

Architecture and Design

Table of Contents

1. Maven modules.....	2
1.1. core.....	2
1.2. core/mavendeps	4
1.3. core/mavenplugins.....	6
1.4. archetypes	6
2. Metamodel	8

This guide describes the internal architecture and design of the framework.

Chapter 1. Maven modules

```
adocs/
├── documentation/
└── template/
core/           # see 'core', below
example/       # see 'archetypes', below
scripts/
```

1.1. core

The core modules

All of these have the same Maven `groupId`, namely `org.apache.isis.core`.

```
core/
├── .m2/           # used in gitlab CI
├── applib/        # isis-core-applib
├── commons/       # isis-core-commons
├── integtestsupport/ # isis-core-integtestsupport
├── log4j/         # isis-core-log4j
├── maven-plugin/  # see 'maven plugins', below
├── mavedeps/      # see 'mavedeps', below
├── runtime/       # isis-core-runtime
├── schema/        # isis-core-schema
├── security/      # isis-core-security
├── security-shiro/ # isis-core-security-shiro
├── specsupport/   # isis-core-specsupport
├── unittestsupport/ # isis-core-unittestsupport
├── unittestsupport-test/ # isis-core-unittestsupport-test
├── viewer-restfulobjects-applib/ # isis-core-viewer-restfulobjects-applib
├── viewer-restfulobjects-rendering/ # isis-core-viewer-restfulobjects-rendering
├── viewer-restfulobjects-server/ # isis-core-viewer-restfulobjects-server
├── viewer-wicket-applib/ # isis-core-viewer-wicket-applib
├── viewer-wicket-impl/ # isis-core-viewer-wicket-impl
├── viewer-wicket-model/ # isis-core-viewer-wicket-model
├── viewer-wicket-ui/ # isis-core-viewer-wicket-ui
├── webdocker/     # isis-webdocker
├── webserver/     # isis-core-webserver
└── wrapper/      # isis-core-wrapper
```

Table 1. core maven modules

Module	Description
<code>isis-core-applib</code>	Core application library.

Module	Description
<code>isis-core-integtestsupport</code>	Integration test support. Application integration tests typically extend from adapter superclasses defined in this module.
<code>isis-core-log4j</code>	Configures Log4j as the logging framework
<code>isis-core-metamodel</code>	<p>The classes that make up the metamodel which is used to render the UI.</p> <p>See the section below which also includes a simplified UML diagram of these classes.</p>
<code>isis-core-runtime</code>	The classes that make up runtime management and persistence of domain objects, as well as framework for security (concepts of authentication or authorisation).
<code>isis-core-schema</code>	Defines XSDs and generated classes that capture commands and interactions in XML form.
<code>isis-core-security</code>	<p>Defines a "bypass" implementation of security, for prototyping only.</p> <p>Using this implementation, any user/password is accepted and</p>
<code>isis-core-security-shiro</code>	Defines an implementation of security authentication which delegates to Apache Shiro.
<code>isis-core-specsupport</code>	Application BDD specs typically inherit from classes defined in this module.
<code>isis-core-unittestsupport</code>	Application unit tests may use some of the utilities defined in this module.
<code>isis-core-viewer-restfulobjects-applib</code>	Defines a client-side Java library for interacting with the REST API exposed by the Restful Objects viewer.
<code>isis-core-viewer-restfulobjects-rendering</code>	<p>Provides a <code>RepresentationService</code> API and a lower-level <code>ContentNegotiationService</code> API, along with implementations of each.</p> <p>These implementations provide support for the representations defined by Restful Objects spec v1.0, as well as a number of other Apache Isis-specific representations.</p>

Module	Description
<code>isis-core-viewer-restfulobjects-server</code>	<p>Defines the JAX-RS resources supported by the Restful Objects viewer.</p> <p>These parse the input, delegate to the runtime for a response, and hand control to the rendering module to generate a representation.</p>
<code>isis-core-viewer-wicket-applib</code>	Currently just defines <code>WicketDeveloperUtilitiesService</code> , for clearing the cache.
<code>isis-core-viewer-wicket-impl</code>	<p>The top-level integration with Wicket, for example defining the Apache Isis-specific implementations/subclasses of the Wicket APIs for application, web session, localizer and request cycle.</p> <p>Also defines registries of pages and components, as well as a number of domain services and mixins (for use by applications) that are only available within the Wicket viewer.</p>
<code>isis-core-viewer-wicket-model</code>	Serializable mementos representing the state of runtime domain objects (or their individual members).
<code>isis-core-viewer-wicket-ui</code>	UI components that render the moduls.
<code>isis-webdocker</code>	Creates a Docker image for Tomcat that also contains the Apache Isis libraries, thereby enabling "skinny war" support.
<code>isis-core-webserver</code>	For development within an IDE, provides a utility class to bootstrap the application (using Jetty).
<code>isis-core-wrapper</code>	Provides an implementation of the <code>WrapperFactory</code> domain service.

1.2. core/mavendeps

The `core/mavendeps` modules ...

All of these have the same Maven `groupId`, namely `org.apache.isis.mavendeps`.

```

core
├── mavendeps/
│   ├── isis-mavendeps-intellij/    # isis-mavendeps-intellij
│   ├── isis-mavendeps-testing/    # isis-mavendeps-testing
│   └── isis-mavendeps-webapp/      # isis-mavendeps-webapp

```

Table 2. core/mavendeps maven modules

Module	Description
isis-mavendeps-intellij	Defunct.
isis-mavendeps-testing	<p>Aggregates dependencies on various test-scope plugins useful for unit- and integration testing a module. These include Apache Isis' own <code>unittestsupport</code>, <code>integtestsupport</code> and <code>specsupport</code> modules, as well as a number of common testing/mocking/assertion libraries.</p> <p>These can then be included using a single dependency declaration:</p> <pre> <dependencies> <dependency> <groupId>org.apache.isis.mavendeps</groupId> <artifactId>isis-mavendeps-testing</artifactId> <type>pom</type> <scope>test</scope> </dependency> </dependencies> </pre>
isis-mavendeps-webapp	<p>Aggregates dependencies on Apache Isis runtime itself when used within a webapp.</p> <p>These can then be included using a single dependency declaration:</p> <pre> <dependencies> <dependency> <groupId>org.apache.isis.mavendeps</groupId> <artifactId>isis-mavendeps-webapp</artifactId> <type>pom</type> </dependency> </dependencies> </pre>

1.3. core/mavenplugins

There is a single Maven plugin module. Its Maven `groupId` is `org.apache.isis.tools`.

Plugin Modules

```
core/
└── maven-plugin/          # isis-maven-plugin
```

Module	Description
<code>isis-maven-plugin</code>	Code to build a maven plugin for the build. This plugin can validate the metamodel and generate Swagger specs for a domain model as part of the application's build pipeline.

1.4. archetypes

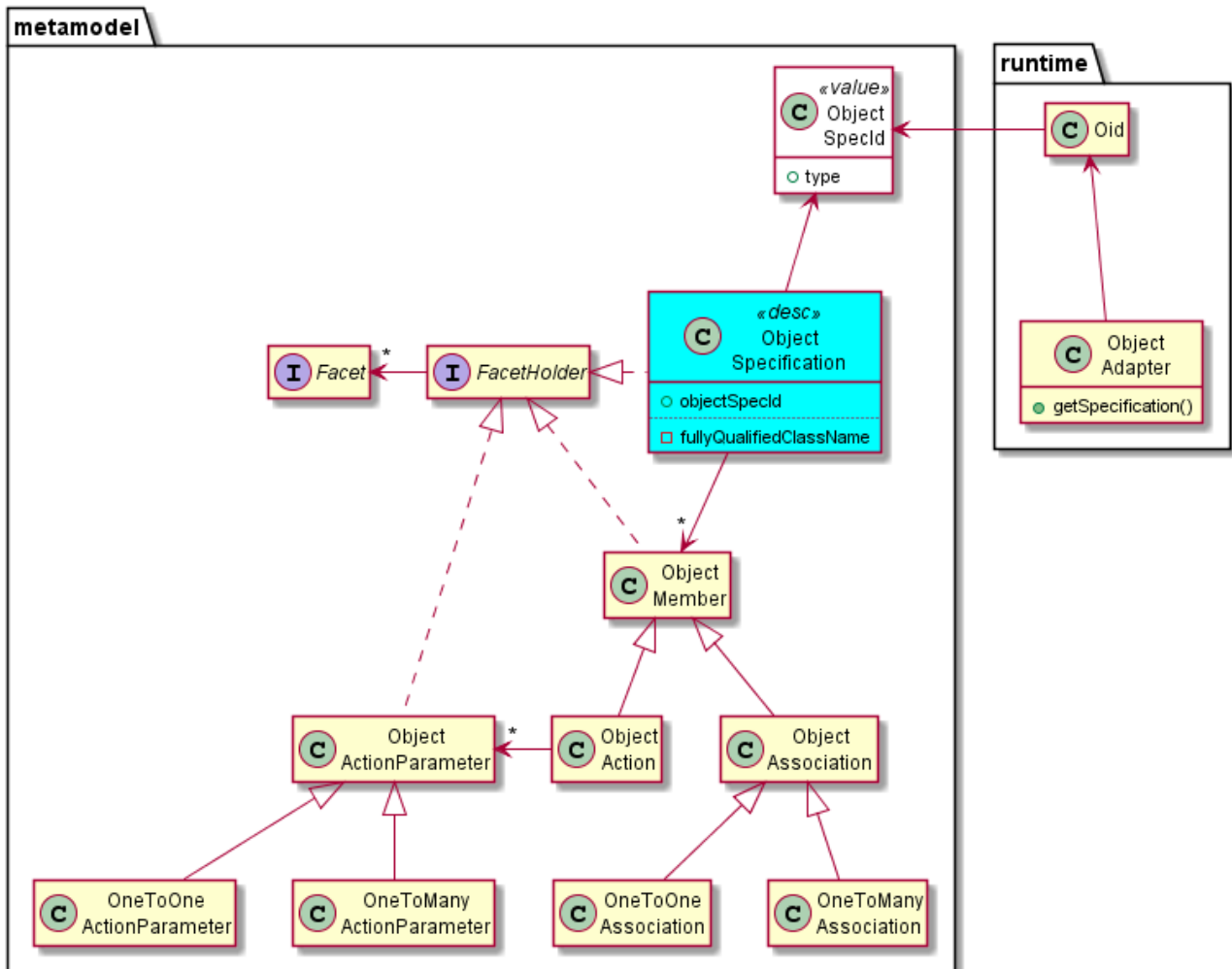
```
example/
├── application/
│   ├── helloworld/      # org.apache.isis.example.application:helloworld
│   └── simpleapp/       # org.apache.isis.example.application:simpleapp
│       └── application/  # org.apache.isis.example.application:simpleapp-
application
│   └── module-simple/   # org.apache.isis.example.application:simpleapp-module-
simple
│       └── webapp/      # org.apache.isis.example.application:simpleapp-webapp
└── archetype/
    ├── helloworld/     # org.apache.isis.archetype:helloworld-archetype
    └── simpleapp/       # org.apache.isis.archetype:simpleapp-archetype
```

Module	Description
<code>helloworld</code>	<p>An example application as a single Maven module, including domain classes themselves plus code to bootstrap Apache Isis.</p> <p>This is reverse engineered into the "helloworld" archetype.</p>
<code>simpleapp</code>	<p>The top-level aggregator module for the "simpleapp" example application.</p> <p>This is an extended version of helloworld, providing more structure (separating out domain model into modules) as well as unit tests, integration tests, BDD specs and fixtures.</p> <p>The simpleapp modules in aggregate are reverse engineered into the "simpleapp" archetype.</p>

Module	Description
simpleapp-application	Defines the contents of the "simpleapp" application using Apache Isis-defined classes, as well as globally scoped domain services and the home page.
simpleapp-module-simple	<p>Contains the domain model for a single module.</p> <p>The intention is to allow this module structure to be copied so that the developer can easily create further modules as their app increases in size.</p>
simpleapp-webapp	Bootstraps Apache Isis as a webapp.
helloworld-archetype	Helloworld archetype, reverse engineered from the "helloworld" application (above).
simpleapp-archetype	Simpleapp archetype, reverse engineered from the "simpleapp" application (above).

Chapter 2. Metamodel

The diagram below shows a simplified version of Apache Isis' internal metamodel.



where in the `metamodel` package:

ObjectSpecification

is equivalent to `java.lang.Class`

ObjectSpecId

is a value object equivalent to the `@DomainObject#objectType` or `@DomainService#objectType` attribute

OneToOneAssociation

represents a scalar property

OneToManyAssociation

represents a collection

ObjectAction

represents an action (with multiple parameters, either scalar or list)

and in the `runtime` package:

`Oid`

is equivalent to the applib `Bookmark`

and appears in URLs in the Wicket and Restful Objects viewers

`ObjectAdapter`

is equivalent to `java.lang.Object`