OlingoProposal

Apache Olingo

Abstract

Apache Olingo is a generic Java language implementation of the OData 2.0 specification which will serve as a code base for the upcoming OASIS OData specification.

Proposal

The Open Data Protocol (OData) [1] is a Web protocol for querying and updating data that provides a way to unlock your data and free it from silos that exist in applications today. OData does this by applying and building upon Web technologies such as HTTP, Atom Publishing Protocol (AtomPub) and JSON to provide access to information from a variety of applications, services, and stores.

The Apache Olingo is a library which enables developers to implement OData producers and OData consumers. Basic principles of the library are to provide an OData 2.0 specification compliant OData Library, enhancements shall be possible in a compatible manner, have a clear separation between Core and API, to provide an option to build extensions on top. This library should be base for implementing future releases of the specification.

Background

OData was originally developed by Microsoft and is released in a version 2.0 under an Open Specification Promise [2]. A lot of companies did show interests in this protocol, used it in products and gave feedback back to Microsoft. This joined effort resulted in a new release OData 3.0 in 2012, this version became the basis for the OASIS technical committee [3] which is currently working on a new version of the specification. This OASIS standard release is expected this year.

The initial Java code of this project was developed by a development team that had already experience with other OData 2.0 and 3.0 implementations at SAP AG. The current code base implements OData 2.0 and because of this version is widely used it is a good starting point to build an open source community for the OData standard.

The current code also comes up with an implementation of an OData sample service. On the one side this is an example for users which want to use the library to expose their own data and on the other side it illustrates how implemented features work.

Additionally, the code base includes an extension which is called JPA processor. With this extension it is easy to expose any JPA persistence model via OData protocol without a lot of coding.

Rationale

More software vendors moving to OData means more choice for customers who will be able to use different implementations. For the standard to succeed, however, ensuring interoperability is paramount: in order to manage an ever growing context and leverage the enormous portability and interoperability issues that a globally adopted standard brings, it is necessary to think about how to make the related ecosystem healthy and sustainable. Successful modern standards are driven by:

- Clear documentation, built iteratively with continuous feedback from stakeholders
- A clearly defined compatibility process, enforced by tools that allow to gauge how implementations can be compatible and interoperable
- Accurate compliance criteria, documented in writing as well as in actual testing code that measure how tools and libraries are able to interoperate
- A sample implementation to clear up potential doubts and ensure that the standard can actually be implemented in real life scenarios

The above mentioned pieces are able to make the development activity, towards an OData implementation, easier and more successful. Having an healthy ecosystem will ensure a smoother implementation process, more compliant products, and ultimately, a wider adoption of the standard.

The OData ecosystem has been successful in creating and documenting early versions of the standard, yet it might potentially lack two very important aspects, that is a exhaustive implementation of the complete protocol that can be used productively and to ensure interoperability. As much as such artifacts can be developed independently by any OData proponent, the value of having a neutral party as a steward of actual code is to be considered. The Apache Software Foundation has been playing this kind of role for many years, and can provide the perfect environment to foster contributions on the OData theme with a great amount of expertise.

Initial Goals

- Implement OData 2.0, make it final and mature
- Start implementation of OASIS OData draft specification
- Provide input and feedback for the draft specification to the OASIS OData TC based on implementation
- Implement OData add-ons (library extensions and toolset)

Current Status

Meritocracy

Most of the initial committers are open source developers with different experience levels and many of them have already worked in other open source or Apache projects. We will follow standard Apache procedures in accepting new contributors to the project team.

Community

Managed by an OASIS Technical Committee, the OData standard definition should be based on the idea of a community driven effort.

Apache Olingo aims to be a community driven initiative in developing a Java OData implementation. Such an approach is allowing more transparency and direct feedback even within the definition and improvement of OData specifications.

We encourage everyone interested in OData to join the Apache Olingo effort.

Core Developers

The development team is international and they have all strong skills in OData protocol. Jens Huesken who is member of the OASIS OData TC is providing specification feedback since OData 2.0. Stephan Klevenz, also a OASIS OData TC member, was a committer of the Apache Chemistry project. He has experience with the Incubator and Apache and was also a speaker on ApacheCon 2012 in Vancouver. Christian Amend, Michael Bolz and Tamara Boehm did implement core parts of the library. Chandan V A, Anirban Roy, Chitresh Chauhan, Jobin John and Joerg Singler are working on the JPA processor add on.

Alignment

The project builds with Apache Maven, the core runtime requires Apache CXF for REST binding (JAX-RS) and the sample scenario can be deployed into any compliant Servlet or J2EE container like Apache Tomcat. Furthermore we see OData protocol as an option to be supported by other Apache projects that have to expose data via a standardized protocol based interface.

Known Risks

Orphaned products

Apache Olingo is a fresh new codebase that targets the still moving OData standardization effort. Thus the future of this project depends heavily on the success of the standard. We hope and expect that our implementation efforts will strengthen and support the OData standard.

Inexperience with Open Source

Some of the initial committers are experienced open source developers. But there are also committers which are new to open source.

Homogenous Developers

The initial committers are from SAP working in different teams. One team is from Germany and has implemented the core parts of the library and the other team is from India and has implemented the JPA processor add-on.

Reliance of Salaried Developers

All of the initial committers are paid by SAP to work on this or related projects.

Relationships with Other Apache Products

Apache Olingo will directly use at least the following projects:

- Apache CXF - for REST bindings
- Apache Commons - for encoding/decoding

Other Apache projects may be interested in using Apache Olingo to add OData support once the standard is final. List of potential integrators:

- Apache Bloodhound

An Excessive Fascination with the Apache Brand

We value Apache as a neutral place where diverse communities can work together on implementing shared standards. We hope that this part of the Apache brand helps attract contributions from many potential OData standard consumers. However, the brand value is not the main reason why we prefer to have this project at Apache.

Documentation

This project is still at an early stage, so there is not much documentation available. See the OASIS OData page and odata.org web site for information about the OData standardization effort.

Initial Source

- https://github.com/SAP/cloud-odata-java
- https://www.ohloh.net/p/cloud-odata-java

Source and Intellectual Property Submission Plan

The complete code is under Apache Software License 2.

External Dependencies

All the external dependencies of the initial codebases comply with Apache licensing policies.

Cryptography

Apache Olingo is not expected to implement or use cryptographic code.

Required Resources

Mailing lists

- olingo-dev'at'incubator.apache.org
- olingo-commits'at'incubator.apache.org
- olingo-private'at'incubator.apache.org

Subversion Directory

writeable Git Repository (preferred)

git://git.apache.org/olingo.git

or SVN Directory

https://svn.apache.org/repos/asf/incubator/olingo

Issue Tracking

JIRA Olingo

Other Resources

none

Initial Committers

Name	Email
Stephan Klevenz	sklevenz'at'apache.org
Jens Huesken	jens.huesken'at'sap.com
Christian Amend	christian.amend'at'sap.com
Michael Bolz	michael.bolz'at'sap.com
Tamara Boehm	tamara.boehm'at'sap.com
Chandan V A	chandan.v.a'at'sap.com
Anirban Roy	anirban.roy'at'sap.com
Chitresh Chauhan	chitresh.chauhan'at'sap.com
Jobin John	jobin.john'at'sap.com
Joerg Singler	joerg.singler'at'sap.com
Francesco Chicchiriccò	ilgrosso'at'apache.org

Affiliations

Name	Affilitation
Stephan Klevenz	SAP AG

Jens Huesken	SAP AG
Christian Amend	SAP AG
Michael Bolz	SAP AG
Tamara Boehm	SAP AG
Chandan V A	SAP AG
Anirban Roy	SAP AG
Chitresh Chauhan	SAP AG
Jobin John	SAP AG
Joerg Singler	SAP AG
Francesco Chicchiriccò	Tirasa

Sponsors

Champion

- Florian Mueller

Nominated Mentors

- Florian Mueller
- Dave Fisher (wave at apache.org)
- Alan Cabrera

Sponsoring Entiy

- Incubator PMC

Links

- [1] http://www.odata.org
- [2] http://www.microsoft.com/openspecifications/en/us/programs/osp/default.aspx
- [3] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=odata
- [4] https://www.oasis-open.org/committees/membership.php?wg_abbrev=odata