Module Definitions 2

Overriding a module definition will allow you to add functionality to the module and override existing functionality. You should always put your files last in the list properties to facilitate the cases where you want to override functionality. While the documentation below assumes that you are overriding the module bean for a delivered module, a very similar process would apply when adding a module. You would just have a heck of a lot more information to change after copying in a template module definition

1. Add your override Spring file for the module.
2. Copy the original module definition into your file. It will have a id like "xxxModule" and a class of `org.kuali.core.KualiModule`. See the `chartModule` bean in `work/src/org/kuali/module/chart/OJB-repository-chart.xml` for an example.
3. Make any necessary changes.
    - Change the module user classes
    - Change authorizer classes or package prefixes
    - Add data dictionary packages
    - Add OJB repository files
    - Add DWR files
    - Add batch jobs
      For example, here we have added to `dataDictionaryPackages`, `databaseRepositoryFilePaths`, and `scriptConfigurationFilePaths`.

edu.sampleu.kuali.module.chart.SampleuSpringBeansChart.xml

```
<bean id="chartModule" class="org.kuali.core.KualiModule">
        <property name="moduleId" value="chart" />
        <property name="moduleName" value="Chart Of Accounts" />
        <property name="moduleCode" value="CA" />
        <property name="moduleUserService" ref="chartUserService" />
        <property name="initializeDataDictionary" value="true" />
        <property name="moduleUserRule">
            <bean class="org.kuali.module.chart.rules.ChartUserRule"
singleton="false" parent="maintenanceDocumentRuleBase" />
        </property>
        <property name="moduleUserPreRules">
            <null />
        </property>
        <property name="moduleAuthorizer">
            <bean
class="org.kuali.kfs.authorization.KfsModuleAuthorizerBase">
                <property name="packagePrefixes">
                    <list>
                        <value>org.kuali.module.chart.</value>
                    </list>
                </property>
            </bean>
        </property>
        <property name="dataDictionaryPackages">
            <list>
                <value>org/kuali/module/chart/datadictionary</value>

<value>edu/sampleu/kuali/module/chart/datadictionary</value>
            </list>
        </property>
        <property name="databaseRepositoryFilePaths">
            <list>

<value>org/kuali/module/chart/OJB-repository-chart.xml</value>

<value>edu/sampleu/kuali/module/chart/sampleu-repository-chart.xml</value>
            </list>
        </property>
        <property name="scriptConfigurationFilePaths">
            <list>
                <value>org/kuali/module/chart/dwr-chart.xml</value>

<value>edu/sampleu/kuali/module/chart/sampleu-dwr-chart.xml</value>
            </list>
        </property>
        <property name="jobNames">
            <list>
                <value>fiscalYearMakerJob</value>
                <value>populatePriorYearDataJob</value>
            </list>
        </property>
    </bean>
```

4. Don't forget to include your new Spring file by overriding the `spring.source.files` build property.

Unable to render {include} The included page could not be found.