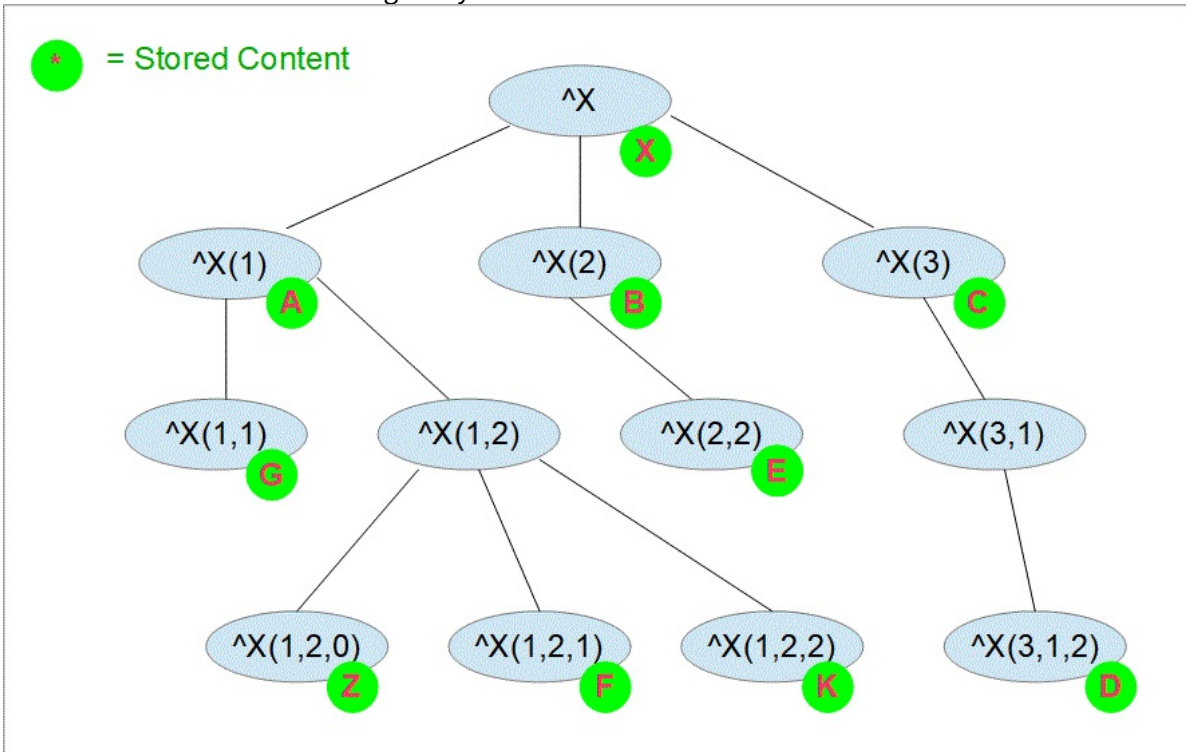
Article
Robert Cemper · Feb 9, 2022 2m read

GlobalToJSON-embeddedPython-pure#2

This package was triggered by the extra bonus points in the contest for writing the example in embedded Python only. Therefore **PURE**

Based on the previous example it looked like being a simple task. A mistake!
I met a bunch of issues that I had to workaround.
But I have to admit that there might be ready solutions and I just didn't find them.

1. The first step was to reduce the class to 2 methods [language = python] only
   - generating the JSON object file
   - loading the JSON object file
2. Within the import **iris.gref** class I was unable to find an equivalent to $DATA()
   - checked with **order()** method for subscripts to add 10 to my d
   - tried with **get()** to access a value and add to my d 1 for success
   - so my values were 0, 1, 10, 11 as I'm used to
3. To scan the global nodes I used method **query()** that didn't just return the next node but ALL nodes.
   - Even theoretical nodes. Referring to my model
   - 

   - I expected to see ^X,^X(1),^X(1,2),^X(1,2,0) ,^X(1,2,1),^X(1,2,2),^X(2), ... and **NEVER ever ^X(1,2),**
   - Value returned **None** for this unexpected Zombie. But None could also signal Nullstring.
4. This worked for rather large Globals, but I had my doubts about size and available memory.
   - A test with ^oddDEF (1410829 nodes) ran with no result and no error. So be careful.
   - In addition, access to PPG, and extended references (^|"USER"^|global) ar not supported
5. The next challenge was handling of **$LISTBUILD()**
   - I found no trace of $LISTVALID() equivalent or any access to elements with $LIST()
   - so $LIST() constructs are just seen and handled as binary strings.
   - ,{"sub": ["1"], "val": "\u000e\u0001Braam,Ted Q.\u0004\u0004\u0099\u00c8"}

    - You see this in the rather unreadable **val** elements of the generated JSON object.
    - Though it has the advantage that it can be stored as a simple string during the load operation.

Summary:

- The attempt to use and implement everything just using embedded Python is a mistake
- Keep access to the core structures of IRIS as a subject for ObjectScript which is grown with it
- But instead, use the range of Python libraries for those areas that are not covered by IRIS yet
- The success will lay in a well-balanced mix of both worlds
- It's no surprise that version 1 of embedded Python has limits that were not mentioned yet anywhere
- But it is the power of this community to learn from each other about such limits

Video

GitHub

#Embedded Python #Globals #JSON #InterSystems IRIS