**Simphony Application Framework (SAF)**

The Simhony Application Framework[[1]](#footnote-0) simplifies the development of Java Swing applications. SAF supports dockable panels, application perspectives and the declarative (via XML) definition of such things as menus and toolbars.

**Getting Started**

SAF is a plug-in based framework and uses the jpf library (jpf.sf.net) for its plug-in support. A plug-in is a structured component that describes itself using a manifest. A SAF application is essentially a plug-in and on application startup, SAF will discover and load this application plug-in.

A SAF application minimally consists of a plug-in definition file (plugin\_jpf.xml), an implementation of the IApplicationRunnable interface and optionally an implementation of the IAppConfigurator interface. The Demo application contains all of these. Examining each of hem in turn while reading this and then modifying them for your own application is the easiest way to get started.

1. The plug-in definition: plugin\_jpf.xml. The plugin-jpf is the application manifest where the application is defined. As such it does three things:
   1. Defines any external code (jars etc.) or resources (icon images etc.) that the application requires
   2. Defines the GUI of the application, such as perspectives, menus, dockable frame layout
   3. Defines the application entry point

The header of the plugin file begins with some standard boilerplate (see the plugin\_jpf.xml in the demo application for the details), followed by the declaration of the plugin proper.

|  |
| --- |
| <plugin id="saf.demo.plugin" version="1.0" class="saf.demo.DemoAppPlugin"> |

The id should some identifying string. The version can be whatever is appropriate for the application and the class if not empty should refer to an implementation of IApplicationRunnable.

The next section names any other plugins that this plugin depends on. All SAF applications will have at least saf.core.runtime and the saf.core.ui.

|  |
| --- |
| <requires>  <import plugin-id="saf.core.runtime" plugin-version="1.0"/>  <import plugin-id="saf.core.ui" plugin-version="1.0"/>  </requires> |

Next comes the <runtime> section which defines any additional code or resources that the application may need.

|  |
| --- |
| <runtime>  <library id="demo.bin" path="bin/" type="code">  <library id=”foo.lib” path=”lib/foo.jar” type=”code” />  </library>  <library type="resources" path="icons/" id="icons"/>  <library type="resources" path="help/" id="help"/>  </runtime> |

The library element is used to define both additional code and resources. The applications own code should be mentioned here as well as any additional jars or class directories that the application requires. Paths are relative to the location of the plugin\_jpf.xml. **Note** that any library entries will automatically be added to the runtime classpath.

The final item in the plugin\_jpf.xml, prior to the actual application GUI definition, is the specification of the IApplicationRunnable class.

|  |
| --- |
| <extension plugin-id="saf.core.runtime" point-id="IApplicationRunnable" id="demo.runnable">  <parameter id="class" value="saf.demo.DemoAppPlugin"/>  <parameter id="name" value="Demo Application"/>  <parameter id="description" value="SAF Demonstration Application"/>  </extension> |

The only part of this that will change from application to application is the values of the parameter elements. The class value should be the fully qualified name of a class that implements IApplicationRunnable and the name and description values should be appropriate to the application.

During application startup, SAF will read your plugin\_jpf.xml identify the libraries etc. and add them to the classpath. It will then instantiate your application’s IAppilicationRunnable as specified in the xml and call that instance’s run(String[] args) method.

1. The IApplicationRunnable interface.

|  |
| --- |
|  |

Ui.properties, boot.properties

1. SAF was originally developed as part of the Repast Simphony agent-based simulation project and thus the name. [↑](#footnote-ref-0)