using omoa cheat sheet

www.omoa-project.org | github.com/omoa

SETUP OMOA

On Windows omoa recommends the FlashDevelop IDE. Download a setup from **flashdevelop.org**
During the setup process you have the option to install the **Flex SDK**. Do it. Download (or clone) the source from https://**github.com/omoa/omoa-as3**.git

SETUP PROJECT

Add the **omoa/src** and **omoa/lib-src** folders to your classpath (check your project settings).

CREATE A SIMPLE MAP

```
map = new Map();
addChild(map);

map.createMapFrame("MyMap");

map.createSpaceModel(
   "borderModel",
   "borders"
   );

map.createLayer(
   "borderLayer",
   map.spacemodel("borderModel"));

map.mapframe("MyMap").addLayer(
   map.layer("borderLayer"));

map.symbolLayer("borderLayer")
   .addSymbol(new VectorSymbol());
```

You may want to make *map* a class variable, which is recommended for all objects you intend to use more often.

GEOMETRY DATA

Geometry data is stored in a *SpaceModel* which implements the *ISpaceModel* interface. Is a collection of *SpaceModelEntity*. Created by *ISpaceModelLoader* implementations like "Shapefile" or "GeoJSON".

Load To create a space model specify ID and filename. By default it expects a Shapefile.

```
var sm:ISpaceModel = map.createSpaceModel(
      "id",
      "url / filename",
      {parameter object},   // optional
      "file format"         // optional
      );
```

"file format" string is name of Loader class. Optional. Examples: Shapefile, GeoJSON. Loader classes live under *org.omoa.spacemodel.loader*
Parameter object is loader specific. See the classdoc. Optional. Example for Shapefile:

```
{id: "ISO_A3", name: "name_latin"}
```

This instructs the loader to look in the "ISO_A3" field of the DBF for the IDs and in the "name_latin" field for the names of the entities.

Load complete After model initialization an *Event.COMPLETE* is fired. You may add an event listener:

```
sm.addEventListener(
   Event.COMPLETE, smComplete);
```

You handle the event this way:

```
private function smComplete(e:Event):void {
   // your code
}
```

Access When initialization is complete the entities may be accessed through an *ISpaceModelIterator* implementation:

```
var i:ISpaceModelIterator = sm.iterator();
while (i.hasNext()) {
   var sme:SpaceModelEntity = i.next();
   trace( sme.id + ": " + sme.name );
}
```

Calling *sm.iterator()* without a parameter returns a *SimpleIterator*. Other iterators live under *org.omoa.spacemodel.iterator*. Some of them have an initialization method *init()*.

Entities Standard properties are *id, name, bounds (BoundingBox), center (Point), attributes (Object)*. Optional properties are *path* (vector data, *IGraphicsPath*), *bitmapData* (raster data, *BitmapData*). All properties are public, so don't mess around.

STATISTICAL DATA

Any non-geometry data is stored in *IDataModel* implementations created by *IDataModelLoader* implementations.

Load _ To create an IDataModel instance from a text or CSV file:

```
var dm:IDataModel = map.createDataModel(
   "id",
   "url / filename",
   {parameter object},
   "Loader Classname"       // optional
   );
```

Crucial for the default text format is the parameter object. It consists of three parts:

```
var fileformat:Object = {
   linebreak:"\r\n",
   separator:"\t",
   hasHeader:true,      // one line only
   properties: [
       { id:'ISO_A3', name:'Country',
       unit:'', type:'ID', column:1 }
   ],
   values: [
       { id:'POP', name:'Population',
       unit:'People Count', type:2,
       column:3 }
   ]
};
```

Load complete After Model initialization is complete an *Event.COMPLETE* is fired.

Access To access the content of a data model you need to create DataDescription first.

```
var d:Datum = dm.getDatum(
   dm.createDescription("GER.FEMALE.POP")
);
trace( d );
```

A description string consists of valid values (including "_" and "*") for each of the property dimensions and the ID of the value dimension.

Data from SME Attributes

To use an attribute of a SpaceModelEntity as statistical data, use *SMEAttributeDataModel* :

```
var dm:IDataModel =
   new SMEAttributeDataModel(
       "pop",      // ID of the new model
       sm,         // SpaceModel
       "POP_2010"  // attribute name );
```

SYMBOL LAYER

Create a SymbolLayer to draw maps (SymbolLayer is the default "Loader Classname"):

```
map.createLayer(
   "id",
   ISpaceModel,
   "Loader Classname" // optional
);
```

Add the layer to a map frame:

```
myMapFrame.addLayer(map.layer("id"));
```

SYMBOL

Symbol / SymbolEntity
properties
manipulators 1

SYMBOLS & DATA

Manipulators 2

BUGS & FEATURE REQUESTS

Go to github.com/omoa/omoa-as3 and switch to the "Issues" tab. Click "New Issue" and fill the form.