This tutorial shows how to use tags in your applications.

A tag is a special 3D model of textured rectangular plane used for scenes to display text or 2D imagery information. Tags always face towards users view, making sure the surface of the tag is shown to the user. There are two types of tags each having different scaling behaviour. The SpriteModelData type will retain its size as it is moved throughout the scene; that is it will scale itself to appear as having constant dimensions. The BillboardModelData type shrinks and grows as it is moved away and towards the eye coordinates respectively, consistent with what is observed in reality.

Creating a Tag

The first step in creating a Tag is to determine which of the two types of behaviour is desired. The next step is to create the appropriate ModelData object by calling its constructor.

```java
BillboardModelData myBillboard = new BillboardModelData(Drawable, width, height);
```

Here we have constructed a billboard Tag called 'myBillboard' that will display any object that is a child of Drawable in the dimensions defined by width and height. Next create a new OAScene passing in the newly created tag as the fourth parameter.

```java
OAScene billboardScene = new OAScene(longitude, latitude, description, myBillboard);
```

Now that the scene is created with the tag wrapped inside, it is added to the list of scenes managed by the DataManager.

```java
getDataManager().addScene(billboardScene);
```

After these three steps your Tag will be displayed at the longitude and latitude specified in the OAScene constructor.
The TutorialARTag application demonstrates the differing behaviours of the two types of data model. An environment has been created that allows individual switching between a SpriteDataModel, BillboardDataModel and a SpriteModelData that takes a string to display on its surface. All tags in this application are created to be the same size and have locations which are parallel to each other. Two buttons are provided that move the camera toward or away from the models.

Upon start up you will notice three toggle buttons near the left of the ARHUD labelled sprite, billboard and custom. Pressing these once will add the data model type displayed on the button to the scene, pressing again removes the data model from the scene. Near the right of the ARHUD are two buttons labelled + and - that allow for incrementing and decrementing the position of the camera respectively. These controls allow you to view the behaviour of the differing data models.

The customDrawable method

A method has been provided that gives an example of how to display a Tag that contains custom text.

```java
public BitmapDrawable customDrawable(String text){
    Bitmap bm = BitmapFactory.decodeResource(getResources(),
          R.drawable.custom).copy(Bitmap.Config.ARGB_8888, true);

    Paint paint = new Paint();
    paint.setStyle(Style.FILL);
    paint.setColor(Color.WHITE);
    paint.setTextSize(100);
    Canvas canvas = new Canvas(bm);
    canvas.drawText(text, 30, bm.getHeight()/2, paint);

    return new BitmapDrawable(bm);
}
```

This method uses a drawable image from the drawable image file as the background to a canvas and writes the text, passed into the method, on top of it. Resizing of the Tag using the parameters in the model data constructor may be required in order to make the text fit correctly. It then returns a bitmapDrawable object that is used as the Drawable in the SpriteDataModel constructor which is further wrapped in the OAScene constructor and added to
the scenes managed by the DataManager.

customScene = new OAScene(-43.522429, 172.583573, "Custom",
    new SpriteModelData(customDrawable("Your text"), 10, 5));
getDataManager().addScene(customScene);

If you have further problem or questions, visit our website (http://www.hitlabnz.org/mobileAR) and post your problem on the Mobile AR Framework Support Forum.