Write your own inheritance resolver

Table of contents

1. Introduction

Tiles definitions are stored in different Tiles definitions files. To avoid code repetition and redundancy, each "higher level" Tiles definition file should inherit from a "lower level" file, so that the "higher level" one inherits all the "lower level" definitions, eventually overriding them. Here comes for help the "inheritance resolver", that, starting from each file, loads all "lower level" files and eventually overrides definitions, starting from the lowest level to the current file.

The `UserDeviceInheritanceResolver` is an implementation of this concept, that gives priority to the user, then to the device. And only user roles and device types are taken into consideration.

2. Why you would need a new inheritance resolver

The reasons for which you need a different inheritance resolver are tightly connected to those that led you to choose a different decider, that is:

- You need some more customization parameters besides the user's role and the device. For example, the locale, or the user's **name**.
- The way the `UserDeviceInheritanceResolver` resolves inheritances is not appropriate for your needs, for example you will need to give priority to the device instead of the user.
- You simply don't want any inheritance.

3. Extending the AbstractInheritanceResolverImpl class

If you want to write your own decider, the most easy way is to extend the `com.free2be.dimensions.tiles.resolver.AbstractInheritanceResolverImpl` class. The other option is to implement the "InheritanceResolver" interface in the same package, but the class above has a simple and useful implementation of common methods.

3.1. Think of an inheritance strategy

If you want to base your inheritance on decisions, the most effective way to resolve inheritance is to build a "decision tree". In other words, organize all suitable decisions in a tree, and the rules to inherit Tiles definition are connected to the tree. See `UserDeviceInheritanceResolver` source for an example.

If inheritance in your case is not based on decisions, then you should invent something :-P

3.2. Implement "resolveInheritances" method

The most important thing to do is implementing the `resolveInheritances` method. Follow the guidelines that you found out in the step before.

You only have to know that `decision2path` contains, as a Map, a correspondence between each decision and each corresponding Tiles definitions file. When you finished resolving everything, in `decision2sequence` must be a Map where the keys are decisions and the values are Lists each including strings, that are the path of Tiles definitions files (those that are present as values in `decision2path`. Again, you should see `UserDeviceInheritanceResolver` source.

4. Configuring Dimensions

Now you have to configure Dimensions. You have to modify the `dimensions-config.xml` file, to use your own inheritance resolver.

```
<inheritance-resolver className="foo.bar.MyInheritanceResolver"/>
```

At the moment, you cannot set customized properties.