MetWare, SKOS and Java Server Faces

Published April 30, 2008

Citation

Willighagen, E. (2008, April 30). MetWare, SKOS and Java Server Faces. *Chem-bla-ics*. https://doi.org/10.59350/13cvj-rgs34

Keywords

Metware, Java, Bioclipse, Skos

Abstract

The MetWare components are slowly coming together. The RAW data upload facility prototype went into beta stage, while the SKOS has proven really useful for various things.

Copyright

chem-bla-ics

The MetWare components are slowly coming together. The RAW data upload facility prototype went into beta stage, while the SKOS has proven really useful for various things.

Because of being compatible with various Java libraries and tools, we decided some time ago to use Java. We also wanted to start of with a HTML GUI to MetWare, which led us to Java Server Faces. Not being so fond of Tomcat (e.g. use by the NMRShiftDB), I was not sure how that would turn out, but Steffen was rather positive about it. And I like it :)

The source code for this screenshot is rather simple:

```
<table>
  <tr valign="top">
    <td><br/>Monoisotopic Mass:<br/>
        min=<inputText id="monomassmin"
value="#{metidMetaboliteQuery.monoisotopicMassMin}"/><br/>
        max=<inputText id="monomassmax"
value="#{metidMetaboliteQuery.monoisotopicMassMax}"/><br/>
        <commandButton value="Search" id="submit"
action="#{metidMetaboliteQuery.search}"/>

        <!--  search results -->
        <p><dataTable value="#{metidMetaboliteQuery.results}" var="mbolite">
           <facet name="caption">Search Results...</facet>
           <column>
             <facet name="header"><outputText value="Monoisotopic mass"/></
facet>
             <outputText value="#{mbolite.monoisotopicMass}"/>
           </column>
           <column>
             <facet name="header"><outputText value="InChIKey"/></facet>
             <outputText value="#{mbolite.inchikey}"/>
           </column>
         </dataTable></p>
       </td>
```

chem-bla-ics

```
    <td width="25%">
      <b><outputText id="tabelName" value="#{metidMetabolite.prefLabel}"/>:</
b>
      <br/>
      <outputText id="tabelDef" value="#{metidMetabolite.definition}"/>
    </td>
  </tr>
</table>
```

Key concept here is that JSF uses Java Beans, which are referred to in the above example with code like `#{bean.field}` for bean fields, and with `#{bean.method}`, assuming a bean exists with `getField()`, `setField()` and `method()`. The `<h:outputText>` stuff is JSF to work out bean details and will create HTML in the output. As really brief intro.

The Metware Beans

It is clear that java beans for Metware would be useful, and this is what I have been working on for the last few weeks. The relevant beans for the above example are automagically created from the SKOS, complemented with extra bits of RDF for the additional details, like field data type, mapping to SQL tables, and an example value. This all works very smoothly (the code to `load()` and `save()` into the SQL database is automatically generated too!) as you can see in the above example. The screenshot shows matches from a (local) live SQL metabolomics database. The text on the right side is directly taken from the SKOS.

Now, the bean library allows integration with other tools too, though this cannot be found in our current roadmap. But, for example, I have been thinking about a simple Bioclipse wrapper around these beans. What is on our roadmap involves workflows for metabolomics.