Lesson 15: The Map Widget and Restricting Access to Data

Created by Adriaan Klue, last modified by Jacques Marais on Nov 30, 2017

> *As a System Admin or Shop Owner, I want to see farmer locations on a map.*

Lesson Outcomes

By the end of this lesson you should:

- Know how to use the `<map>` widget
- Know how to restrict which data a user is allowed to see by using the **`@Restrict`** object annotation

Scenario

Suppose we want to enable system administrators and shop owners to see the location of farmers, but with different requirements for the two user roles viewing the data:

- System administrators should be allowed to see the locations of all farmers.
- Shop owners should be allowed to see the locations of all farmers except those residing in the state of Inland, for the state government of Inland has decreed that private location data about its residents should no longer be available to persons below the level of tutorial system administrator.

New & Modified App Files

`./model/roles/Farmer.mez`

`./web-app/images/Globe.png`

`./web-app/lang/en.lang`

`./web-app/presenters/user_managment/FarmerMap.mez`

`./web-app/views/user_management/FarmerMap.vxml`

Placing Restriction Rules on Objects

Instead of effecting viewing restrictions in presenter function logic, it may seem the safer option to mark the persistent object itself as restricted in a particular way. We do this by adding the **`@Restrict`** annotation to the object. According to the requirements mentioned in the *Scenario* section, the `Farmer` object needs to be changed as follows:

```
1  @Restrict("System Admin", all())
2  @Restrict("Shop Owner", notEquals(state, STATES.Inland))
3  @Role("Farmer")
4  persistent object Farmer {...}
```

> Despite the apparent redundancy of using `all()` within a **`@Restrict`** annotation, it is commonly seen in apps written to coexist with a Helium Android Client counterpart, as the annotation also doubled as a way to define the rules by which data synced between the two. The Helium Android Client is no longer supported.

Note that the role being restricted - the first parameter for **`@Restrict`** - is specified using the role name (e.g. `"System Admin"`), and not the object name (e.g. `"SystemAdmin"`). The second parameter for the **`@Restrict`** annotation can be any of the Selector built-in functions (see Quick Reference), or combination thereof, used in the same way as you would when reading from the persistence layer. The above says that system administrators will be allowed to see all farmers, but shop owners will only be able to see farmers whose state is *not* "Inland"**.**

You will have noticed that objects were, throughout the tutorial thus far, universally visible without adding an `all()` restriction anywhere. Indeed, the first line in the snippet above is completely redundant and can be safely removed.

Testing the New Restriction Using the Map Widget

Create a view accessible to system administrators as well as shop owners. Add a `<map>` widget by adding the following to the view:

view snippet

```
<map lat="latitude" long="longitude" refreshIntervalSec
    <collectionSource function="getFarmers"/>
    <markerTitle value="getMarkerTitle"/>
    <markerIcon value="getMarkerIcon"/>
    <markerDesc value="getMarkerDescription"/>
</map>
```

The above says the map will be populated using a collection returned by `getFarmers`, and that an object from this collection has `latitude` and `longitude` attributes that should be used to determine its placement on the map.
The `refreshIntervalSeconds` widget attribute is optional.

presenter snippet

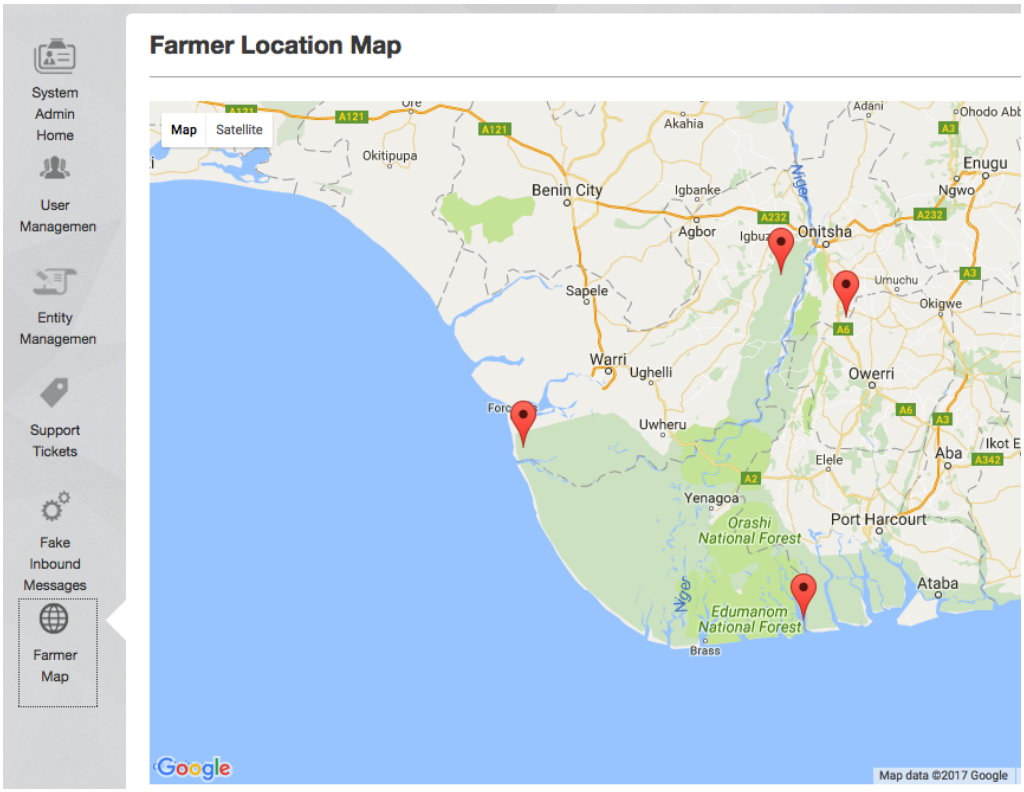
```
string getMarkerTitle(Farmer farmer) {
   return String:concat(farmer.firstName, " ", farmer.
}
string getMarkerDescription(Farmer farmer) {
   return farmer.farmAddress;
}
string getMarkerIcon(Farmer farmer) {
   return null;
}
Farmer[] getFarmers() {
   return Farmer:equals(deleted, false);
}
```

Note that, while the `<markerIcon>` child element is required, we can have its value as **`null`** to fall back to the default pin icon. Clicking on the icon opens a pop-up bearing the specified title and description.

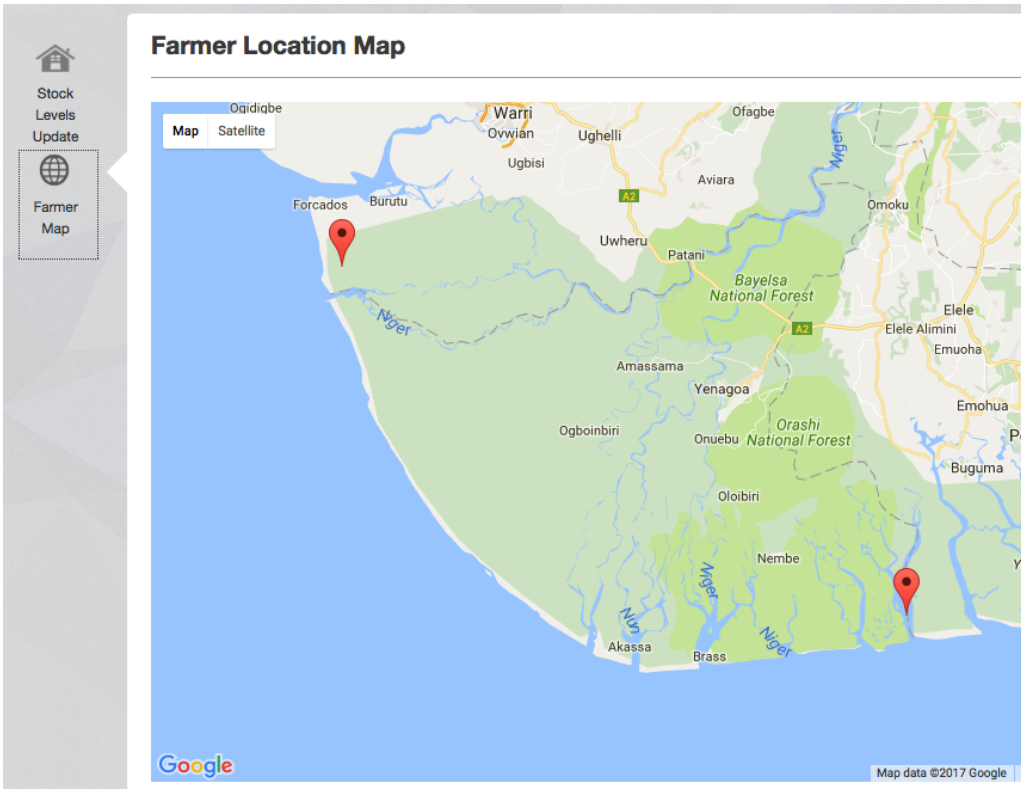
The older map attributes `"latitudeAttribute"` and `"longituteAttribute"` (sic) instead of `"lat"` and `"long"` is also valid.

Create some farmers residing in the Inland state, and some others residing in any of the other states. Also be sure to invite yourself as a `"Shop Owner"`. If you navigate to your new view as a `"System Admin"`, you should now see all the farmers on the map, but switch to using your app as a `"Shop Owner"` and look at the map again, and you should only see those not from the state of Inland.

As `"System Admin"`:



As `"Shop Owner"`:



Lesson Source Code

Lesson 15.zip

No labels