Article
Timothy Leavitt · Jun 4, 2020 3m read

Open Exchange

AppS.REST - a new REST framework for InterSystems IRIS

Over the past year or so, my team (Application Services at InterSystems - tasked with building and maintaining many of our internal applications, and providing tools and best practices for other departmental applications) has embarked on a journey toward building Angular/REST-based user interfaces to existing applications originally built using CSP and/or Zen. This has presented an interesting challenge that may be familiar to many of you - building out new REST APIs to existing data models and business logic.

As part of this process, we've built a new framework for REST APIs, which has been too useful to keep to ourselves. It is now available on the Open Exchange at https://openexchange.intersystems.com/package/apps-rest. Expect to see a few more articles about this over the coming weeks/months, but in the meanwhile, there are good tutorials in the project documentation on GitHub (https://github.com/intersystems/apps-rest).

As an introduction, here are some of our design goals and intentions. Not all of these have been realized yet, but we're well on the way!

Rapid Development and Deployment

Our REST approach should provide the same quick start to application development that Zen does, solving the common problems while providing flexibility for application-specific specialized use cases.

- Exposing a new resource for REST access should be just as easy as exposing it a a Zen DataModel.
- Addition/modification of REST resources should involve changes at the level being accessed.
- Exposure of a persistent class over REST should be accomplished by inheritance and minimal overrides, but there should also be support for hand-coding equivalent functionality. (This is similar to %ZEN.DataModel.Adaptor and %ZEN.DataModel.ObjectDataModel.)
- Common patterns around error handling/reporting, serialization/deserialization, validation, etc. should not need to be reimplemented for each resource in each application.
- Support for SQL querying, filtering, and ordering, as well as advanced search capabilities and pagination, should be built-in, rather than reimplemented for each application.
- It should be easy to build REST APIs to existing API/library classmethods and class queries, as well as at the object level (CRUD).

Security

Security is an affirmative decision at design/implementation time rather than an afterthought.

- When REST capabilities are gained by class inheritance, the default behavior should be to provide NO access to the resource until the developer actively specifies who should receive access and under what conditions.
- Standardized implementations of SQL-related features minimize the surface for SQL injection attacks.
- Design should take into consideration the OWASP API Top 10 (see: https://owasp.org/www-project-api-security)

Sustainability

Uniformity of application design is a powerful tool for an enterprise application ecosystem.

- Rather than accumulating a set of diverse hand-coded REST APIs and implementations, we should have similar-looking REST APIs throughout our portfolio. This uniformity should lead to:
    - Common debugging techniques
    - Common testing techniques
    - Common UI techniques for connecting to REST APIs
    - Ease of developing composite applications accessing multiple APIs
- The set of endpoints and format of object representations provided/accepted over REST should be well-defined, such that we can automatically generate API documentation (e.g., Swagger/OpenAPI) based on these endpoints.
- Based on industry-standard API documentation, we should be able to generate portions of client code (e.g., typescript classes corresponding to our REST representations) using third-party/industry-standard tools.

#API #Best Practices #Data Model #Framework #JSON #REST API #Security #InterSystems IRIS #InterSystems IRIS for Health #Open Exchange
Check the related application on InterSystems Open Exchange

Source URL: https://community.intersystems.com/post/appsrest-new-rest-framework-intersystems-iris