Paraname8 MetaModel Facets

Table of Contents

This extension (`isis-metamodel-paraname8`) for Apache Isis' metamodel support means that the name of action parameters can be inferred from the parameter name itself; that is, there is no need to annotate the parameter.

Background

Prior to Java 8 it was not possible to obtain the parameter names of methods using reflection. Since Apache Isis builds the UI from the code features, this required the parameters to be annotated with the `@ParameterLayout(named=…)` annotation or the (deprecated) `@Named(…)` annotation.

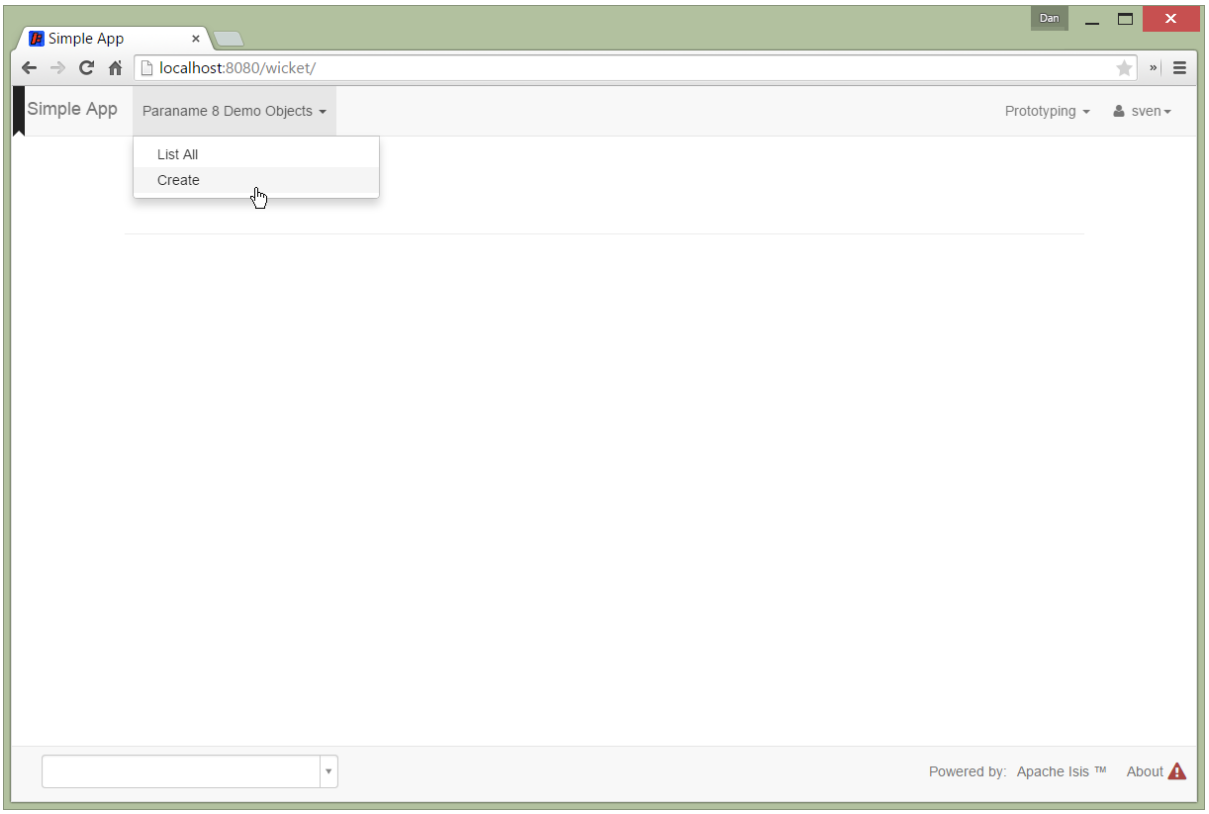
In Java 8 the Reflections API has been extended so that the parameter name is available (with the proviso that the code must also be compiled with a new `-parameters` compile flag).

This module provides an implemenation of Apache Isis' `FacetFactory` for Apache Isis so that this parameter name can be used, thereby avoiding the need to annotate action parameters.

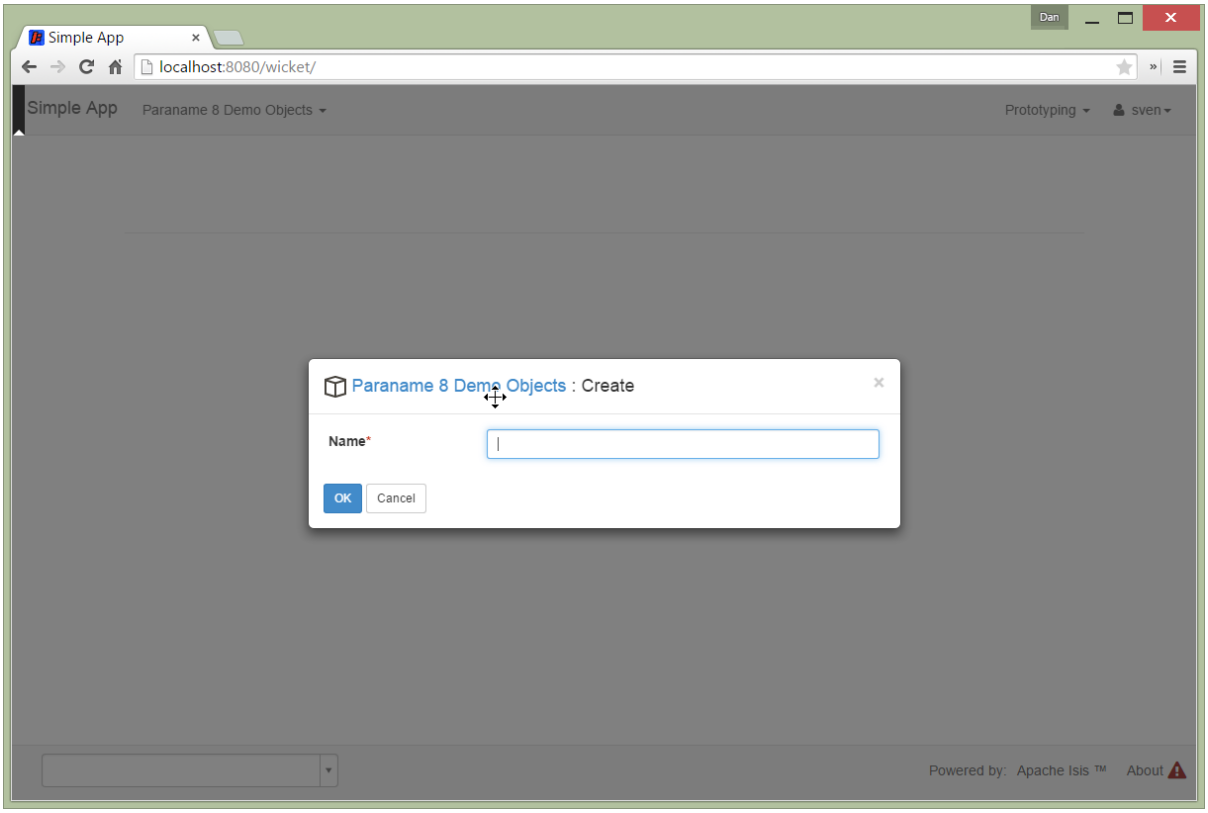
Screenshot and Corresponding Code

The screenshots below **do** demonstrate the functionality of this module, but are out of date in that they are taken from the original isisaddons/incodehq module (prior to being amalgamated into the incode-platform).

From the demo app, here's the screenshot of an action to create a new object:



which renders the following prompt:



The corresponding code is simply:

```
public Paraname8DemoObject create(final String name) {
    ...
}
```

Compare this to the "normal" way, which required using either the `@ParameterLayout(named=…)` annotation:

```
public Paraname8DemoObject create(
        @ParameterLayout(named="Name")
        final String name) {
    ...
}
```

How to Configure/Use

Compiler plugin

In your project's root `pom.xml`, update the `maven-compiler-plugin` definition (in `<build>/<pluginManagement>/<plugins>`) to compile with JDK8, and specify the `-parameters` argument:

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-compiler-plugin</artifactId>
    <version>3.1</version>
    <configuration>
        <source>1.8</source>
        <target>1.8</target>
        <compilerArgument>-parameters</compilerArgument>
    </configuration>
    <executions>
        <execution>
            <id>source</id>
            <phase>compile</phase>
        </execution>
        <execution>
            <id>test</id>
            <phase>test-compile</phase>
        </execution>
    </executions>
</plugin>
```

If using mavenmixins (as the parent pom configures), then note that the standard mavenmixin already configures the `maven-compiler-plugin` for this, so no further configuration is necessary.

Classpath

Update your classpath in the `pom.xml` of your project's `integtests` module and also its `webapp` module:

```
<dependency>
    <groupId>org.isisaddons.metamodel.paraname8</groupId>
    <artifactId>isis-metamodel-paraname8-dom</artifactId>
</dependency>
```

Check for later releases by searching Maven Central Repo.

Configuration Properties

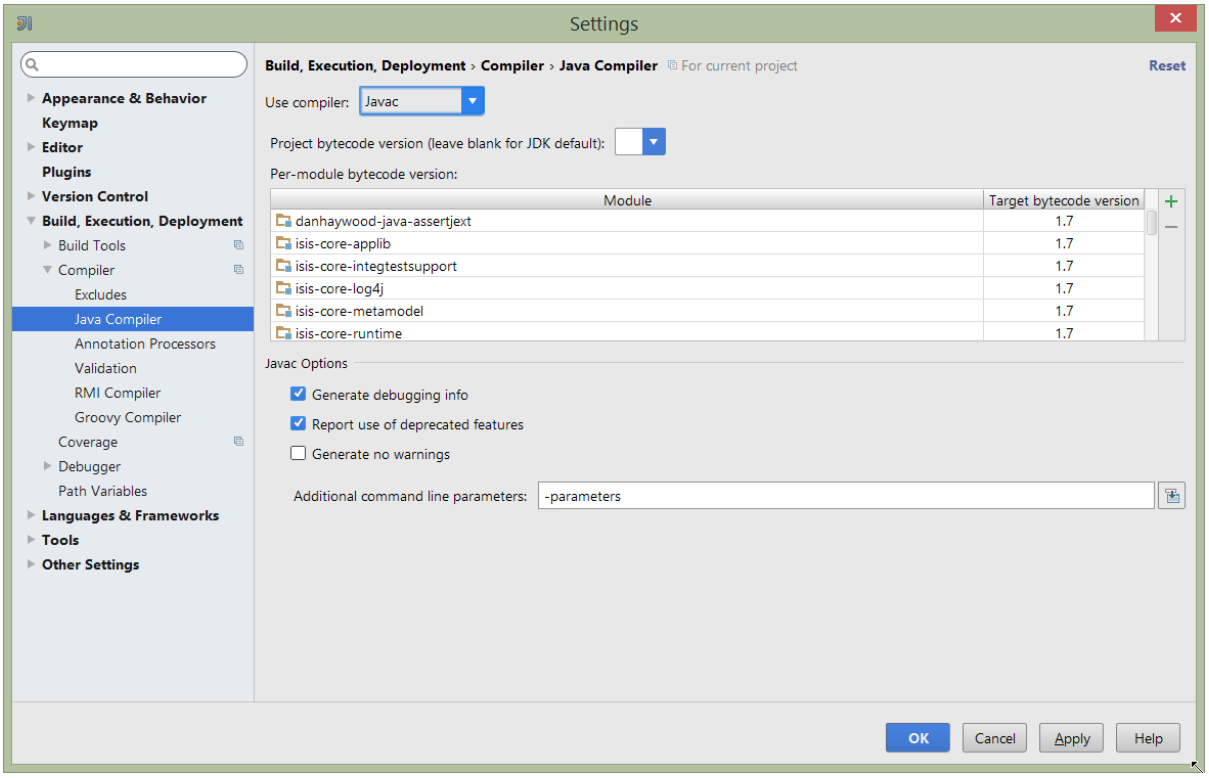
Update the `isis.reflector.facets.include` configuration property, eg in the `AppManifest` or in `isis.properties`:

```
isis.reflector.facets.include=\
            org.isisaddons.metamodel.paraname8.NamedFacetOnParameterParaname8Factory
```

Configuring your IDE

Most IDEs compile the Java source code independently of Maven; this is certainly the case with both IntelliJ IDEA and Eclipse. You will therefore need to ensure that the IDE is set up to build using the `-parameters` flag.

For IntelliJ IDEA, for example, this can be found under the "Settings" dialog:



Other IDEs should have similar dialogs.

You'll also need to make sure that the IDE is set up to build and run with JDK8. In IntelliJ, this can be found under the "Project Structure" dialog.

Known issues

None known at this time.

Dependencies

Maven can report modules dependencies using:

```
mvn dependency:list -o -pl modules/mml/paraname8/impl -D excludeTransitive=true
```

which, excluding Apache Isis itself, returns no direct compile/runtime dependencies.