JuneauProposal

Juneau Toolkit

⚠️ **FINAL** ⚠️

This proposal is now complete and has been submitted for a VOTE.

Abstract

Apache Juneau is a toolkit for marshalling POJOs to a wide variety of content types using a common framework, and for creating sophisticated self-documenting REST interfaces and microservices using VERY little code.

Proposal

An external website has been set up to point to existing libraries and documentation: https://sites.google.com/site/apachejuneau/

Juneau consists of 4 components: **Core**, **Server**, **Client**, and **Microservice**.

- **Core**
  - Simple, flexible, all-in-one API for serializing/parsing Java POJOs to JSON, XML, HTML, URL-Encoding, UON, RDF/XML, RDF/XML-ABBREV, N3, Turtle, N-Tuple, XML-Schema, JSON-schema, and Cognos-XML.
  - Serialize/parse structured documents using beans, or unstructured free-form documents using generic Maps and Collections.
  - DTO objects for creating ATOM feeds, Cognos, and JSON-schema in any supported languages (e.g ATOM in JSON or RDF).
  - Serializers have many options for tweaking output (e.g. whitespace options, XML namespace options, strict/non-strict syntax, etc...)
  - Sophisticated JSON parsing support. Supports ALL valid JSON. It also supports Javascript-style comments, single or double quoting, quoted (strict) or unquoted (non-strict) attributes, JSON fragments (e.g JSON string, numeric, boolean values), and concatenated strings.
  - Sophisticated API for reading and manipulating configuration INI files using POJOs.
- **Server**
  - Build complex REST interfaces using annotations and servlets.
  - Automatic parsing of POJOs from HTTP bodies, form posts, query parameters, URL variables, request attributes, headers, etc...
  - Automatic serializing of POJOs to HTTP responses.
  - Automatic error handling.
  - Automatic self-documenting OPTIONs pages.
  - Support for guards, filters, etc...
  - Simple internationalization support.
  - JAX-RS integration support.
- **Client**
  - Simple REST Client API for interacting with REST services using POJOs. Built on top of Apache HTTP Client.
  - Ability to retrieve POJO proxies that use REST as the communcations protocol.
- **Microservice**
  - Build self-contained REST interfaces as executable jars.
  - Built on top of embedded Jetty and configured through either external INI config files or manifest files.

Background

Juneau has been in use for years on over 50 projects throughout IBM. Recently it was hosted as a component of Jazz Foundation where it was used by several projects that make up Rational Collaborative Lifecycle Management. Many projects use only the marshalling framework, while others use the entire API to create sophisticated REST server/client interfaces.

It started life as an IBM Community Source (internal open source) project called "JJSON", a POJO JSON marshalling library. It was later renamed "Juno" due to existing use of the name.

Between 2008-2012, the project was used by many IBM projects and was at times the 2nd-most popular community source project in IBM.

In 2012, the project was merged into Jazz Foundation, but it also remained as a separate IBM Community Source project. The "main" stream was in Jazz Foundation, and the streams would periodically be synced to produce standalone releases for non-Jazz products.

In April 2016, the lead developer left IBM to join Salesforce. IBM agreed to make it an open source project so that development on it could continue.

Peter Haumer is the IBM liaison in this endeavor. He will be handling the legal aspects of this contribution. He uses Juneau in multiple projects and has an interest in seeing development continue. He has plans on enhancing the RDF support.

Rationale

Currently, there is no common framework for marshalling POJOs in a consistent fashion to a variety of content types. Having a single common API for serializers and parsers allows for a REST API that's considerably simpler than similar frameworks such as JAX-RS.

The use of a common infrastructure for marshalling has many advantages. Enhancements can often be made in the core framework without having to modify the serializers and parsers. And an exhaustive set of JUnit testcases ensure that POJOs are represented in a consistent way when represented in any of those content types.

Combining the marshalling framework with an annotation-based REST API produces a framework that allows developers to create REST interfaces using POJOs without having to deal with complexities of proper handling of Accept and Content-Type headers on HTTP requests and responses. The result is a REST interface that can automatically parse requests into POJOs and serialize POJOs in responses in any of the supported languages.

Initial Goals

The goals of Juneau include:

- Simplicity! The ability to create complex REST interfaces in as little code as possible. Let the developers deal with POJOs and let the framework handle the details of marshalling those POJOs.
- Discoverability! Users should be able to navigate through and figure out how to use a REST interface through the use of auto-generated OPTIONs pages.
- Customizability! Lots of easy-to-understand configurable options throughout the framework.
- Performance! Lots of work has been done to optimize the marshalling framework for use in high-concurrency environments. POJOs are marshalled directly without the use of intermediate DOM objects.
- No code generation or code scanning. Code generators are great until they break and nobody knows how to fix them.
- No configuration or setup for using. Simply include the library in your existing code and start using. Packaged as either an all-in-one jar or as OSGi bundles.
- Mimimal dependence on any 3rd party libraries. See prereqs below.
- Always up-to-date information by using Javadocs for documentation. The Javadoc overview document provides an overview of the toolkit with links to detailed package-level topics.

Current Status

Juneau currently exists under the code name "Juno" within Jazz Foundation and is actively used in many business units throughout IBM and throughout the world. This project is an attempt to convert that code base into an open source library. "Juneau" is a fork from this existing codebase that will be maintained and hosted in a separate GitHub repository.

The codebase is mature and tested. The code and documentation is up-to-date. It includes 3000+ JUnit testcases that ensure that POJOs are represented consistently across all supported content types.

Meritocracy

Juneau was originally created by James Bognar, team lead in Jazz Foundation, in 2007 as part of Rational Asset Analyzer as a tool for converting POJOs to JSON. Later, it grew as a superior replacement to Restlet for creating REST interfaces. Then in 2012, it was merged into Jazz Foundation as the preferred framework for creating REST interfaces in Jazz-based products.

Community

Juneau created a strong following over the years as an IBM Community Source project.

Slack has been used heavily by this community within IBM. A new Slack community has been set up as a transition for this community: https://juneau-cloud.slack.com

Core Developers

Juneau was started as an internal IBM Community Source project with a diverse set of community members from around the world.
The initial list of core developers started from that community and have an interest in seeing it opened up as an open source project outside of IBM.

Alignment

Juneau contains dependencies on the following projects:

- Apache Jena
- Apache HttpClient

Known Risks

Orphaned products

Most of the development of the project has been done by James Bognar. Much work has been done by this developer to make this a consumable library, and so there is a desire to continue it's development.

This reliance on a single developer is a known issue that we hope to remediate.

Inexperience with Open Source

The initial developers are very familiar with the use of existing open source products in commercial products and working on projects within the Eclipse community.

This is the first time the developers have submitted and worked on an Apache open source project.

Homogenous Developers

The two initial developers span two different companies: Salesforce and IBM.

We highly anticipate that the list of developers will quickly grow to include existing community members located in places such as China and India.

Reliance on Salaried Developers

It's expected that development will occur primarily on volunteer time as it has in the past.

Relationships with Other Apache Products

Juneau contains dependencies on the following projects:

- Apache Jena
- Apache HttpClient

A Excessive Fascination with the Apache Brand

Our interest in submitting this as an Apache-branded product is mostly due to its existing co-dependence on other Apache-branded products.

We plan on making this an open-source project regardless of whether it makes it out of incubator stage.

Documentation

Links to all documentation: https://sites.google.com/site/apachejuneau/links

Javadocs: https://juno-cloud-api-javadocs.mybluemix.net/index.html

Overview: https://juno-cloud-api-javadocs.mybluemix.net/overview-summary.html#TOP

Release Notes: https://juno-cloud-api-javadocs.mybluemix.net/overview-summary.html#ReleaseNotes

Initial Source

The source code is currently located in a Rational Team Concert source code repository internally in IBM.

Source and Intellectual Property Submission Plan

IBM legal has given approval for making this project available as an Apache-branded product. Peter Haumer will handle requests for more information.

All source code will be made available under the Apache V2 license.

External Dependencies

The dependences all have Apache compatible licenses.

Cryptography

Juneau contains no cryptographic resources.

Required Resources

Mailing lists

private@juneau.incubator.apache.org dev@juneau.incubator.apache.org

Subversion Directory

Not used.

Git Repository

Source code: https://git-wip-us.apache.org/repos/asf/incubator-juneau.git

Site code: https://git-wip-us.apache.org/repos/asf/incubator-juneau-site.git

Issue Tracking

JIRA Juneau

Other Resources

None.

Initial Committers

- James Bognar
- Peter Haumer
- Craig Chaney
- Min Idzelis
- David M Goddard
- Brian Laskey

Affiliations

- Salesforce: James Bognar, Craig Chaney, Min Idzelis
- IBM: Peter Haumer, David M Goddard, Brian Laskey

Sponsors

Champion

John D Ament - (johndament at apache dot org)

Nominated Mentors

John D Ament - (johndament at apache dot org)

Jochen Wiedmann - (jochen at apache dot org)

Craig L Russell - (craig dot russell at oracle dot com)

Unofficial Mentors

Stian Soiland-Reyes - (stain at apache dot org)

Sponsoring Entity

We request that the Incubator PMC sponsors this project