HMC Exercise

The hybris Management Console (hMC) is a central view to the hybris Suite and is configured in an extension's hmc.xml file.
As the hmc will be replaced in a future version by a backoffice application you should separate hmc related configuration and functionality in an extension.
In this step you will learn how to do it and how to configure the extension's hmc.xml file to enable the Telcotraining types for the hMC.

Tasks

Administrative change for easier experimentation

To ensure that changes made to the hmc.xml file are reflected in the hMC (without having to load the changes explicitly):

1 Add the following line to your config/local.properties file:
config/local.properties

2
```
hmc.structure.db=false
```

3 Restart your server to ensure the new value is loaded
Note
Restarting as a way to make changes effective has only been tested on a hybris Server

4 For all non-bundled servers, including stand-alone installations of the Apache Tomcat and tcServer, changes in the configuration might require calling ant deploy and restarting the application server.

Tip

If you want to set your default language to something other than English, add hmc.language=<your_language_isocode> to your local.properties file

Create the Telcotraininghmc extension

Create the new extension Telcotraininghmc in the same way as you've did it with Telcotraining. Using a different extension gives you the possibility to disable the hmc connection rather easily.
extension name
Telcotraininghmc
package
com.virtuosa.telco.hmc
template
yempty

- Create Telcotraininghmc extension by invoking extgen in Eclipse
- Add Telcotraininghmc to localextensions.xml
- > ant all
- Import into Eclipse
- Set dependecies to Telcotrainingcore and hmc (Both in Eclipse and extensioninfo.xml file. Please note, extension name for hmc is hmc)
- Remove webmodule from the Telcotraininghmc/extensioninfo.xml
- > ant all

Create the hmc module

1 Edit the extensioninfo.xml file in the Telcotraininghmc extension and add the hmcmodule element below and make sure that it is the last line before the close of the <extension> element.
Telcotraininghmc/extensioninfo.xml
2 <hmcmodule extensionclassname="com.virtuosa.telco.hmc.TelcotrainingHMCExtension"/>
3 Now create a new source folder hmc/src in the Telcotraininghmc project (add it to your build path!!!).
4 Create the following Java class in that folder and the package com.virtuosa.telco.hmc
Telcotraininghmc/hmc/src/com/virtuosa/telco/hmc/TelcotrainingHMCExtension.java
5 package com.virtuosa.telco.hmc;

```
import de.hybris.platform.hmc.AbstractEditorMenuChip;
import de.hybris.platform.hmc.AbstractExplorerMenuTreeNodeChip;
import de.hybris.platform.hmc.EditorTabChip;
import de.hybris.platform.hmc.extension.HMCExtension;
import de.hybris.platform.hmc.extension.MenuEntrySlotEntry;
import de.hybris.platform.hmc.generic.ClipChip;
import de.hybris.platform.hmc.generic.ToolbarActionChip;
import de.hybris.platform.hmc.webchips.Chip;
import de.hybris.platform.hmc.webchips.DisplayState;

import java.util.*;

/**
 * Created by abhay on 03/11/15.
 */
public class TelcotrainingHMCExtension extends HMCExtension {


    private static final String RESOURCE_PATH = "com.virtuosa.telco.hmc.locales";

    @Override
    public List<EditorTabChip> getEditorTabChips(DisplayState displayState,
         AbstractEditorMenuChip abstractEditorMenuChip) {
        return Collections.EMPTY_LIST;
    }

    @Override
    public List<AbstractExplorerMenuTreeNodeChip>
         getTreeNodeChips(DisplayState displayState, Chip chip) {
        return Collections.EMPTY_LIST;
    }

    @Override
    public List<MenuEntrySlotEntry> getMenuEntrySlotEntries(DisplayState
         displayState, Chip chip) {
        return Collections.EMPTY_LIST;
    }

    @Override
    public List<ClipChip> getSectionChips(DisplayState displayState, ClipChip
         clipChip) {
        return Collections.EMPTY_LIST;
    }

    @Override
    public List<ToolbarActionChip> getToolbarActionChips(DisplayState
         displayState, Chip chip) {
       return Collections.EMPTY_LIST;
    }

    @Override
    public ResourceBundle getLocalizeResourceBundle(Locale locale) throws
         MissingResourceException {
       return null;
    }

    @Override
    public String getResourcePath() {
       return RESOURCE_PATH;
    }
}
```

6 Run ant all and refresh the workspace. You should be able to see the hmc folder

Add TelcotrainingCore to the hMC

1 Create the file Telcotraininghmc/hmc/resources/hmc.xml: `Telcotraininghmc/hmc/resources/hmc.xml`

2 <?xml version="1.0" encoding="ISO-8859-1"?>

```
<configuration xmlns="hybris.de/schemas/hmc/generic"
          xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
          xsi:schemaLocation="hybris.de/schemas/hmc/generic hmc.xsd">


    <!– Configuration for extension acceleratorcms –>

    <explorertree>
       <group name="Telcotraininggroup"  mode="append">
          <typeref type="Customer"/>
       </group>
    </explorertree>
```

```
  <type name="Customer" mode="append">
    <organizer mode="append">
      <search mode="append">
        <condition attribute="dob"></condition>
      </search>
      <result>
        <listview>
          <itemlayout>
            <attribute name="dob"/>
          </itemlayout>
        </listview>
      </result>
      <editor mode="append">
        <tab name="tab.Telcotraining" position="0">
          <section name="section.Telcotraining">
            <listlayout>
              <attribute name="dob"/>
            </listlayout>
          </section>
        </tab>
      </editor>
    </organizer>
  </type>



</configuration>
```

1 Invoke ant all
2 If not already running, start hybris
3 Navigate to the hMC: http://localhost:9001/hmc/hybris and note that a new node is visible in the tree for our new extension Telcotraining
5 Key points to note:
    a We have added a reference to Customer in the hMC's explorer tree (the tree displayed on the left side of the hMC)

b We specify an organizer tag for Customer describing how the Organizer's three main areas, i.e. search, result and editor areas should appear.

c If we have multiple tabs, content under <essentials> tag will always appear on the top of each.

d The build process combines the hmc.xml files found in each extension into one large hmc.xml in platform/ext-platform-optional/hmc/web/webroot/WEB-INF/classes/de/hybris/platform/hmc/hmc.xml

e The hMC layout is then determined at run-time by the contents of this master hmc.xml file

f Because you specified hmc.structure.db=false in config/local.properties, hybris will load the hmc configuration from this file rather than storing it in the database

6 Note that the Telcotraining node name appears as [Telcotraininggroup]. Expand this node – the child node's name may can vary, depending on whether or not you have already performed the type-system localizations in the previous trail step, Trail ~ New Data Model. (If you have *not*, the child node appears as [Customer] - otherwise, the hMC defaults to the localized type's 'name' attribute). Either way, we will override those defaults and localize the names of these hMC nodes next.

Localize the hMC labels

1 Create the following file and folders: Telcotraininghmc/hmc/resources/com/virtuosa/telco/hmc/locales_en.properties

2 telcotraininggroup=Telco Training

type_tree_Customer=Customer

tab.telcotraining=Telco Training

section.telcotraining=Telco Training

3 Create the following files: Telcotraininghmc/hmc/resources/com/virtuosa/telco/hmc/locales_de.properties

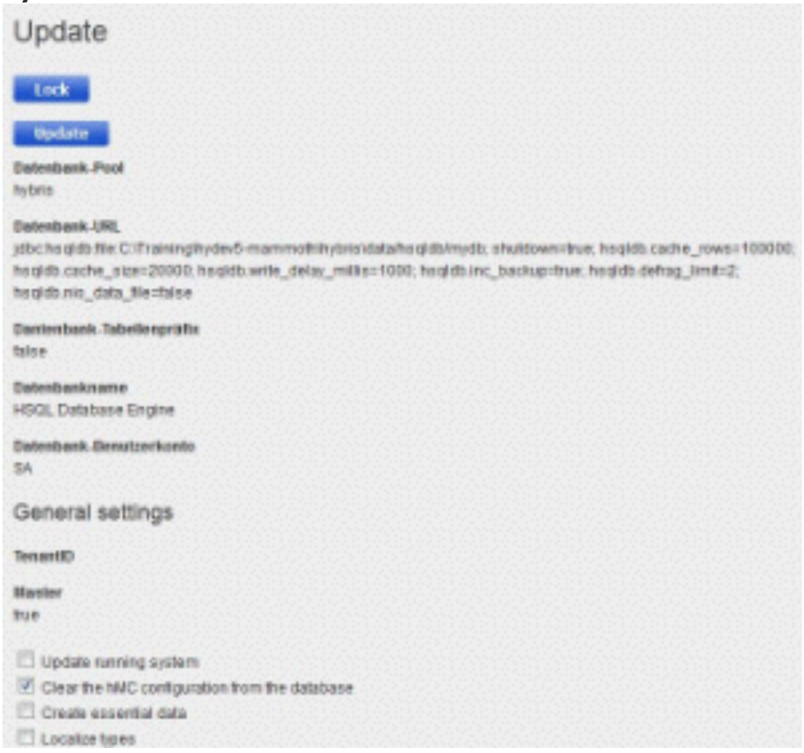
telcotraininggroup=Das Telco Training

```
type_tree_Customer=Customer

tab.telcotraining=Das Telco Training

section.telcotraining=Das Telco Training
```

4 Make sure your project structure looks like this
5 Reinvoke the Ant All task from within Eclipse.
6 If not already running, restart hybris
7 Explore how Telcotraining tree elements are now localized in the hMC at http://localhost:9001/hmc/hybris.
8 We recommended above to add the entry hmc.structure.db=false to force the hMC to reload its configuration. Otherwise you would have to tell hybris to reload the hMC:



Update

Datenbank-Pool
hybris

Datenbank-URL
jdbc:hsqldb:file:C:\Training\hydev5-mammoth\hybris\data\hsqldb\mydb; shutdown=true; hsqldb.cache_rows=100000; hsqldb.cache_size=20000; hsqldb.write_delay_millis=1000; hsqldb.inc_backup=true; hsqldb.defrag_limit=2; hsqldb.nio_data_file=false

Datenbank-Tabellenpräfix
false

Datenbankname
HSQL Database Engine

Datenbank-Benutzerkonto
SA

General settings

TenantID

Master
true

- ☐ Update running system
- ☑ Clear the hMC configuration from the database
- ☐ Create essential data
- ☐ Localize types

9 Note that the localizations we performed here apply only to hMC-specific view elements, such as the Telcotraining node in the left-hand tree. Other localizations, such as the names of the Customer attributes (despite being visible from within the hMC search windows) come from within the hybris type system via a system-wide data model, and are not affected by hMC localizations. For example, attribute localization strings are used in hybris Cockpits as well, whereas hMC localization strings are not.

Summary

In this step you should have learned

- that you should decouple hMC functionality from your regular extensions
- that the hMC is quite useful for viewing your extension's Data Model
- setting hmc.structure.db=false in local.properties will ensure hMC text localizations are visible after a rebuild. However, the hMC can be refreshed manually