define the dir of the Umap component in the <include-library> tag.
In the Application Development chapter of the Adobe Help site you can find detailed info on Flex compilers, which includes a part about config files and .swc files.

In Flex builder you just put the file in the libs dir of your Flex project. In case of an ActionScript project you link to the file when creating the project, in the library define tab. The rest of this tutorial will be based on Flex Builder.

Step 2. A basic application.
In your mxml file, this a piece of code you should at least add:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute"
   creationComplete="init()" width="100%">
   <mx:Script>
   <![CDATA[
   import com.afcomponents.umap.gui.*;
   import com.afcomponents.umap.core.UMap;
   import mx.core.UIComponent;

   private var map:UMap;
   private var container:UIComponent;

   public function init() : void
   {
      container = new UIComponent();

      map = new UMap();
      map.setSize(700, 700);

      container.addChild(map);
      mapCanvas.addChild(container);
      container.focusManager.deactivate();

      map.addControl(new MapTypeControl());
      map.addControl(new ZoomControl());
      map.addControl(new PositionControl());

   }

   ]]>
   </mx:Script>

   <mx:Panel id="mapCanvas" width="100%" height="100%" />
</mx:Application>
```

Here a new map is created, added to a UIComonent, and shown on a Panel. The size and some controls are added. If you run the project you'll see your first map.

Example 1: change provider/type of map.
There are different map providers you can use. For instance Google and Yahoo, and Bing as default. They all look different and have different map type show options (map/satellite etc).
For a different provider, add:

```
import com.afcomponents.umap.providers.Providers;
```

and:

```
map.setProvider(Providers.GOOGLE);
```

For a different map type, add:

```
map.setMapType("aerial");
```

Example 2: marker.

You might want to show several markers or other objects on your map. I'll show a simple example how to create one so you'll understand the basics. The styling and appearance of the objects you can get to know yourself when using the umapper and export the actionscript.

Import the marker and the positioning components:

```
import com.afcomponents.umap.types.LatLng;
import com.afcomponents.umap.overlays.Marker;
```

Declare variables:

```
private var marker:Marker;
private var position:LatLng;
```

Inside init(), call:

```
addMarkers();
```

The function:

```
private function addMarkers():void
 {
          marker = new Marker();
          position = new LatLng(52.36,4.89);
          marker.position = position;
          map.addOverlay(marker);
          map.setCenter(position,6);
 }
```

The addOverlay will place the marker on the map, and the setCenter places the default position at the same as the marker's position. The number is the zoom. The higher, the more zoomed in.

Now you know a little bit of the basics it's time to learn how you get the exported ActionScript from Umapper working. This is the code you'll get when you put a marker and an image on the map:

```
import com.afcomponents.umap.core.UMap;
import com.afcomponents.umap.interfaces.IOverlay;
import com.afcomponents.umap.overlays.Layer;
import com.afcomponents.umap.overlays.Marker;
import com.afcomponents.umap.providers.Providers;
import com.afcomponents.umap.styles.DropShadowStyle;
import com.afcomponents.umap.styles.GeometryStyle;
import com.afcomponents.umap.styles.GradientStyle;
import com.afcomponents.umap.styles.IconStyle;
import com.afcomponents.umap.styles.MarkerStyle;
import com.afcomponents.umap.styles.StrokeStyle;
import com.afcomponents.umap.styles.TextStyle;
import com.afcomponents.umap.types.Align;
import com.afcomponents.umap.types.LatLng;
import com.afcomponents.umap.types.Offset;
import flash.display.CapsStyle;
import flash.display.GradientType;
import flash.display.InterpolationMethod;
import flash.display.JointStyle;
import flash.display.LineScaleMode;
import flash.display.SpreadMethod;
import flash.geom.Matrix;
import flash.text.AntiAliasType;
import flash.text.GridFitType;
```

```
import flash.text.TextFieldAutoSize;
import flash.text.TextFormat;
import flash.text.TextFormatAlign;

function addOverlaysToUmap(map:UMap, ...args:Array):void{
	var len:int = args.length;
	for(var i:int = 0; i < len; i++)
	{
		if(args[i] is IOverlay) map.addOverlay(args[i]);
	}
}
function addOverlaysToLayers(layer:Layer, ...args:Array):void{
	var len:int = args.length;
	for(var i:int = 0; i < len; i++)
	{
		if(args[i] is IOverlay) layer.addOverlay(args[i]);
	}
}

//a list of these MarkerStyles:

var _style8:MarkerStyle = new MarkerStyle({radius:10,
icon:"http://umapper.s3.amazonaws.com/ufiles/2009-08/0bf3782"});

var _marker1:Marker = new Marker({name:"Nederland", position:new
LatLng(52.22779942222279, 5.240478515625),
infoStyle:{contentStyle:{html:true}, titleStyle:{html:true}}}, _style6);
var _marker2:Marker = new Marker({id:1, name:"Leidsegracht", position:new
LatLng(52.365838949081315, 4.882543087005615)}, _style8);
var _layer1:Layer = new Layer();
addOverlaysToLayers(_layer1, _marker1, _marker2);

var umap:UMap = new UMap();
addOverlaysToUmap(umap, _layer1);
umap.width = 550;
umap.height = 400;
this.addChild(umap);
```

Unfortunately, just copy-pasting in the script tag doesn't work. The ActionScript has to be put in a basic framework described earlier. Only leave out the grey part and adjust the red;
Copy the red umap initialisation part into init(). It has to look like this:

```
	var umap:UMap = new UMap();
	addOverlaysToUmap(umap, _layer1);
	umap.width = 550;
	umap.height = 400;
	container = new UIComponent();
	container.addChild(umap);
	mapCanvas.addChild(container);
```

Now it will work!
You'll see a difference with the map from Umapper. The map provider and the zoom are back to default

Instructions for Flex builder:

1. Download & install required patches for Flex.
2. Download UMap package from our site, save and unzip it somewhere.
3. Go to Flex Builder and create new ActionScript Project.
4. Give your project a name, e.g. UMapTest
5. Click Next
6. Click Library Path tab
7. Click Add SWC... button and select the UMap_Flex.swc file that is located in the package.
8. Click Finish and the new project will be created.
9. Copy & paste this AS to your document class file:

Code:

```
package {
   import flash.display.Sprite;
   import com.afcomponents.umap.core.UMap;
   import flash.display.StageScaleMode;
   import flash.display.StageAlign;

   public class UMapTest extends Sprite
   {
      private var _map:UMap;

      public function GMapTest()
      {
         _map = new UMap();
         stage.scaleMode = StageScaleMode.NO_SCALE;
         _map.setSize(300, 300);
         addChild(_map);
      }
   }
}
```

10. Run your project. Should be working fine without any errors.

11. Here is a MXML example:

Code:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
layout="absolute"
   creationComplete="onCreationComplete(event)" width="100%">
   <mx:Script>
   <![CDATA[

   import com.afcomponents.umap.gui.*;
   import com.afcomponents.umap.core.UMap;
   import mx.core.UIComponent;

   private var map:UMap;

   public function onCreationComplete(event:Event) : void
   {
      var ref:UIComponent = new UIComponent();

      map = new UMap();
      map.setSize(300, 300);

      ref.addChild(map);
      mapCanvas.addChild(ref);
      ref.focusManager.deactivate();

         map.addControl(new MapTypeControl());
```

```
            map.addControl(new ZoomControl());
            map.addControl(new PositionControl());
      }
      ]]>
      </mx:Script>

      <mx:VBox>
         <mx:Canvas id="mapCanvas" width="400" height="300"
backgroundColor="#352D69" x="6" y="10"/>
      </mx:VBox>
</mx:Application>
```