sb11-program-management

The Program Management Component is responsible for configuration management and UI custom branding.

Usage

Rest Module

The REST module is a deployable WAR file (`prog-mgmnt.rest-VERSION.war`) that provides REST endpoints that can be used to access and modify Program Management data. The REST module has an internal dependency to the SB11 Program Management Persistence module.

In order to run the REST WAR application, all setup necessary for the Persistence module must be performed. No additional setup is required other than deploying the WAR to a Tomcat-compatible application server.

Webapp Module

The Webapp module is a deployable WAR file (`prog-mgmnt.webapp-VERSION.war`) that provides the administrative UI for Program Management functionality. The Webapp module uses the REST module for all data access, but this is a runtime dependency through a REST endpoint and not a direct code dependency.

The Webapp module requires a few things to be setup in order to run correctly:

- A properties file called rest-endpoints.properties should be created. This file contains two properties that are required. An example file is in this module. This file must be either on the classpath or within the ${SB11_CONFIG_DIR}/progman directory.
- Define the ${SB11_CONFIG_DIR} environment variable to be some directory on the file system. This could be set as a system variable or a startup variable to Tomcat or other application server.

Persistence Module

The Persistence module is a JAR artifact that is used by the REST module. This module is responsible for persistence of application data. It also encrypts and decrypts sensitive data.

In order for encryption of data to work, the unlimited JCE security policy must be installed. Copy encryption/UnlimitedJCEPolicy/local_policy.jar and encryption/UnlimitedJCEPolicy/US_export_policy.jar into your JDK's lib/security folder, replacing the existing files (please back up existing files). See the README.txt in that directory for more details.

Data encryption requires an externally defined property to hold a secret password. A file called pbe.properties should be placed into the ${SB11_CONFIG_DIR}/progman directory. The single property should have the key of pm.pbe.pass. The directory and file should have very restricted permissions as knowing the password will compromise any encrypted data in the database. The file will only be read from this configuration location. It cannot be placed on any other area of the classpath. If this password is lost, any encrypted data will be unable to be decrypted.

Two other properties files, mongo.properties and rest-endpoints.properties, must also be created. Example files are in the module.

Domain Module

The domain module contains all of the domain beans used to model the Program Management data as well as code used as search beans to create Mongo queries. It is a JAR artifact that is used by other modules.

Client Modules

There are three modules that make up the client:

- The Client Interfaces module contains the interface classes for the client, AOP advice and point cuts, bootstrap initialization
- The Null Client module is a client implementation that sends all messages to the local logging subsystem (SLF4J)
- The Integrated Client sends all messages to the advertised REST endpoints from the REST module

In order to use one of the client implementations:

- Add the Maven dependencies to the POM of the project that is using Program Management.

```
<dependency>
    <groupId>org.opentestsystem.shared</groupId>
    <artifactId>prog-mgmnt-client<artifactId>
    <version>${progman.client.version}<version>
</dependency>

<dependency>
    <groupId>org.opentestsystem.shared</groupId>
    <artifactId>prog-mgmnt-null-client</artifactId>
    <version>${progman.client.version}</version>
</dependency>
```

- Add a context initializer class name to the web.xml

```
<context-param>
    <param-name>contextInitializerClasses</param-name>
    <param-
value>org.smarterbalanced.progman.init.InitSpringPropertyConfigLoad</param-
value>
</context-param>
```

- Add a file called /src/main/resources/spring/progman-loader-config-props-context.xml with contents

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

    <bean id="tibProgmanPropertyConfigurer"
class="org.springframework.context.support.PropertySourcesPlaceholderConfigurer"
>
        <property name="ignoreResourceNotFound" value="true"/>
        <property name="ignoreUnresolvablePlaceholders" value="true"/>
    </bean>

</beans>
```

- Add environment variables to application server startup
    - `-Dspring.profiles.active="progman.client.impl.integration"`
        - Use `progman.client.impl.integration` to use the fully integrated client
        - Use `progman.client.impl.null` to use the null client
    - `-Dprogman.baseUri=http://localhost:8089/prog-mgmnt.rest/`
        - This URI is the base URI for where the Program Management REST module is deployed
    - `-Dprogman.locator=tib,dev[,overlay];`
        - The locator variable describes which combinations of name and environment (with optional overlay) should be loaded from Program Management. For example: `"component1-urls,dev"` would look up the name component1-urls for the dev environment at the configured REST endpoint. Multiple lookups can be performed by using a semi-colon to

delimit the pairs (or triplets with overlay): `"component1-urls,dev;component1-other,dev"`

- Before attempting to start the application that integrates with Program Management, be sure that the name/environment configuration parameters are configured in the Program Management UI

Build

These are the steps that should be taken in order to build all of the Program Management related artifacts.

Pre-Dependencies

- Mongo 2.0 or higher
- Tomcat 6 or higher
- Maven (mvn) version 3.X or higher installed
- Java 7
- Access to sb11-shared-build repository
- Access to sb11-shared-code repository
- Access to sb11-rest-api-generator repository
- Access to sb11-program-management repository

Build order

- sb11-shared-build
- sb11-shared-code
- sb11-rest-api-generator
- sb11-program-management

Dependencies

Program Management has a number of direct dependencies that are necessary for it to function. These dependencies are already built into the POM files.

Compile Time Dependencies

- Apache Commons IO
- Apache Commons Beanutils
- Jackson Datatype Joda
- Google Guava
- Hibernate Validator
- Apache Commons File Upload

- Jasypt
- SB11 Shared Code
    - Logback
    - SLF4J
    - JCL over SLF4J
    - Spring Core
    - Spring Beans
    - Spring Data MongoDb
    - Mongo Data Driver
    - Spring Context
    - Spring WebMVC
    - Spring Web
    - Spring Aspects
    - AspectJ RT
    - AspectJ Weaver
    - Javax Inject
    - Apache HttpClient
    - JSTL API
    - Apache Commons Lang
    - Joda Time
    - Jackson Core
    - Jackson Annotations
    - Jackson Databind
- SB11 REST API Generator
    - JSTL
    - Apache Commons Lang

Test Dependencies

- Spring Test
- Hamcrest
- JUnit 4
- Flapdoodle
- Podam
- Log4J over SLF4J

Runtime Dependencies

- Servlet API
- Persistence API