Spring Beans 2

See the complete Spring framework documentation for more generation information. One reason that Spring was selected for the Kuali architecture was its support for delivering flexible and configurable applications.

1. Create your new Spring XML file within the project source directory (`work/src`).
   You may want to follow the package and file naming conventions used in the delivered code. If you are adding functionality to the core application or an existing module or you are overriding a delivered bean definition, start by locating the appropriate file. `SpringBeans.xml` in the root of the project defines the data source, configures transaction management and caching, and configures Rice. `org.kuali.kfs.KualiSpringBeansKfs.xml` configures other application-level beans. And, the Spring files that contain the beans for each module follow the pattern: `org.kuali.module.*.KualiSpringBeans*.xml`. Once you've located the file, name your file similarly. For example, if you are overriding a bean defined by the chart module, create a file named `SampleuSpringBeansChart.xml` in a `edu.sampleu.kuali.module.chart` package. If you are adding a module, create a file called `SampleuSpringBeans<your module name>.xml` in a `edu.sampleu.kuali.module.<your module name>` package.
2. Add the appropriate beans to your new file.
   When overriding a delivered bean, you *must* use the same **id**. Given that and the fact that you may well want to keep many properties the same, you will probably want to start by copying in the definition of the bean that you want to override. For example, if you want to overriding the `webAuthenticationService` bean defined in `SpringBeans.xml`...

   work/src/edu/sampleu/kuali/SampleuSpringBeans.xml

   ```
   <beans>
       <bean id="webAuthenticationService"
   class="edu.sampleu.kuali.SampleuWebAuthenticationService">
           <property name="validatePassword" value="${validate.password}" />
       </bean>
       ....
   </beans>
   ```

3. Create the new Java class, e.g. `edu.sampleu.kuali.SampleuWebAuthenticationService`.
   When overriding a delivered bean and changing the class, your class *must* implement the same interfaces that the class specified by the delivered bean implements.
4. Override the `spring.source.files` configuration property, and add the path to your new Spring file to the end of the list.
   See the build properties overview for details. You *must* put your Spring bean configuration files at the end of the list. The last bean definition with a given **id** will be used by the application.
   For example...

   work/src/edu/sampleu/kuali/sampleu-build.properties

   ```
   spring.source.files=org/kuali/kfs/KualiSpringBeansKfs.xml,\
   org/kuali/module/cg/KualiSpringBeansCg.xml,org/kuali/module/chart/KualiSpringBean
   sChart.xml,\
   org/kuali/module/financial/KualiSpringBeansFinancial.xml,org/kuali/module/gl/Kual
   iSpringBeansGl.xml,\
   org/kuali/module/kra/KualiSpringBeansKra.xml,org/kuali/module/labor/KualiSpringBe
   ansLabor.xml,\
   org/kuali/module/pdp/KualiSpringBeansPdp.xml,org/kuali/module/purap/KualiSpringBe
   ansPurap.xml,\
   org/kuali/module/vendor/KualiSpringBeansVendor.xml,SpringRiceBeans.xml,\
   edu/sampleu/kuali/SampleuSpringBeans.xml
   ```

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