Article
Rob Tweed · Aug 30, 2018 5m read

Using QEWD.js to create REST APIs for IRIS

As a result of Evgeny's recent questions regarding the use of QEWD.js with IRIS in another post, I thought it would be a good idea to create a separate post focusing specifically on how to use QEWD.js to create REST APIs for IRIS.

QEWD.js is, of course, a Node.js framework, and all the REST API code can be written entirely in JavaScript.

The RealWorld Conduit reference application makes a great exemplar for this kind of thing, as the APIs are all published and there are numerous implementations of both front-ends and back-ends for it, using different technologies, frameworks and databases. So you'll be able to compare this QEWD /IRIS implementation with lots of other technologies and databases and see how they compare. You should find that QEWD + IRIS is extremely fast - it would be interesting to see some comparisons with other back-end implementations.

For details on the RealWorld Conduit reference application, see https://github.com/gothinkster/realworld

For the REST API specs: https://github.com/gothinkster/realworld/tree/master/api

My QEWD /IRIS REST back-end implementation of these APIs can be found here:

https://github.com/robtweed/qewd-conduit

I have an instance running here if you want to try it out:

http://54.86.139.171:8080/api

If you want, try installing one of the many front-end examples and point the UI at my example instance. For example, follow the installation instructions for the React/Redux version:

https://github.com/gothinkster/react-redux-realworld-example-app

Then edit the file /src/agent.js and change this line:

const API_ROOT = ' https://conduit.productionready.io/api';

To:

const API_ROOT = ' http://54.86.139.171:8080/api';

Then start up your front-end:

npm start

and point your browser at port 4100 on the server you've started it on, eg:

http://192.168.1.100:4100

The Conduit application should start up. Any data you see is coming from and being saved on an IRIS database on my server at 54.86.139.171

Getting your Own Instance Up and Running

Alternatively, to get your own instance of the back-end up and running against your own IRIS database, in summary:

- install IRIS as normal on your server, if you haven't already

- install Node.js on your IRIS server

  - for Windows, use the installer and instructions at *https://nodejs.org/en/download/*

  - for Linux I recommend using NVM. See *https://github.com/creationix/nvm*

- install QEWD.js. First decide on and create a directory for it, eg */qewd* or *C:\qewd*. Then:

   cd /qewd (or cd C:\qewd)
   npm install qewd

- Then just move a few files to new destinations:

   cd /qewd
   mkdir www
   cd www
   mkdir qewd-monitor
   cp /qewd/node_modules/qewd-monitor/www/bundle.js /qewd/www/qewd-monitor
   cp /qewd/node_modules/qewd-monitor/www/.html /qewd/www/qewd-monitor*
   cp /qewd/node_modules/qewd-monitor/www/.css /qewd/www/qewd-monitor*

- Now you can install the Conduit back-end application:

   cd /qewd
   npm install qewd-conduit

- Copy the appropriate Node.js interface file from the */bin* folder in your IRIS installation directory. eg if you're using Node.js v8, it's the file

   /bin/node800.iris

  Copy it to your QEWD directory's node_modules directory, eg */qewd/node_modules*

  Finally rename it to **node.iris**

- Copy the QEWD startup file to its proper location:

   cp /qewd/node_modules/qewd-conduit/startup/qewd-iris.js /qewd

- You should now edit the *qewd-iris.js* file:

  - change the management password to something else - any string value will do

  - note that it is already configured to expect a database of type *iris* - don't change this!

  - however, if necessary, adjust the IRIS database parameter values to match your IRIS setup

Provided you followed every one of the steps above, you should now be able to start up your Conduit back-end which will be listening for incoming REST requests on port 8080 (you can change this port by editing the qewd-iris.js file):

node qewd-iris

If you have installed a front-end instance (see earlier), adjust the REST URL to your server's address, eg if you are using the React/Redux version, edit the file */src/agent.js* and change this line:

const API_ROOT = ' https://conduit.productionready.io/api*';*

To:

const API_ROOT = ' http://192.168.1.101:8080/api*';* *(or to whatever your server's IP address is)*

and restart the front end.

Monitoring your QEWD Instance

QEWD includes a browser-based monitoring application (itself a QEWD application - a WebSocket on rather than a REST one) which allows you to check what's happening in your QEWD environment. To run it, point your browser at:

http://192.168.1.101:8080/qewd-monitor (adjust the IP address to the one you're running QEWD on)

You'll be asked for the QEWD Management password - that's the value you edited in your startup file (*~/qewd/qewd-iris.js*)

Provided you enter the correct password, you should now see the overview panel which shows you your overall configuration and also the activity in your QEWD Worker processes. You can stop any or all of these Worker processes at any time - it won't cause any harm to stop them: QEWD will automatically restart them as demand requires. If you stop the Master process, QEWD will be stopped and you'll need to restart it from the command line again.

You'll also see tabs which, if clicked, allow you to examine your IRIS database and active QEWD Sessions (which are also stored in IRIS)

QEWD treats IRIS as a persistent JSON database, and the *qewd-monitor* application allows you to drill down through the JSON hierarchy.

That's all for now - hopefully you'll have fun trying it out. I'll explain in more detail how the APIs use IRIS to store the data in another post. tl;dr it's all just persistent JSON as far as QEWD is concerned! See:

https://www.slideshare.net/robtweed/ewd-3-training-course-part-25-document-database-capabilities

#Node.js #REST API #JavaScript #InterSystems IRIS

Source URL:https://community.intersystems.com/post/using-qewdjs-create-rest-apis-iris