Chemistry (Outdated)

This document is outdated!
The former Chemistry library and OpenCMIS were merged into one Java library (OpenCMIS).
Chemistry is the name of the Apache CMIS implementation project, while OpenCMIS is the name of Chemistry's Java implementation.

Chemistry's goal is to provide general purposes libraries for interaction using CMIS between a server and a client.

Chemistry provides a high level object-oriented API so that an application developer can manipulate objects like documents or folders and can call simple methods on them without having to deal with details of a specific low-level communication transport. In addition to that, Chemistry also provides a SPI (Service Provider Interface) for backend developers, making it quite easy to use Chemistry to store documents in a project-specific manner. The SPI can also be used by the application developer if he wishes to get to the "bare metal" of the CMIS protocol.

Underlying this, Chemistry has implementations for the two CMIS transports, AtomPub and SOAP.

The following modules are available in Chemistry. The title is a link to a more detailed description of each module.

Chemistry API

There are two API levels:

- a low-level SPI between a client and a server that mirrors the CMIS specification closely (it is expected that the SPI will be used when either the client or the server implements one of the HTTP protocols defined in CMIS),
- a high-level API that wraps the SPI to provide more object-oriented notions of connections, folders and documents, and that hides the nitty-gritty details of the protocols.

Both levels of APIs are based on a common set of low-level constants and enums tied to the CMIS domain model (property types, capabilities, etc.).

Chemistry Commons

Several reusable pieces of codes are hosted under this module:

- "simple", an in-memory implementation of the API/SPI that is useful for unit testing,
- an ANTLR3 parser for CMISQL that turns a CMISQL query string into an AST (Abstract Syntax Tree) that can be reused by different backends, and is used by the in-memory implementation.

Chemistry AtomPub

This module holds some base constants and classes useful both for the AtomPub client and server.

Chemistry AtomPub Server

This module is the AtomPub implementation of the API/SPI. It receives AtomPub requests, converts them into SPI calls, then marshals then answer back into AtomPub responses.

This server can be bound either as a servlet, or as a JAX-RS resource.

Chemistry AtomPub Client

This module is the client implementation of the API/SPI. It turns Java calls into AtomPub requests to a remote server, reads the answer, and turns it back into Java objects.

Chemistry WS

These are the Java JAX-WS classes that are automatically generated from the WSDL and used by the SOAP server and client.

Chemistry SOAP Server

This module is the SOAP implementation of the API/SPI. It receives SOAP requests, converts them into SPI calls, then marshals then answer back into SOAP responses.

Chemistry SOAP Client

This module is not yet written. It would turn Java calls into SOAP requests to a remote server, read the answer, and turn it back into Java objects.
Chemistry JCR
This module is an implementation of the API/SPI for the JCR. It turns Java calls to the Chemistry API/SPI into calls to a JCR store, using Jackrabbit in the current implementation.

Chemistry Shell
This module is a command-line shell that can be used to quickly and simply make requests to a remove CMIS AtomPub server. It can browse documents, make queries, etc.

Chemistry Tests
This module contains unit tests that stress both the client and server parts of the API/SPI, either by direct calls, or using round-trips through HTTP for the AtomPub implementation.

There is also a simple AtomPub servlet bound to the in-memory implementation that can be started so that third-party tools can quickly test against a running CMIS AtomPub server.

Chemistry Abdera
This module is a set of Abdera classes that are designed for CMIS.

Chemistry TCK AtomPub
This module is a Technology Compatibility Kit that can be run against a CMIS AtomPub server in order to test its compliance with the standard. Its code base is independent of the other Chemistry modules above except for Chemistry Abdera.