Base Language - Feature #3855

implement equivalent support for REST (classic appserver, PASOE REST and WEB transports)

01/10/2019 10:26 AM - Greg Shah

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	90%
Category:		**Estimated time:**	0.00 hour
Target version:			
billable:	No	**vendor_id:**	GCD

Description

Related issues:

Related to Base Language - Feature #3310: add support for the equivalent of t... **Closed**

History

#1 - 01/17/2019 03:27 PM - Greg Shah

- Subject changed from implement equivalent support for REST (both as classic appserver and as the PASOE REST transport) to implement equivalent support for REST (classic appserver, PASOE REST and WEB transports)

There are 2 ways to expose REST web services in the 4GL:

- Nasty Wizard Approach
  - A development tool is used to generate artifacts that are used by OpenEdge to map REST requests to 4GL code call/return.
  - The artifacts include .pidl, .pamf, .xml and ultimately a .war that includes a .paar (also a Java archive) which has some .xml files and a .restoe document. It is the contents of the .paar that seem to be used for runtime.
  - There is a REST adapter in classic appserver that will use those artifacts and handle the runtime REST support.
  - The PASOE REST transport is this same adapter running inside PASOE. The artifacts are basically the same with some very slight differences in how the project is laid out.
- Clean DIY (Do It Yourself) Approach
  - This is only possible in PASOE.
  - The idea is to use the handler support in the WEB transport to implement your own REST services. This is similar to a servlet approach.

Both facilities will be provided from a runtime perspective.

We are NOT going to implement the wizard based development tool that generates the artifacts. Customers agree it is nasty and not the preferred approach anyway.

#2 - 04/05/2019 06:05 PM - Constantin Asofiei

Details about questions we need answered for the REST support. We need these answered for each mode (classic, PASOE, web handler):

- parameters (input and output):
  - all type combinations - https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/dvwsv%2Fparameter-mapping.html%23
  - mismatch between the value received by the REST call and the actual 4GL parameter definition (i.e. you send an character value and the

4GL parameter is an integer). Is it some automatic conversion happening - from i.e. "12" string to 12 integer? If yes, what happens if this automatic conversion is not possible? We need all data-type combinations here.
    - there is mention of TABLE and DATASET allowed in the body - we need examples for this; again, with all possible data-types at the TABLE/DATASET field level.
    - all ways of passing the parameters - via the URI path (/customer/{customerName}), via the URI query parameter, the request's body.
- error handling
    - parameter validation (number, position, mode - input/output).
    - the REST is configured to call some 4GL program, but that program is not found on the PROPATH
    - authentication failure
    - other possible ways for a REST call to fail?
    - what if there is an ERROR, QUIT, STOP, ENDKEY condition generated during the call? How is the remote side informed of this?
- can the RETURN-VALUE be exposed to the caller?
- tests for all supported HTTP verbs
- how is /customer/{custname} mapping the custname to a 4GL parameter?
- examples with REST annotations - targeting both external and internal procedures
- can internal procedures be targeted only by using REST annotations?
- JSON message formats

Security - for each mode (classic, PASOE, web handler):

- how is 4GL authenticating a REST call?
- how is the Agent handling this REST call running - in Stateless mode, something else?

Configuration, deploy - for each mode (classic, PASOE, web handler):

- how is the exposed REST service configured
- what artifacts are needed to run the REST services (the .paar file? something else?)
- what configuration files are generated and needed to run the REST services
- what is the OpenEdge configuration available for REST (number of Agents, log files, etc). This should be a document describing how a REST gets configured and deployed, what available options are for the user to setup the Agents, etc.
- tests to prove that all modes behave the same, at the 'executor side'. What I need to know is if the parameter transport, validation, error handling, etc is all the same when executing the REST call, regardless of the deployment approach (classic, PASOE, web handler).

We need tests for both the REST consumer (client) and the producer (server), all of this in 4GL.

#3 - 04/08/2019 12:40 PM - Constantin Asofiei

Regarding PASOE: we need to determine in which mode the REST requests are ran (I assume session-free), when handled by the PASOE Agent. Also, are there SESSION attributes or methods (like REMOTE) which are dependant on executing a REST request?

#4 - 04/08/2019 03:52 PM - Constantin Asofiei

Data Object Services examples and documentation for:

- testing with both ABL classes and external programs implementing the operations
- what artifacts are generated and needed to work with the Data Object Service
- what is the structure of the Progress Data Service Catalog?
- what are the annotations for each service
- what at are parameters for each service (OO method or internal procedure)
- how are RETURN ERROR and conditions handled (Both at runtime and as a response to the requestor).
- what are the URIs for each service - what is their structure
- how are parameters handled - format at the requestor side, how is processed at the service side. This is important as TABLE and DATASET are used as parameters, and we need to know the JSON (I assume) format for (de)serializing them, and how the field data-type maps from 4GL to JS.
- differences between /rest and /web/pdo transports
- what are the HTTP verbs for each operation and how do they map to the service - what happens if a service is invoked with a different verb?
- the API specification for OpenEdge.BusinessLogic.BusinessEntity and any other 4GL classes/interfaces which are used to implement the DataObjectService in 4GL.
- can a Data Object Service be executed without JSDO?
- how is the response handled by the requestor side (via AJAX? synchronous? asynchronous? something else?)
- security and authentication for a request - how is it implemented?
- how are the Agents executing the request behaving? Are they State-free? Is SESSION:REMOTE attribute set? Other SESSION attributes being used when executing the request?
- how is a Data Object Service configured - what options are available? Number of Agents? something else?
- when using a class or external program for the service, are they singleton - if so, when are they initiated?
- is an Agent bound to only one service? Because if singleton is used, then this state must be common for all Agents executing that request; can more than one Agent handle the same singleton class/persistent external program?
- examples should use minimal UI (is better without UI). I'd like to have some pure Javascript code detailing how these calls are performed.

Regarding JSDO - the documentation is pretty complex (https://documentation.progress.com/output/pdo/index.html#page/pdo%2Foverview-of-progress-data-objects%2C-services%2C-and.html) and the JSDO APIs even more so (https://documentation.progress.com/output/pdo/index.html#page/pdo%2Fjsdo-properties%2C-methods%2C-and-events-reference.html%23).

I assume we would need to understand how the OpenEdge JSDO classes (in Javascript) communicate with the 4GL side (so a Kendo UI app can integrate easily with FWD), and this is only possible by testing (and documenting) how each and every JSDO class and API behaves in regard to CRUD operations, transaction support, error management, etc.

#5 - 04/12/2019 05:21 PM - Greg Shah

In regard to the "clean" DIY (do it yourself) REST approach listed in #3855-1, this is implemented using the web handler facility in PASOE. I believe

this is done by hooking the web handler (@web/objects/web-handler.p). For this support we must implement our own version of this runtime code but we will not be converting that file (or any other of the dependent files or classes in the OpenEdge release). This means that we need a specification for the exact features in this handler, such that we can implement a replacement from scratch without reading or converting the source code.

This specification is not a test, but is a description of the functionality, API and control flow that must be implemented. To the degree that there are specific program names or classes that are referenced or subclassed by ABL code in order to use these facilities, those must be part of the specification as well.

Tests should be written to prove the specification's implementation.

#6 - 04/15/2019 07:36 AM - Greg Shah

- Related to Feature #3310: add support for the equivalent of the OpenEdge Web Services Adapter (and ProxyGen) and PASOE SOAP transport added

#7 - 04/15/2019 07:37 AM - Greg Shah

See #3310-9 for details on some expectations for the test suite.

#8 - 04/25/2019 04:20 PM - Greg Shah

Question from Marian:

> For classical appsrv there are some session-managed operating modes, do you need to have those covered as well or state-free should be enough - this is normally used by soap/rest.

#9 - 04/26/2019 05:39 AM - Constantin Asofiei

Greg Shah wrote:

> Question from Marian:
>
> > For classical appsrv there are some session-managed operating modes, do you need to have those covered as well or state-free should be enough - this is normally used by soap/rest.

For the classic appsrv, the session-managed modes (State-reset, State-aware. Stateless) are already supported by FWD. But I need some confirmation that an Agent executing a SOAP/REST service will always run in State-free mode - is this configuration automatic in OpenEdge - i.e. it automatically sets the mode to State-free and it can't be overridden by another configuration? If it can be overridden, how will the Agent behave?

On the other hand, for #3854, we need to make sure how the PASOE Session-managed and Session-free work, so we will need some tests for session-managed, too.

#10 - 06/06/2019 10:52 AM - Constantin Asofiei

Marian, I need an example of a openapi.openedge.export annotation for internal procedures, functions (if is possible in OpenEdge) and class methods (with void and non-void return type) - these will be REST services, with 2 or more PATH parameters.

The reason I need this: the .paar generated file (which is found in ./appsrv/rest/goldencode.war) contains a .restoe file, for which I need to decode its syntax, so I can process it.

#11 - 06/06/2019 11:11 AM - Constantin Asofiei

Marian, another issue: the .paar archive has a mapping.xml file with some info like:

```
        <mapping:rule action="None" resource="" source="${http.uristring}" target="${idl.param['inputParam1']}
" type="INTERFACE_PARAM"/>
        <mapping:rule action="None" resource="" source="${http.headers}" target="${idl.param['inputParam2']}"
type="INTERFACE_PARAM"/>
        <mapping:rule action="None" resource="" source="${http.header['X-MTC-Token']}" target="${idl.param['in
putParam3']}" type="INTERFACE_PARAM"/>

        <mapping:rule action="None" resource="" source="${idl.param['retVal']}" target="${http.statuscode}" ty
pe="INTERFACE_PARAM"/>
        <mapping:rule action="None" resource="" source="${idl.param['outputParam2']}" target="${http.body}" ty
pe="INTERFACE_PARAM"/>
```

I assume retVal is the returned value, but I don't know how the parameters are mapped to the http memebers - is this done via a GUI wizard in OpenEdge, when defining the REST application? If so, I need all possible combinations how to map these...

#12 - 06/06/2019 11:13 AM - Constantin Asofiei

The high-level approach for implementing the REST services is this:

1. use the .paar resources to identify any exposed REST services (this will be loaded at conversion time)
2. using the .paar info, and the legacy 4GL openapi.openedge.export annotations found in the source code, create new 4GL-level annotations (LegacyService), which will contain all the required info to export this as a REST service (PATH value, how to map parameters, etc). These new annotations can be used in 4GL development after FWD conversion, to create/modify/etc REST services directly in the 4GL code. They will convert to Java-style annotations.
3. the external programs/classes with exposed REST services will be marked in name_map.xml, so that we will be able to know the exposed services, but the annotation will have the details.
4. there will be a Jetty handler on a root path (like /rest/*) - this handler will intercept the request, process it, identify the REST service, execute it and send the response.
5. the executor will be either PASOE or classic appserver, depending either on the request path or some configuration in the directory. This shouldn't matter on how the REST gets processed.

#13 - 06/06/2019 12:52 PM - Greg Shah

I like the plan. From a 4GL development perspective, it seems quite easy. I wonder why the wizard is needed.

#14 - 06/11/2019 06:13 PM - Constantin Asofiei

3855a was created from trunk rev 11315. Rev 11316 contains the conversion changes and incomplete runtime. What I really really need to move forward with the compatibility is payload examples (request and response), for combinations of data types, return/non-return values, etc - this video https://youtu.be/e4_4Ls2XLOM?list=WL&t=281 shows a way to associate parameters with various parts of the request and response payloads, all these ways need to be tested and the payloads documented.

At least, I am aware of stuff like:

- for INPUT parameters, http.body, http.headers, http.uristring, http.header['header'], rest.queryparam['name'], rest.pathparam['name'], json.object['request'].boolean['argument'].
- for OUTPUT parameters, http.body, http.statuscode, json.object['response'].boolean['argument'].

I can make assumptions to some extent, but there may be more possible mapping types (see mappings.xml in the .paar generated file); plus, I need to know exactly how, for example, INPUT parameters mapped to http.headers look like, or how an OUTPUT parameter writes itself to the body or JSON. So this is two ways:

- for INPUT, how the request looks like (with all possible mappings), and how the arguments at 4GL level store these values
- for OUTPUT, how the 4GL level argument values look like, and how they look when they are stored in the response (be it http.body or a JSON element).

And I think I forgot to mention the unknown value until now...

For example, this is how a Java annotation is emitted:

```
@LegacyService(type = "REST", name = "pipe_char", path = "/pipe/char/{value}", verb = "GET", produces = "appli
cation/json", consumes = "application/json", parameters =
{
   @LegacyServiceParameter(name = "inputValue", type = "character", source = "${rest.pathparam['/pipe/char/{va
lue};value']}", input = true),
   @LegacyServiceParameter(name = "outputValue", type = "character", target = "${json.object['response'].strin
g['outputValue']}", output = true)
})
```

#15 - 06/12/2019 03:41 AM - Marian Edu

OK, in PSDOE (Eclipse) there is an editor for defining/editing those services in a project (if the 'ABL Rest Service' facet is added to the project). This is keeping all service definitions and mappings in a folder .settings in the project's root. I don't know of any other way to define those services, although the end result is that '.paar' archive that contains more or less the same mappings in a bit different format than the one used by the service editor inside Eclipse.

If your plan is to let them define those services using whatever they used before and don't bother with this service definition phase then you can probably start from the .paar content. For mapping this is what is available:

1. input

- URL (complete)
- QUERY parameter (name)
- PATH (only if path parameters defined in resource mapping - use {name} entries in resource URI (/rest/mymethod/{param1}/{param2})
- HTTP Method
- HTTP Headers (name)
- HTTP Cookies (name)
- Servlet Request
- Servlet Response
- Servlet Context
- Servlet Config

1. output

- Response Code
- HTTP Headers (name)
- HTTP Cookies (name)
- BODY (either full or when using parameters a JSON object is returned with a property for each of those output parameters)

We have a bunch of procedures for primitive and data structures input/output defined already, I will make now one that uses all various in/out mappings and re-export the service definition so you can maybe use the .paar content to figure it out how that works.

#16 - 06/12/2019 03:50 AM - Constantin Asofiei

Marian Edu wrote:

> We have a bunch of procedures for primitive and data structures input/output defined already, I will make now one that uses all various in/out mappings and re-export the service definition so you can maybe use the .paar content to figure it out how that works.

Great, thanks! Yes, I'm using the .paar content as input during conversion, to determine what 4GL code maps to what service.

When you can, please document the payloads, too. And I also need some examples on how internal procedures/functions or OO methods map to REST services.

#17 - 06/12/2019 08:49 AM - Marian Edu

Constantin Asofiei wrote:

> Marian Edu wrote:
>
> > We have a bunch of procedures for primitive and data structures input/output defined already, I will make now one that uses all various in/out mappings and re-export the service definition so you can maybe use the .paar content to figure it out how that works.
>
> Great, thanks! Yes, I'm using the .paar content as input during conversion, to determine what 4GL code maps to what service.
>
> When you can, please document the payloads, too. And I also need some examples on how internal procedures/functions or OO methods map to REST services.

Added a somehow more complex mappings and exported the service, the content of the .paar was added into /appsrv/rest/export/paar folder.

The input mappings are pretty easy to read from the resourceModel.xml, see the last resource entry there ('/mapping/{value}') - first operation(POST) is handled by an external procedure, second one (GET) is an internal function and last one (PUT) is an internal procedure. Note the '..' between external procedure name and internal function/procedure one in operation's name when internal entries are used.

However, this can be misleading since not all input parameters are visible in that file so you have to dig into **mapping.xml** for all the details. Look for resource name '_mapping_value' and for each operation/verb there is the complete mapping between source/target.

1. http.uristring - the full URL
2. rest.queryparam['name'] - query string parameter
3. rest.pathparam['name'] - path parameter (defined in resource URL)
4. rest.verb - http method
5. http.headers - all http headers
6. http.header['name'] - http header
7. rest.cookieparam['name'] - http cookie
8. rest.formparam['name'] - form parameters (POST/PUT only)
9. rest.context.httpservletrequest
10. rest.context.httpservletresponse
11. rest.context.httpservletcontext
12. rest.context.httpservletconfig
13. http.body - the whole response body, can be json or whatever
14. json.object['request'] - json elements if body is a json object (POST/PUT only)

Mind you there are two separate sections for each operation, one for input and one for output... however the XML should be pretty straight forward to parse.

#18 - 06/12/2019 11:24 AM - Constantin Asofiei

Marian Edu wrote:

> Mind you there are two separate sections for each operation, one for input and one for output... however the XML should be pretty straight forward to parse.

Thanks for the help. I'm already parsing the .paar artifacts.

I have a question about your latest mapping.xml - the mappings_ip..mapTest is referenced only in the mapping:mapInput section, the output parameters do not appear in the mapping:mapOutput section.

Is this an OpenEdge bug?

#19 - 06/12/2019 08:12 PM - Constantin Asofiei

The issue in note #3855-18 I think is not a bug, but a feature - OpenEdge allows only a portion of the parameters to be mapped to the request/response (as I've found this same usage in some production code).

Marian, to be more clear about what else I need - I don't have a way to run the REST services in OpenEdge, that's why I mentioned that I need captures of the request/response payloads, plus a log file with how the input arguments are populated from the request, and how the output arguments look like before writing them to the response. Otherwise, I'm running blind.

#20 - 06/13/2019 03:48 AM - Marian Edu

Constantin Asofiei wrote:

> The issue in note #3855-18 I think is not a bug, but a feature - OpenEdge allows only a portion of the parameters to be mapped to the request/response (as I've found this same usage in some production code).

In this case it was a 'bug' as in we did forgot to map the output - this is done in a separate tab in the editor :(
The 'feature' you mention is that indeed not all parameters needs to be mapped - that holds true for both input and output, input will be left unknown (?) if not mapped to anything from request and output will simply be ignored and not present in response.

Fixed the mapping and pushed an update.

> Marian, to be more clear about what else I need - I don't have a way to run the REST services in OpenEdge, that's why I mentioned that I need captures of the request/response payloads, plus a log file with how the input arguments are populated from the request, and how the output arguments look like before writing them to the response. Otherwise, I'm running blind.

We have a test suite in SoapUI made for this - appsrv/rest/test/fwd-soapui-project.xml - that shows how the requests looks like. We didn't save any responses in bazaar as I was under impression you will get an appsrv/pasoe instance setup on your side to run those tests. We have various appsrv instances for different session modes and one for PASOE, if you need help to have those set-up on your side we can help but if that is not the case I wonder what we can do... beside saving responses for each requests so you can match those agains your implementation. If you need us to save requests as well we can do that as well, otherwise you maybe can just use the SoapUI project since everything is there including conditions on expected response.

#21 - 06/13/2019 04:22 AM - Constantin Asofiei

Marian Edu wrote:

> We didn't save any responses in bazaar as I was under impression you will get an appsrv/pasoe instance setup on your side to run those tests.

I am not able to set this up locally at this time (and I'm not sure when I will be able to).

> We have a test suite in SoapUI made for this - appsrv/rest/test/fwd-soapui-project.xml - that shows how the requests looks like.

Hmm... I assume this is independent of how the REST service is implemented, right? If so, this can be used to test the FWD implementation. But what is missing are input parameter validation for the 4GL code. And even so, I need to take a look at the "raw" values for the request and response payloads (and here I mean unformatted/plaintext, just as they are sent/received). You can use a Chrome REST client extension, manually perform some calls, and log:

- on 4GL side - the input and output parameter values
- from the REST client side, for the request:
  - the URL
  - the body (if method allows it)
  - the headers
- from the REST client side, for the response:
  - the headers
  - the body

I think is enough to expand mappings.p and mappings_ip.p, which have the mappings you listed in #3855-17. For example, I have no idea what rest.context.httpservletrequest wants to hold.

#22 - 06/13/2019 04:29 AM - Igor Skornyakov

I have experience with implementing REST "mocks" as Spring boot apps. It is much more convenient for testing than SoapUI. If we have specs and a number of requests are not too big, I can implement another one in one day at the weekend.

#23 - 06/13/2019 06:50 AM - Marian Edu

Constantin Asofiei wrote:

> I think is enough to expand mappings.p and mappings_ip.p, which have the mappings you listed in #3855-17. For example, I have no idea what rest.context.httpservletrequest wants to hold.

Those seems to be straight from Tomcat:

- request, config and context come as json array with entries that have 'theName' and optionally 'theValue' properties :)
  - httpservletrequest
    [{\"theName\":\"AuthType\"},{\"theName\":\"ContextPath\",\"theValue\":\"\\/goldencode\"},{\"theName\":\"Method\",\"theValue\":\"POST\"},{\"theName\":\"PathInfo\",\"theValue\":\"\\/goldencodeService\\/mapping\\/value\"},{\"theName\":\"PathTranslated\",\"theValue\":\"D:\\\\Progress\\\\OpenEdge11.7\\\\WRK\\\\oepas1\\\\webapps\\\\goldencode\\\\goldencodeService\\\\mapping\\\\value\"},{\"theName\":\"QueryString\"},{\"theName\":\"RemoteUser\"},{\"theName\":\"RequestedSessionId\"},{\"theName\":\"RequestURI\",\"theValue\":\"\\/goldencode\\/rest\\/goldencodeService\\/mapping\\/value\"},{\"theName\":\"RequestURL\",\"theValue\":\"http:\\/\\/localhost:8810\\/goldencode\\/rest\\/goldencodeService\\/mapping\\/value\"},{\"theName\":\"ServletPath\",\"theValue\":\"\\/rest\"}]
  - httpservletcontext
    [{\"theName\":\"ContextPath\",\"theValue\":\"\\/goldencode\"},{\"theName\":\"ServerInfo\",\"theValue\":\"Apache Tomcat\\/8.5.34\"}]
  - httpservletconfig
    [{\"theName\":\"ServletName\",\"theValue\":\"OERestAdapter\"}]

- response seems to be just the toString of a Java class :)
  org.apache.cxf.jaxrs.impl.HttpServletResponseFilter@76e5a259

#24 - 06/13/2019 07:18 AM - Marian Edu

Some more details about input data:

- request cookies (rest.cookieparam['name']), the value sent as input to mapped parameter is like 'name=value' (not just the value) if the cookie is set, null otherwise.
- request headers, only the header value is sent not 'name: value'
- request body
  - if you need the whole body then http.body gives you everything (doesn't have to be json)
  - if parameters mapping is used (json.object['request']) then the body must be a JSON object that includes a 'request' object with all those parameters (none is mandatory, no error is thrown if missing but null sent to input). This is how data is sent for mappings POST request.

```
{
    "request": {
        "jsonChar": "test",
        "jsondate": "2020-07-29",
        "jsonLogical": true
    }
}
```

- response cookies, all cookies are sent inside the Set-Cookie header with no expiration or anything like domain, https only, etc
- response body
  - if output param is mapped to the whole body (http.body) then that is written as-is (can be anything)
  - if output parameters are created in response body (json.object['response']) then response is a JSON object wrapped in a 'response' object:

```
{"response": {
   "outPath": "value",
   "outMethod": "POST",
   "outType": "application/json;charset=UTF-8",
   "outInt": -1,
   "outdate": "2020-07-29",
   "outLogical": false
}}
```

#25 - 06/13/2019 07:26 AM - Greg Shah

Marian: In regard to the pre-defined SoapUI projects that are checked in, can the open source version of SoapUI be used to run this testing?

#26 - 06/13/2019 07:31 AM - Greg Shah

Igor Skornyakov wrote:

> I have experience with implementing REST "mocks" as Spring boot apps. It is much more convenient for testing than SoapUI. If we have specs and a number of requests are not too big, I can implement another one in one day at the weekend.

This is very interesting. These mocks are used as calls into real REST or SOAP services which then test the responses against expected results? Or are these mocks used to replace the real REST/SOAP API with a mocked version?

I want to understand your proposal so I can evaluate how it may help us.

For now, let's get the SoapUI running and use that since it is already prepared. We can schedule time later to move to this approach as makes sense.

#27 - 06/13/2019 07:40 AM - Marian Edu

Greg Shah wrote:

> Marian: In regard to the pre-defined SoapUI projects that are checked in, can the open source version of SoapUI be used to run this testing?

Sure, this is what we've used for test suites - both for soap and rest.

#28 - 06/13/2019 07:44 AM - Igor Skornyakov

Greg Shah wrote:

> This is very interesting. These mocks are used as calls into real REST or SOAP services which then test the responses against expected results? Or are these mocks used to replace the real REST/SOAP API with a mocked version?
>
> I want to understand your proposal so I can evaluate how it may help us.
>
> For now, let's get the SoapUI running and use that since it is already prepared. We can schedule time later to move to this approach as makes sense.

I had a mock of the server side in mind. The client-side part is much more simple and one can just use JUnit/TestNG. The advantage of the Spring boot server mock that it is very easy to deploy (one needs a JRE and single jar file). Another good thing is that after the basic functionality is implemented it is very easy to modify the details of the behavior including adding new requests' types and configurable simulation of failures.

#29 - 06/13/2019 07:59 AM - Greg Shah

> I had a mock of the server side in mind.

We have other cases where this will be very useful. For example, one of the current applications we are working on uses a separate REST service for authentication. This makes the testing environment more complex to setup and support. Your Spring Boot solution is a good fit there.

> The client-side part is much more simple and one can just use JUnit/TestNG.

We are still considering our unit testing approach, but I agree that we would want to surface such testing there.

#30 - 06/13/2019 09:33 AM - Constantin Asofiei

Greg Shah wrote:

> > I had a mock of the server side in mind.
>
> We have other cases where this will be very useful.

Another use case is testing the implementation of various OpenEdge builtin classes, like:

```
USING OpenEdge.Net.HTTP.IHttpRequest.
USING OpenEdge.Net.HTTP.IHttpResponse.
USING OpenEdge.Net.HTTP.ClientBuilder.
USING OpenEdge.Net.HTTP.RequestBuilder.
```

#31 - 06/13/2019 02:46 PM - Constantin Asofiei

I think there is a problem in FWD's State-free mode for appserver agents - these should remain bound until all persistent procedures ran on it are deleted. But currently we are not tracking this properly. For REST, I have an in-server FWD authenticated session (a pool of these), and each one will handle a REST request; it will connect to the appserver (which is in the same FWD server), and delegate the call to it. A certain REST FWD session can handle both singleton and single-run services - but if a singleton was ran, it MUST remain bound to the agent which initially ran the persistent program; when this is mixed with single-run, the Agent gets unbound.

I think I know how to fix this (i.e. use a set to keep all persistent programs at the agent, and unbind it only when this set gets empty), but now I'm chasing/fixing issue related to having a OO instance as 'entry point' - currently this assumes there is an existing external program on the stack, which is not the case.

Greg, a question: for in-server FWD sessions, SessionManager.getSession() returns null - this is normal, right? My problem is that this session must be headless, and I can either rely on the process account setting in the directory (to mark it headless), or on the fact that SessionManager.getSession() is null - but I'm worried that this will interfere with the in-server real FWD client.

#32 - 06/13/2019 04:19 PM - Constantin Asofiei

An addendum to the previous note: I think I remembered it wrong (been a while since I implemented this). State-free forces an Agent to be bound on the first PERSISTENT invocation - after this, it can handle only requests to run internal entries from that persistent program. If the client (on the same connection) runs another external program, then another Agent will pick it up.

The real issue is when we have multiple REST services, and they are all ran in singleton. The first time a REST worker gets a singleton service, it will bind an Agent. So all Agents can end up bound to singleton services, and not allow any other services to be ran. And, considering that we don't know which worker gets which service, you can get in a case where no agents are available to process it (as the request ended up on a worker which doesn't know the singleton, and it can't create a new one, as all agents are bound).

This might be an issue even in 4GL, if all Agents get bound, and you will not be able to process a request unless is picked up by a worker which has a singleton instance for the service. And if all Agents are bound, you will not be able to run single-run services, either.

I think the solution is to route the request to a worker which has a singleton instance for this service, and queue it to it. But this means we need to decide how many workers are allowed for each singleton instance, as we shouldn't leave only 1. So the number of workers (and appserver Agents) will be equal to the sum of agents dedicated to singleton services, plus any additional agents which will process single-run services.

#33 - 06/13/2019 06:14 PM - Constantin Asofiei

3855a rev 11320 contains a stable runtime and conversion.

There are two files which need finished javadocs (ServiceSupport and RestHandler), but it can be reviewed.

If testing passes, then the branch can be committed once I finish the javadocs.

Note that I didn't run the SoapUI testing - I plan to do this tomorrow.

#34 - 06/13/2019 06:59 PM - Greg Shah

> Greg, a question: for in-server FWD sessions, SessionManager.getSession() returns null - this is normal, right? My problem is that this session must be headless, and I can either rely on the process account setting in the directory (to mark it headless), or on the fact that SessionManager.getSession() is null - but I'm worried that this will interfere with the in-server real FWD client.

Hmmm. Yes, that is probably normal but it would also be the case for any random uninitialized thread on the server I think. I don't think relying upon getSession() is a good idea.

BTW, I'm not sure that single process mode still works. It was experimental and is not expected to be used in the near future.

Can we leverage the fact that all appserver agents are running in batch mode? The client driver knows this fact. If we need it on the server side, we can send it up.

#35 - 06/14/2019 02:39 AM - Marian Edu

Constantin Asofiei wrote:

> An addendum to the previous note: I think I remembered it wrong (been a while since I implemented this). State-free forces an Agent to be bound on the first PERSISTENT invocation - after this, it can handle only requests to run internal entries from that persistent program. If the client (on the same connection) runs another external program, then another Agent will pick it up.

This is true, as in every time one runs a persistent procedure on the appsrv then that agent is bound to the client regardless of the operating more - so even for stateless and state-free.

> The real issue is when we have multiple REST services, and they are all ran in singleton. The first time a REST worker gets a singleton service, it will bind an Agent. So all Agents can end up bound to singleton services, and not allow any other services to be ran. And, considering that we don't know which worker gets which service, you can get in a case where no agents are available to process it (as the request ended up on a worker which doesn't know the singleton, and it can't create a new one, as all agents are bound).
>
> This might be an issue even in 4GL, if all Agents get bound, and you will not be able to process a request unless is picked up by a worker which has a singleton instance for the service. And if all Agents are bound, you will not be able to run single-run services, either.

The singleton/single-run options were added specifically to overcome this bounding issues with persistent procedures. When those options are used instead of persistent then the agent is not bound (stateless or state-free). In this case the persistent procedure is not even run on the appsrv before an actual call to one internal entry in that procedure is made, the only difference is that for single-run after the internal entry was called the procedure is deleted automatically (it was only run persistent to make the call to the IP) while for singleton the procedure remains loaded so it won't be loaded persistent on subsequent request to IP (unless the request lands on an agent that doesn't have it loaded already or someone did delete it from session). The internal entries from those single-run/singleton procedures should be idempotent normally and should not count on any 'state', in practice this is not really the case cause the data always change in time and most of the time those api's work with that data :)

We have tests for showing that in appsrv/test for persistent/single-run/singleton, those are 4GL tests and I just realize although we did test persistent from SOAP we didn't cover single-run/singleton for SOAP so we will add some extra tests. Imho it's crazy they actually let one instantiate persistent procedures over SOAP requests, those have different endpoints and needs to be deleted through another soap call in the end to free the agent... I do not know of any case when something like that was used but hey, it's possible so someone out there might well be using it :(

For REST they did took a more sane approach, to call internal procedures/functions/methods the only option is to use singe-run/singleton so there is no need to clean-up, no agent remains bound. This is not set in the service definition but using annotations in the procedure/class (executionMode).

#36 - 06/14/2019 03:30 AM - Marian Edu

OK, just a quick update... knew there was something but slipped my mind :(

In proxygen single-run/singleton are only allowed for java and .net open clients, so for SOAP this is not permitted. While I think it should be the only way to run internal entries and you might well choose to allow it in your implementation, guess is hard sometimes to follow the same mistakes someone did before :)

#37 - 06/14/2019 06:22 AM - Constantin Asofiei

Marian Edu wrote:

> The singleton/single-run options were added specifically to overcome this bounding issues with persistent procedures. When those options are used instead of persistent then the agent is not bound (stateless or state-free).

Something confuses me - in OpenEdge, there is the RUN ... {SINGLETON|SINGLE-RUN} SET hproc ON SERVER hsrv. Later, you invoke it via RUN internalProc IN hproc. So, if the Agent is not bound, how does OpenEdge know that the RUN ... IN hproc. must reach the hproc handle, if the Agent is not bound? Because you are using a handle, not an external program name... or is OpenEdge ignoring the handle and it just checks if there is a the program name referenced by the received handle exists in that Agent?

In FWD, now I'm emulating the singleton (and single-run actually) via these steps, at the REST worker (not appserver Agent):

1. if there is a known handle for the procedure, it uses that. Otherwise, it invokes the procedure persistent (thus Agent becomes bound).
2. it invokes the internal entry (which may be function, internal procedure or OO method).
3. if the mode is single-run, it deletes the handle, freeing the Agent. Otherwise, the Agent remains bound to this external program, and this is the reason I'm using different REST worker pools for singleton mode, for each REST resource defined in this mode.

   For REST they did took a more sane approach, to call internal procedures/functions/methods the only option is to use singe-run/singleton so there is no need to clean-up, no agent remains bound.

This doesn't make sense, unless they use something like "invoke this function in this external program name (NOT handle)" and the Agent decides to create or re-use an instance of that program. But what if the PROPATH gets changed, and the external program name is resolved to another program? And what happens if there is a delete this-procedure. when running the external program in singleton mode?

And even if state-free is assumed to not keep any state, an Agent with a singleton program MUST keep its state...

> This is not set in the service definition but using annotations in the procedure/class (executionMode).

Exactly, I'm relying on that.

#38 - 06/14/2019 06:50 AM - Marian Edu

Constantin Asofiei wrote:

> Something confuses me - in OpenEdge, there is the RUN ... {SINGLETON|SINGLE-RUN} SET hproc ON SERVER hsrv. Later, you invoke it via RUN internalProc IN hproc. So, if the Agent is not bound, how does OpenEdge know that the RUN ... IN hproc. must reach the hproc handle, if the Agent is not bound? Because you are using a handle, not an external program name... or is OpenEdge ignoring the handle and it just checks if there is a the program name referenced by the received handle exists in that Agent?

It's not ignoring the handle, that handle is a valid handle on the client side but at that point nothing was instantiated on the server side. You can even run a non existing procedure singe-run/singleton and that will not throw an error and the handle set is going to be valid. So, basically, the handle procedure holds information about the external procedure that is going to be ran on the server. When you actually call something on that handle then the external procedure in ran on the server and if not found this will throw an error. After calling that internal entry the procedure handle on the server side gets either deleted (single-run) or remains loaded (singleton), only when running singleton the agent check to see if there is a procedure already loaded that matches the name.

> In FWD, now I'm emulating the singleton (and single-run actually) via these steps, at the REST worker (not appserver Agent):
>
> 1. if there is a known handle for the procedure, it uses that. Otherwise, it invokes the procedure persistent (thus Agent becomes bound).
> 2. it invokes the internal entry (which may be function, internal procedure or OO method).
> 3. if the mode is single-run, it deletes the handle, freeing the Agent. Otherwise, the Agent remains bound to this external program, and this is the reason I'm using different REST worker pools for singleton mode, for each REST resource defined in this mode.

This is not what happens in 4GL, as said single-run and singleton does not left the agent bound.

> This doesn't make sense, unless they use something like "invoke this function in this external program name (NOT handle)" and the Agent decides to create or re-use an instance of that program.

As said the 'program name' is probably kept somewhere in the procedure handle so when making a call of an internal entry in that handle on the client the server receives something like what you've said: 'call this entry in that procedure'.

> But what if the PROPATH gets changed, and the external program name is resolved to another program? And what happens if there is a delete this-procedure. when running the external program in singleton mode?

If PROPATH changes then the new procedure is pickup for singe-run, for singleton only if there isn't an instance loaded in which case is used regardless of the fact that after propath change another file would have been resolved.
If one deletes the procedure handle - either because it has the handle or is walking the session procedures tree - then the procedure is not there so when another call is made to singleton it will be loaded then just like in the first place. There is no protection built in 4GL that doesn't let you delete a procedure that was loaded due to a single-run/singleton run.

> And even if state-free is assumed to not keep any state, an Agent with a singleton program MUST keep its state...

There is no actual state other than the one kept internally by the singleton procedure/class but since the agent is not bound to the client the next request might well be executed on another client. A different process, different procedure, different state... hence the requirement for idempotence.

#39 - 06/14/2019 07:32 AM - Constantin Asofiei

Marian Edu wrote:

> So, basically, the handle procedure holds information about the external procedure that is going to be ran on the server.

OK, thanks for confirming this. Although I don't understand why you would make something prone to errors, and say 'don't do that', when you can just say 'you can't do that'. And here I refer to what you say, "the requirement for idempotence", which must be enforced by the programmer.

So this handle is just a specification of the program to run (if ever will be ran). There are still some questions to answer here, like when are the arguments for the external program evaluated (think of a function call passed as argument) and what happens with any output arguments - are they evaluated the time the initial RUN for the external program is executed via RUN ... SINGLETON/SINGLE-RUN, or when the internal entry is ran. And for SINGLE-RUN, are the arguments evaluated each time an internal entry is executed? Anyway, I'll work on this on a different task.

#40 - 06/14/2019 07:39 AM - Marian Edu

Constantin Asofiei wrote:

> OK, thanks for confirming this. Although I don't understand why you would make something prone to errors, and say 'don't do that', when you can just say 'you can't do that'. And here I refer to what you say, "the requirement for idempotence", which must be enforced by the programmer.

Well, hence the expression shoot himself in the foot :)

> So this handle is just a specification of the program to run (if ever will be ran).

Yes, they even call that handle a 'proxy handle' in the docs... if you don't run anything on that procedure 'proxy' handle then nothing happens on the appsrv.

> There are still some questions to answer here, like when are the arguments for the external program evaluated (think of a function call passed as argument) and what happens with any output arguments - are they evaluated the time the initial RUN for the external program is executed via RUN ... SINGLETON/SINGLE-RUN, or when the internal entry is ran. And for SINGLE-RUN, are the arguments evaluated each time an internal entry is executed? Anyway, I'll work on this on a different task.

Singleton and single-run procedure can't have parameters so no need to worry about those, unless of course you decide FWD will support parameters in that case but then you define the specs.

#41 - 06/14/2019 07:47 AM - Marian Edu

OK, finally found a way to make the form parameters work... fighting that since yesterday and was constantly greeted with a 'null pointer exception' with the full java stack-trace that goes with it and a 500 internal server error on each request made through SoapUI :(

It turns out the content type accepted by the rest adapter is 'application/x-ww-form-urlencoded' and 'form/multipart' constantly crashed the server even if nothing was sent in the body. Again, I don't know of anyone sending form data over a REST request but since this things is present in the service editor and looks like being supported (to some extent) by the rest adapter there we are writing test for it.

So, Content-Type: 'application/x-ww-form-urlencoded' and url encoded parameters (name=value&name=value) in the body is what is required. Mapped form (rest.formparam['name']) parameters are not mandatory, the parameter names are case sensitive (resolved on Java side), only the value is sent in the mapped input parameter.

#42 - 06/14/2019 08:34 AM - Constantin Asofiei

Greg Shah wrote:

> Hmmm. Yes, that is probably normal but it would also be the case for any random uninitialized thread on the server I think. I don't think relying upon getSession() is a good idea.
> Can we leverage the fact that all appserver agents are running in batch mode? The client driver knows this fact. If we need it on the server side, we can send it up.

I'm not referring here to appserver Agents, I mean the FWD process account which is used to create the REST worker's context (so that the requests don't run on the server's context). I wanted to hard-code this, but as is not that easy, I removed that check and added the directory setting to mark the account as headless.

#43 - 06/14/2019 09:22 AM - Constantin Asofiei

Marian, regarding the REST implementation, I need some answers for these questions:

1. how is an unknown value for an output (or input) argument passed? Is the argument just missing?
2. parameter type mismatch - how is this handled?
3. when writing or reading a parameter using http.body as source, I think the format is Java-style; here I need to know how the format will look like, for each Java type, so please confirm these assumptions:
   - BigDecimal - only decimal can be associated to it; what is the decimal separator? Is it locale-specific (dot vs comma)?
   - Boolean - only logical can be associated to it
   - byte[] - only memptr and rowid can be associated to it. How does the value look like - is it base64 encoded? Or just bytes?
   - GregorianCalendar - all date, datetime, datetime-tz - what is the date format in this case? Does it depend on some locale-specific format, like mm/dd/yyy vs dd/mm/yyy?
   - Integer - integer can be associated to it
   - Long - int64 can be associated to it
   - String - character and longchar can be associated to it
   - Object - I see in the .restoe that these are associated with table, dataset and clob - so how does these look like (serialized) for the request and response?
4. can extent arguments be used?
5. error processing - STOP/QUIT/ERROR - how are these handled? Is it a 500 - server error? Or is otherwise sent in the response?
6. error processing - request on a path not associated with a REST service - is this 500 - server error?
7. DATASET/TABLE - I need to know how the input and output looks like, at the JSON level.
8. http.headers - I need an example how the payload looks like, for INPUT arguments.
9. can the RETURN-VALUE be exposed to the caller?

10. how is authentication performed? I assume some CLIENT-PRINCIPAL object will be used - if so:
    - how are the user credentials configured in the Tomcat server in OpenEdge?
    - how is the authentication performed? Does the client call some API to authenticate, and this returns a token or sets a cookie?
    - once authentication is performed, what will the request headers/cookies have, for any subsequent REST call?
    - details about how the authentication works (does the token expire? what happens if a request with unauthenticated state is performed? etc)

Also, do you have a WebHandler specification/details yet? My thinking we need to implement the (currently skeleton) Java WebHandler class, and any associated helper classes, like the ones mentioned in note #3855-30.

#44 - 06/14/2019 10:24 AM - Constantin Asofiei

I think 3855a 11321 is a good candidate to release, although there are some remaining issues (which are most likely runtime only). I still need a MAJIC run, to clear false negatives.

Please review.

#45 - 06/14/2019 02:49 PM - Greg Shah

Code Review Task Branch 3855a Revision 11321

This is a really good update.

1. In legacy_services.rule, this code looks wrong <action on="false">tref = createProgressAst(prog.bool_false, "true", tref)</action>. Although brew.xml may ignore the text when emitting the boolean, using "true" for the text is confusing to the reader.

2. The LegacyService instance returned from ServiceSupport.resolveRestAnnotation() will generate NPEs if the service name is not in the operations map. This occurs for anything that dereferences service without protection.

3. In the progress.g the oea_annotation needs modifications:

- The any_symbol_at_all! should be s:any_symbol_at_all! { hide(#s); }.
- LPARENS! should use lparens (it drops the token and hides it)
- RPARENS! should use rparens (it drops the token and hides it)
- COMMA! should use comma (it drops the token and hides it)

In oea_annotation_array:

- COMMA! should use comma
- The RBRACKET! should be r:RBRACKET! { hide(r); }.

4. In RestHandler.initialize() is the use of authenticateServer() meant to associate a server context with the newly created threads?

5. In RestHandler.initialize(), I'm confused about the use of AppServerHelper.connect(). Am I misremembering the purpose of that? I thought it was used by client code (e.g. code that used to use the Java Open Client and now is using our version to remotely access the appserver). This usage is inside the server process.

6. I would prefer for most (if not all) of the inner classes of RestHandler and ServiceSupport to be moved out as top level classes in the same package. It is more likely that code will be designed/improved/used as common code in that case.

#46 - 06/14/2019 02:56 PM - Constantin Asofiei

Greg Shah wrote:

> 1. In legacy_services.rule, this code looks wrong <action on="false">tref = createProgressAst(prog.bool_false, "true", tref)</action>. Although brew.xml may ignore the text when emitting the boolean, using "true" for the text is confusing to the reader.
> 2. The LegacyService instance returned from ServiceSupport.resolveRestAnnotation() will generate NPEs if the service name is not in the operations map. This occurs for anything that dereferences service without protection.
> 3. In the progress.g the oea_annotation needs modifications:

Fixed.

> 4. In RestHandler.initialize() is the use of authenticateServer() meant to associate a server context with the newly created threads?

No. sm.authenticateServer is meant to authenticate 'from within the FWD process'. The context will be the one for the configured process (non-server) alias.

> 5. In RestHandler.initialize(), I'm confused about the use of AppServerHelper.connect(). Am I misremembering the purpose of that? I thought it was used by client code (e.g. code that used to use the Java Open Client and now is using our version to remotely access the appserver).

The idea here is this: we don't have yet in-server FWD clients (which for example don't need OS access and other stuff). So I rely on Appserver clients to execute the requests.

> This usage is inside the server process.

Is inside the server process, but non using the server's context.

> 6. I would prefer for most (if not all) of the inner classes of RestHandler and ServiceSupport to be moved out as top level classes in the same package. It is more likely that code will be designed/improved/used as common code in that case.

I think I'll move these to a p2j/rest package, keeping them in p2j/util doesn't feel right.

#47 - 06/14/2019 04:17 PM - Constantin Asofiei

3855a rev 11325 fixes the review issues.

I can run the REST tests via SoapUI, and I can find the answers for some questions from #3855-43, but the assertions look incomplete. For example, there are assertions like .*The value was not an INTEGER - this is not enough, I need to know the entire response body.

#48 - 06/14/2019 06:19 PM - Constantin Asofiei

According to this: https://community.progress.com/community_groups/openedge_development/f/19/t/56961 the date format is ISO 8601, at the request/response payloads.

#49 - 06/15/2019 05:13 AM - Constantin Asofiei

Marian, do you have a payload for the response, when there is a message like The value was not an INTEGER? The SoapUI assertion is like .*The value was not an INTEGER, but the body sent by Jetty differs. This might not be a problem (as the client should rely on the HTTP status code).

#50 - 06/15/2019 03:28 PM - Constantin Asofiei

Greg, when do you want me to merge 3855a?

#51 - 06/15/2019 04:21 PM - Greg Shah

You can merge to trunk whenever you are ready.

#52 - 06/17/2019 08:17 AM - Constantin Asofiei

Branch 3855a passed testing and was merged to trunk rev 11318, and archived.

There remaining issues are:

- decimal - what is the precision at the 4GL side? For example, a value of 2.123456789123456789 is received by the response JSON as 2.123456789123456 - I can't tell if the decimals are stripped by the Tomcat servlet (before the 4GL code is invoked) or by the 4GL code.
- decimal - the formatting is not quite right currently, for large numbers.
- memptr - this is transported as byte[] - so, what is the value at the 4GL side, if you send a base64-encoded? My assumption would be that it sends to the remote side the string value, but it doesn't quite make sense when you have an integer argument - this fails to match...
- is it possible to have a raw argument?
- QUIT/STOP/ERROR is not emitting proper HTTP Status code (and a QUIT actually terminates the FWD Agent, which I'm not sure is OK).
- TABLE/DATASET transport is not implemented at all.

Marian, from note #3855-42, I need answers for:

- 1, 4, 8, 9, 10 questions.
- TABLE/DATASET - more detailed examples how these are (de)serialized, via the request and response.

#53 - 06/17/2019 11:45 AM - Constantin Asofiei

The REST web transport is not necessarily related to REST - the custom implementation of OpenWedge.Web.WebHandler acts like a custom servlet in Java (so the response/request can be anything, as the business logic decides what to do with it). I plan to look at the 4GL documentation for this and the supported classes mentioned in #3855-30 and work on their implementation.

#54 - 06/17/2019 12:01 PM - Marian Edu

Constantin Asofiei wrote:

> The REST web transport is not necessarily related to REST - the custom implementation of OpenWedge.Web.WebHandler acts like a custom servlet in Java (so the response/request can be anything, as the business logic decides what to do with it). I plan to look at the 4GL documentation for this and the supported classes mentioned in #3855-30 and work on their implementation.

This is indeed not part of REST, it's a 'WEB' transport added in PASOE - some are using it for REST implementation just like WebSpeed but in fact it's more like WebSpeed - you get the request object and you can write the response as see fit. I will add a sample for it tomorrow just to get an idea.

#55 - 06/17/2019 12:07 PM - Greg Shah

Marian: As noted in #3855-5, for the web handler we need both a specification and testcases.

#56 - 06/17/2019 03:44 PM - Constantin Asofiei

Marian Edu wrote:

> I will add a sample for it tomorrow just to get an idea.

Yes, please do, so I can start work until you have more complete details. Please include any artifacts needed to configure these handlers (i.e. mapping of handler to the URI path handled by it).

#57 - 06/18/2019 07:35 AM - Constantin Asofiei

Marian, something confuses me - how does OpenEdge know that the OpenEdge.Web.WebResponseWriter instance created in the request processing is the exact instance which will send the response? What if someone creates two instances? I'm missing something, and I don't know what... in the examples I have, the HTTP response or the response writer is not linked with the request, or in any other way, it just 'exists' and OpenEdge knows to write it to the HTTP response? Maybe it attaches this writer to STDOUT (or other stream) which in turn is redirected to the real HTTP response?

#58 - 06/18/2019 07:45 AM - Marian Edu

Constantin Asofiei wrote:

> Marian, something confuses me - how does OpenEdge know that the OpenEdge.Web.WebResponseWriter instance created in the request processing is the exact instance which will send the response? What if someone creates two instances? I'm missing something, and I don't know what... in the examples I have, the HTTP response or the response writer is not linked with the request, or in any other way, it just 'exists' and OpenEdge knows to write it to the HTTP response? Maybe it attaches this writer to STDOUT (or other stream) which in turn is redirected to the real HTTP response?

It was confusing for me as well when we've added support for it in akera.io :)
The response is not an object that is returned from web handler's methods but something that is used just to prepare that information and then everything is streamed back to the 'web stream' using the OpenEdge.Web.WebResponseWriter (https://documentation.progress.com/output/oehttpclient/oe117/).

#59 - 06/18/2019 07:56 AM - Constantin Asofiei

Marian Edu wrote:

> The response is not an object that is returned from web handler's methods but something that is used just to prepare that information and then everything is streamed back to the 'web stream' using the OpenEdge.Web.WebResponseWriter (https://documentation.progress.com/output/oehttpclient/oe117/).

Hmm... so you mean there is a 'hidden web stream' which is linked to any OpenEdge.Web.WebResponseWriter created by the programmer? Because this instance is explicitly created and not obviously linked to some implicit instance... so I think I need some tests to show what happens if multiple instances are used, and:

- each instance executes the write, flush, close separately from the other:

```
writer1.write().
writer1.flush().
writer1.close().

writer2.write().
writer2.flush().
writer2.close().
```

- the call are mixed, like:

```
writer1.write().
writer2.write().

writer1.flush().
writer2.flush().

writer1.close().
writer2.close().
```

or

```
writer1.write().
writer1.flush().

writer2.write().
writer2.flush().

writer1.close().
writer2.close().
```

And also what happens if you:

- flush/write after the close is performed
- you do not close/flush the stream

#60 - 06/18/2019 08:13 AM - Marian Edu

Constantin Asofiei wrote:

> Branch 3855a passed testing and was merged to trunk rev 11318, and archived.
>
> There remaining issues are:
>
> - decimal - what is the precision at the 4GL side? For example, a value of 2.123456789123456789 is received by the response JSON as 2.123456789123456 - I can't tell if the decimals are stripped by the Tomcat servlet (before the 4GL code is invoked) or by the 4GL code.
> - decimal - the formatting is not quite right currently, for large numbers.

Max precision is 50, scale is 10 so I would say it's pretty odd you get values with more than 10 decimals.

- memptr - this is transported as byte[] - so, what is the value at the 4GL side, if you send a base64-encoded? My assumption would be that it sends to the remote side the string value, but it doesn't quite make sense when you have an integer argument - this fails to match...

- is it possible to have a raw argument?

nope, not supported by the rest service designer - no procedure that have raw parameter can be used.

- QUIT/STOP/ERROR is not emitting proper HTTP Status code (and a QUIT actually terminates the FWD Agent, which I'm not sure is OK).

QUIT/STOP returns HTTP 504, no content
ERROR returns 500 with json information about the error.

> Marian, from note #3855-42, I need answers for:
>
> - 1, 4, 8, 9, 10 questions.
> - TABLE/DATASET - more detailed examples how these are (de)serialized, via the request and response.

Temp-table, dataset are written using JSON format (write-json), it depends to what the parameter is mapped - either to the body (full) or one parameter inside the response just like any other parameters but serialization is JSON and it matches the one of write-json.

Added the full log of SoapUI test suite ran against pasoe instance and appsrv/rest adapter in appsrv/rest/test folder.

#61 - 06/18/2019 08:20 AM - Marian Edu

Constantin Asofiei wrote:

> Marian Edu wrote:
>
> > The response is not an object that is returned from web handler's methods but something that is used just to prepare that information and then everything is streamed back to the 'web stream' using the OpenEdge.Web.WebResponseWriter (https://documentation.progress.com/output/oehttpclient/oe117/).
>
> Hmm... so you mean there is a 'hidden web stream' which is linked to any OpenEdge.Web.WebResponseWriter created by the programmer? Because this instance is explicitly created and not obviously linked to some implicit instance... so I think I need some tests to show what happens if multiple instances are used, and:
>
> - each instance executes the write, flush, close separately from the other:

There is a preprocessor variable defined for WebSpeed - WEBSTREAM - and that is most probably used by the web handler, it's probably just an OO wrapper on top of WebSpeed functionality. The 'web' transport in PASOE is imho just the same as in WebSpeed.

We had no idea we had to write tests for all classes in OpenEdge.Web package, not that it can't be done if you really need it :(

#62 - 06/18/2019 09:45 AM - Constantin Asofiei

Marian Edu wrote:

> We had no idea we had to write tests for all classes in OpenEdge.Web package, not that it can't be done if you really need it :(

The legacy builtin classes we need tests for are:

```
./Net/MessagePart.cls
./Net/UriEncodingTypeEnum.cls
./Net/MultipartEntity.cls
./Net/HTTP/HttpResponse.cls
./Net/HTTP/Filter/Writer/BodyWriterRegistry.cls
./Net/HTTP/HttpHeaderCollection.cls
./Net/HTTP/ClientOptions.cls
./Net/HTTP/IHttpResponse.cls
./Net/HTTP/HttpMessage.cls
./Net/HTTP/ConfigBuilder.cls
./Net/HTTP/Cookie.cls
./Net/HTTP/IHttpClient.cls
./Net/HTTP/IHttpRequest.cls
./Net/HTTP/HttpHeaderBuilder.cls
./Net/HTTP/BuilderRegistry.cls
./Net/HTTP/HttpHeader.cls
./Net/HTTP/Lib/ClientLibraryBuilder.cls
```

```
./Net/HTTP/ClientBuilder.cls
./Net/HTTP/IHttpMessage.cls
./Net/HTTP/RequestBuilder.cls
./Net/HTTP/StatusCodeEnum.cls
./Net/URI.cls
./Net/ISupportMultipartEntity.cls
./Web/WebResponseWriter.cls
./Web/IWebRequest.cls
./Web/WebHandler.cls
./Web/WebResponse.cls
```

And any support other support classes (from implements or inherits) which you find are required (like Progress.Web.IWebHandler or Progress.IO.OutputStream).

#63 - 06/18/2019 09:55 AM - Marian Edu

Constantin Asofiei wrote:

> Marian Edu wrote:
>
> > We had no idea we had to write tests for all classes in OpenEdge.Web package, not that it can't be done if you really need it :(
>
> The legacy builtin classes we need tests for are:
> [...]
>
> And any support other support classes (from implements or inherits) which you find are required (like Progress.Web.IWebHandler or Progress.IO.OutputStream).

This might then be part of the OO implementation - there are however a large number of classes there since this is part of the 4GL REST client implementation. Testing all methods/properties for all those can be quite time consuming and I have no idea if it does fit in the current SOW :(@Greg what do you think?

#64 - 06/18/2019 10:02 AM - Marian Edu

Greg Shah wrote:

> Marian: As noted in #3855-5, for the web handler we need both a specification and testcases.

Working on testcases now, for specification what exactly is expected? A word document explaining how that works, the subject is documented to some extent by PSC so no idea if we need to duplicate that :(

1. WebHandler - https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/pdsoe%2Fworking-with-pas-for-openedge-for-webapp.html%23
2. Web Transport - https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/ompas%2Fmanaging-web-transports.html%23
3. Quick Intro - https://docs.progress.com/bundle/service-developer-guide/page/Use-a-Web-Handler.html

#65 - 06/18/2019 10:31 AM - Greg Shah

> Testing all methods/properties for all those can be quite time consuming and I have no idea if it does fit in the current SOW :(

@Greg what do you think?

Is this something that can be done in a few weeks?

#66 - 06/18/2019 10:44 AM - Greg Shah

> for specification what exactly is expected? A word document explaining how that works, the subject is documented to some extent by PSC so no idea if we need to duplicate that :(

A pseudo-code description of the logic and state changes is the most important thing. We are going to implement a clean room version of this implementation and we want it to match the features exactly.

Instead of a word doc, I prefer the text to be posted in Redmine. The PSC documentation is useful on its own, but it does not explain or diagram the logic and state machine of the WebHandler, WebTransport or web-disp.p. These things are the objective of the spec. If there is some other "core web component" that is also part of this logic/state machine, please include it in the spec.

You don't need to write specifications for every OO class/method that is used, we are looking for the core logic/state machine. However, where the

implementation depends on some utility or helper, the specific class/method should be clear in the spec so that we can know where the dependencies are.

#67 - 06/18/2019 12:19 PM - Constantin Asofiei

Marian Edu wrote:

> Max precision is 50, scale is 10 so I would say it's pretty odd you get values with more than 10 decimals.

It's not me who's getting these values, it's from your tests - there is one pipe_decimal test which has as input 2.123456789123456789 and gets 2.123456789123456.

- memptr - this is transported as byte[] - so, what is the value at the 4GL side, if you send a base64-encoded? My assumption would be that it sends to the remote side the string value, but it doesn't quite make sense when you have an integer argument - this fails to match...

For memptr and decimal, please log to file the received value - it's not clear what the value is at the 4GL side, the tests show only the sent value and the received value.

#68 - 06/18/2019 12:38 PM - Marian Edu

Greg Shah wrote:

> A pseudo-code description of the logic and state changes is the most important thing. We are going to implement a clean room version of this implementation and we want it to match the features exactly.

OK, I will try to cover most of use-cases I'm aware of.

> You don't need to write specifications for every OO class/method that is used, we are looking for the core logic/state machine. However, where the implementation depends on some utility or helper, the specific class/method should be clear in the spec so that we can know where the dependencies are.

I'm just thinking, those classes provided by PSC are still 4GL and some of them can be extracted in the source code format so you could probably 'translate' those directly instead of implementing them... beside those changed quite frequently lately and there are important differences between what gets shipped with different OE versions :(

#69 - 06/18/2019 01:14 PM - Marian Edu

Greg Shah wrote:

> Is this something that can be done in a few weeks?

Probably, although writing tests to cover all methods in JsonObject takes more that I've expected and it looks like the team is somehow getting tired of testing... I need to give them a break to do some real development, there is a reason adoption of test driven development didn't happen :)

#70 - 06/18/2019 01:30 PM - Greg Shah

> I'm just thinking, those classes provided by PSC are still 4GL and some of them can be extracted in the source code format so you could probably 'translate' those directly instead of implementing them... beside those changed quite frequently lately and there are important differences between what gets shipped with different OE versions :(

We suspect that that code is under the community license, but it is unclear enough that we are being conservative. This means we are not planning to convert that code since we don't want any possible infringement of PSC copyrights. If we were sure that the code was released under the community share license, then it would be OK.

For now, we must plan on a clean room implementation.

> Probably, although writing tests to cover all methods in JsonObject takes more that I've expected and it looks like the team is somehow getting tired of testing... I need to give them a break to do some real development, there is a reason adoption of test driven development didn't happen :)

Constantin: Can you make a list of the specific methods that are needed? Perhaps that can reduce the necessary tests.

#71 - 06/20/2019 06:57 PM - Constantin Asofiei

3855b was created from trunk rev 11319.

3855b rev 11320 contains the SINGLE-RUN/SINGLETON support for appservers, and also RUN statements. Need to cleanup and finish some testing with RUN statement (appserver calls via REST services seems to work).

#72 - 06/20/2019 07:40 PM - Constantin Asofiei

Greg Shah wrote:

> Constantin: Can you make a list of the specific methods that are needed? Perhaps that can reduce the necessary tests.

When I made these skeletons, I used the docs to make them complete. So, I think we can start with these:

```
openedge.net.http.httpheader.name,
openedge.net.http.httpheader.tostring,
openedge.net.http.httpmessage.contenttype,
openedge.net.http.httpresponse.statuscode,
openedge.net.http.ihttpmessage.contentlength,
openedge.net.http.ihttpmessage.entity,
openedge.net.http.ihttpmessage.getheaders,
openedge.net.http.ihttprequest.method,
openedge.net.http.statuscodeenum.ok,
openedge.net.http.statuscodeenum.unauthorized,
openedge.net.uri.decode,
openedge.web.iwebrequest.getcontextvalue,
openedge.web.iwebrequest.getpathparameter,
openedge.web.iwebrequest.pathinfo,
openedge.web.webresponse.setheader,
openedge.web.webresponsewriter.close,
openedge.web.webresponsewriter.flush,
openedge.web.webresponsewriter.write,
```

The openedge.web.webhandler and openedge.web.webresponsewriter are sub-classed, so we need to test what is the default behaviour of the methods.

These ones I think are used only for invoking HTTP requests from 4GL code, and not serving them; so I think they can be postponed, unless the ones above are dependent on them.

```
openedge.net.http.clientbuilder.build,
openedge.net.http.clientbuilder.client,
openedge.net.http.filter.writer.bodywriterregistry.registry,
openedge.net.http.httpheaderbuilder.build,
openedge.net.http.httpheaderbuilder.header,
openedge.net.http.httpheaderbuilder.value,
openedge.net.http.httpheadercollection.put,
openedge.net.http.ihttpclient.execute,
openedge.net.http.ihttpmessage.entity,
openedge.net.http.ihttpresponse.statuscode,
openedge.net.http.ihttpresponse.statusreason,
openedge.net.http.requestbuilder.addheader,
openedge.net.http.requestbuilder.delete,
openedge.net.http.requestbuilder.get,
openedge.net.http.requestbuilder.httpversion,
openedge.net.http.requestbuilder.post,
openedge.net.http.requestbuilder.put,
openedge.net.http.requestbuilder.request,
openedge.net.messagepart.contentid,
openedge.net.messagepart.headers,
openedge.net.multipartentity.addpart,
openedge.net.multipartentity.boundary,
```

#73 - 06/24/2019 07:51 AM - Constantin Asofiei

Marian, can you provide a simple example of a doGet/doPost implementation of WebHandler? It doesn't have to do something complicated, just pass back something in the response's body. I'm most interested in how this is exposed in OpenEdge (so the handler can be mapped to a URL), and to have a simple example which is known to work.

Thanks!

#74 - 06/24/2019 08:06 AM - Marian Edu

Constantin Asofiei wrote:

> Marian, can you provide a simple example of a doGet/doPost implementation of WebHandler? It doesn't have to do something complicated, just pass back something in the response's body. I'm most interested in how this is exposed in OpenEdge (so the handler can be mapped to a URL), and to have a simple example which is known to work.
>
> Thanks!

There is a web handler I've started to work on in *PASOEContent/WEB-INF/openedge/FwdHandler.cls*, so far only GET is 'handled' and returns either plain text content or json (customer info in json format if customer number sent in path params is found).

PASOEContent/WEB-INF/openedge is what PSDOE (eclipse) uses when deploying web handlers to a PASOE instance. The resource mapping is fairly simple, you can map the same handler for multiple resources, the order of the mapping is kinda important. PSDOE have a service definition for weh handlers as well but that it's not that complicated as the REST mapper, you can find the 'artifact' in *.services/Expose/web/* - fdwHanlder is the service name and inside it's a service.properties file that is basically something like handlerN=Web Handler Class:Resource[,Resource]*

When deployed this ends-up in PASOE properties file (openedge.properties) under web app WEB section and the list gets split for each resource mapped to the handler:

```
[oepas1.goldencode.WEB]
   handler1=FwdHandler: /fwdHandler/{id}/{value}
   handler2=FwdHandler: /fwdHandler/{id}
   handler3=FwdHandler: /fwdHandler
```

But I don't know if you really have to use the same properties file like PASOE.

#75 - 06/26/2019 03:26 PM - Constantin Asofiei

Marian Edu wrote:

> When deployed this ends-up in PASOE properties file (openedge.properties) under web app WEB section and the list gets split for each resource mapped to the handler:

Do the path parameter names ({id}, {value}) have any meaning on 4GL side? I mean, are they mapped automatically to something or accessed via this name, using some OpenEdge method?

#76 - 06/26/2019 03:55 PM - Marian Edu

Constantin Asofiei wrote:

> Marian Edu wrote:
>
> > When deployed this ends-up in PASOE properties file (openedge.properties) under web app WEB section and the list gets split for each resource mapped to the handler:
>
> Do the path parameter names ({id}, {value}) have any meaning on 4GL side? I mean, are they mapped automatically to something or accessed via this name, using some OpenEdge method?

There is a method in IWebRequest - GetPathParameter - that returns the value for a path parameter, the list of available parameters is available through property PathParameterNames.

Those should already be used in the FwdHandler class, Mihai was extending that today to some extent and we're making tests in soapui so we will update the full http log tomorrow and I will also try to outline the use of those web handlers as for testing the OpenEdge.Web package that isn't really possible out of the web (handler) environment. The WebRequest is using the web-context system handle to get information about the web request and WebResponseWriter uses the web-stream so none can be tested outside of the webspeed/pasoe environment so most probably you will have to figure out what this does from the source code and the test we're writing with the web handler :(

#77 - 06/30/2019 05:06 PM - Constantin Asofiei

3855b rev 11323 finishes the Singleton and Single-Run support for REST/RUN, adds TABLE output and other misc fixes. There are still other issues to fix/add (TABLE input, DATASET input/output, PASOE support), but this version can be reviewed, tested and released.

Marian: please check in 4GL the 2.123456789123456789 issue from #3855-67; if the REST call has as input 2.123456789123456789, what value does 4GL see? Because in your SOAP UI log, the REST response value is 2.123456789123456 - and I don't know where the decimals are trimmed, before or after the value is received by 4GL. Add also a test which reaches the limit of the maximum precision (50).

#78 - 07/01/2019 03:36 AM - Marian Edu

Constantin Asofiei wrote:

> Marian: please check in 4GL the 2.123456789123456789 issue from #3855-67; if the REST call has as input 2.123456789123456789, what value does 4GL see? Because in your SOAP UI log, the REST response value is 2.123456789123456 - and I don't know where the decimals are trimmed, before or after the value is received by 4GL. Add also a test which reaches the limit of the maximum precision (50).

Precision is still 50 but the scale seems to be moved to 15, so the decimal can have maximum 15 decimals. The logic seems to be like follow, the number of decimals are 'trimmed' to 15 and if the resulted number precision is not higher than 50 then this is the value received in 4GL side. If the precision of the number is higher that 50 then the rest adapter throws an 500 error status, it does connect to the appsrv but the only thing you see in the logs is an 'incomplete request' - in the rest adapter log it's a cryptic error about 'input result set error, stream protocol error' so I assume the value is not checked on the Java side and it fails in the protocol somewhere when the Java open client try to send that to the appsrv.

1. 99999999999999999999999999999999999.999999999999999 - works, 35 digits plus 15 decimals)
2. 99999999999999999999999999999999999.999999999999999999999... - works, 35 digits plus any number of decimals (those are trimmed to first 15 decimals)
3. 99999999999999999999999999999999999.999999999999999 - fails, more than 35 digits when 15 or more decimals are set
4. 9999999999999999999999999999999999999999999999999999 - works, any combination between 35 and 50 digits if total including decimals not higher than 50 (36+14, 37+13, ...)
5. 99999999999999999999999999999999999999999999999999999 - fails, anything more than 50 digits even if no decimals

#79 - 07/01/2019 04:55 AM - Marian Edu

The decimal value sent back is rounded and/or concerted to scientific format depending on it's value, by trial and error this seems to be the behavior:

1. when number is higher than 10 millions the scientific notation is used
2. otherwise there might be rounding but the results there left me puzzled

- 9999998.999999999 -> 9999998.999999998 (why is that loosing one using on last decimal escapes me)
- 9999998.9999999991-> 999999 (if more than 9 decimals it rounds)
- 9999998.123456789-> 9999998.12345679
- 9999998.123456785-> 9999998.123456785
- 9999998.123456781-> 9999998.123456782 ???

I can't really figure our what is the exact 'logic' there, especially when they just seems to add up to the last decimal for no reason :(

The test case and log was updated in bazaar.

#80 - 07/01/2019 05:50 AM - Constantin Asofiei

Marian, do you know how the WebHandler instances are created - singleton or single-run?

#81 - 07/01/2019 06:11 AM - Marian Edu

Constantin Asofiei wrote:

> Marian, do you know how the WebHandler instances are created - singleton or single-run?

Those are for persistent procedures, for WebHandler on each request a new instance is created, the corresponding method is called passing in the IWebRequest and the instance goes away as soon as the response is sent back (the method returns). The Web transport of PASOE is really the same thing like WebSpeed just with an OO wrapper around.

#82 - 07/01/2019 10:22 AM - Constantin Asofiei

Marian, how do wildcards work for the webhandler path? Can you have something like /path/to/* and another like /* ?

#83 - 07/01/2019 05:08 PM - Greg Shah

Code Review Task Branch 3855b Revision 11324

Good update.

1. In JsonResponseArguments.writeArgumentInt(), is there a danger of ClassCastExceptions with the unconditional casting of the val parameter?

2. In ControlFlowOps.invoke(InvokeConfig cfg), it seems you are taking inspiration from recent efforts to rewrite some language statement usage using chaining. Is the plan to shift all RUN statement and function call usage to this new API?

3. InvokeConfig.readExternal() and InvokeConfig.computeWriting() would both benefit from using named constants instead of hard coded int values when shifting bits.

#84 - 07/01/2019 05:39 PM - Constantin Asofiei

Greg Shah wrote:

> 1. In JsonResponseArguments.writeArgumentInt(), is there a danger of ClassCastExceptions with the unconditional casting of the val parameter?

The way I coded the caller, it shouldn't happen; but I'll switch it to toString(), to make it safe, instead of casting.

> 2. In ControlFlowOps.invoke(InvokeConfig cfg), it seems you are taking inspiration from recent efforts to rewrite some language statement usage using chaining. Is the plan to shift all RUN statement and function call usage to this new API?

At some point, yes. But for now, the testing is too time expensive. This API was just moved from InvokeConfig and improved a little (and is still used

by Call).

> 3. InvokeConfig.readExternal() and InvokeConfig.computeWriting() would both benefit from using named constants instead of hard coded int values when shifting bits.

OK, I will improve it.

#85 - 07/02/2019 04:35 AM - Marian Edu

Constantin Asofiei wrote:

> Marian, how do wildcards work for the webhandler path? Can you have something like /path/to/* and another like /* ?

Constantin, there is an implicit wildcard suffix used for the URI pattern so wether or not you add that to the path it's always there. The explanation on how the right web handler is picked-up is more or less described here - [[https://knowledgebase.progress.com/articles/Article/what-does-it-make-pasoe-to-call-the-default-webhandler]]

On the web handler side you get all that information either using PathInfo, URI or PathParameterNames - the latter actually includes the 'template' parameter which is always there and that is the URI as defined in the URI mapping, if you strip that from beginning of PathInfo you're probably left with whatever come afterwards but then it defies the purpose of path parameters doesn't?

#86 - 07/02/2019 11:24 AM - Marian Edu

We're almost done with testing the net/web OO classes so probably the only thing left to do here is the 'business catalog' (jsdo) support. This is not a trivial task so I would rather make sure this feature is needed before doing anything about it. @Greg, do you have an answer for that already, is there any customer looking for this or it remains for latter?

#87 - 07/02/2019 02:56 PM - Greg Shah

JSDO is not needed at this time.

#88 - 07/03/2019 08:29 AM - Constantin Asofiei

The review fixes are in 3855b rev 11326.

3855b 11327 contains the conversion and infrastructure for WebHandler service support. Runtime (i.e. legacy OE classes) is not there yet.

#89 - 07/05/2019 01:41 PM - Greg Shah

Code Review Task Branch 3855b Revision 11331

I'm good with the changes.

The progress.g change in particular should be fine. Before qualified OO members, the reserved keywords couldn't be duplicated but now this change is needed.

#90 - 07/05/2019 02:02 PM - Constantin Asofiei

I'm testing 3855b 11331, if it passes, I'll merge it, as it contains some conversion fixes that we need.

#91 - 07/07/2019 02:37 PM - Constantin Asofiei

Branch 3855b passed runtime/conversion testing and was merged to trunk rev 11322. It contains:

More improvements for #3855:

- Conversion support and infrastructure for WebHandler services.
- Added TABLE output support and other misc fixes for REST invocations.
- Support for SINGLE-RUN and SINGLETON, for RUN statement, appsever calls and REST services.

Misc fixes:

- Added a stub for OUTPUT TO ... LOB-DIR
- PUBLISH can have a POLY event expr. Fixed a DYN-FUNC issue when emitting the cast for an object function. Added JsonObject:Write(character).
- Fixed SUBSTRING var name used with ASSIGN statement. Refs #4131
- Removed COLUMN-CODEPAGE for temp-tables defined in an interface. Refs #3815
- Fixed cross-namespace collisions between class members, variables and buffers, defined in different compilation units (super-classes). Refs #4125
- Added another browse:add-calc-column(character,character,character,string) version. Refs #4125

#92 - 07/09/2019 03:27 PM - Constantin Asofiei

Marian, please commit the SoapUI text log for the FWDHandler tests.

#93 - 07/10/2019 08:43 AM - Constantin Asofiei

Marian, another question: any idea what a HTTP_idContext context value means? And what IWebRequest.getContextNames usually returns?

#94 - 07/10/2019 09:04 AM - Marian Edu

Constantin Asofiei wrote:

> Marian, another question: any idea what a HTTP_idContext context value means? And what IWebRequest.getContextNames usually returns?

Sorry for the late reply, was out for a few days. The log file for running the soapui test case on webhandler was uploaded - appsrv/rest/test/fwd-soapui-webhandler.log.

As for context you will find all context names along with their values (as returned by PASOE) in the log file, just search for 'ContextNames' and there should be the full list in a JSON object in one of the responses. Most are what we call CGI variables or server context, specific to the web handler could be URI_TEMPLATE and URI_FINAL_MATCH_GROUP.

#96 - 12/03/2019 12:38 PM - Greg Shah

From Constantin:

> We need some help in understanding the WebHandler APIs, but not from the WebHandler.cls, here I mean the Web request and response supporting classes. At least for the WebResponseWriter.cls, there are lots of unknowns on how this should work. If we could have some more in-depth specification (or testing) of these APIs, it would be great.

Marian: What is the effort needed to put this together?

#97 - 12/04/2019 05:51 AM - Marian Edu

Greg Shah wrote:

> From Constantin:
>
> > We need some help in understanding the WebHandler APIs, but not from the WebHandler.cls, here I mean the Web request and response supporting classes. At least for the WebResponseWriter.cls, there are lots of unknowns on how this should work. If we could have some more in-depth specification (or testing) of these APIs, it would be great.
>
> Marian: What is the effort needed to put this together?

There are already some tests for those in OO\web folder in bazaar. Basically what WebResponseWriter is doing is act like a wrapper (OO) for the WEBSTREAM stream where you can just throw whatever content back to the client. You give it a WebResponse in constructor and then do Open and Close on it to send that response to the stream - sample WebHandler in PASOEContent\WEB-INF\openedge\FwdHandler.cls.

Writing tests out of the WEB environment is not really possible at lest for the WebResponseWriter, if you have more specific questions or needs for test cases we will try to answer those of course.

#98 - 12/04/2019 07:38 AM - Constantin Asofiei

Marian, my questions related to WebResponseWriter are:

- when is the flush happening? What happens if you try to write after the flush?
- what kind of handle is WEBSTREAM?

- there are multiple write APIs, with character, longchar, memptr - what happens if you mix these? Are they written immediately to the WEBSTREAM? Is the value cached and flushed at some point - if so, what is the order of writing these?
- what happens if the params for write is unknown, or invalid memptr offset/length)?
- there are writeBody, writeHeaders, WriteHttpPreamble, WriteStatusLine, WriteCookies - these can be called in any order; so where do they get written in the end? Can you call these more than once? Also, what exactly is each one of these writing (what is the source of the data being written)?

#99 - 04/22/2020 10:42 AM - Constantin Asofiei

Marian, another question please: who/when sets the SERVLET_SRVR_DEBUG and SERVLET_SERVER_APP_MODE CGI vars? I can't find an equivalent for these in the servlet...

#100 - 04/22/2020 11:23 AM - Marian Edu

Constantin Asofiei wrote:

> Marian, another question please: who/when sets the SERVLET_SRVR_DEBUG and SERVLET_SERVER_APP_MODE CGI vars? I can't find an equivalent for these in the servlet...

Damn, never used those so it took some time to find the answer... both are configuration options set for `web applications` in PASOE: srvrAppMode and srvrDebug.

```
[fwd-pasoe.ROOT.WEB]
    adapterEnabled=1
    defaultCookieDomain=
    defaultCookiePath=
    defaultHandler=OpenEdge.Web.CompatibilityHandler
    srvrAppMode=development
    srvrDebug=1
    wsRoot=/static/webspeed
```

#101 - 04/22/2020 11:39 AM - Constantin Asofiei

Marian Edu wrote:

> Constantin Asofiei wrote:
>
> > Marian, another question please: who/when sets the SERVLET_SRVR_DEBUG and SERVLET_SERVER_APP_MODE CGI vars? I can't find an equivalent for these in the servlet...
>
> Damn, never used those so it took some time to find the answer... both are configuration options set for `web applications` in PASOE: srvrAppMode and srvrDebug.

Thanks. Looks like these are documented in openedge.properties.README.

#102 - 04/27/2020 12:39 PM - Constantin Asofiei

4231b rev 11490 contains improvements related to TABLE, DATASET and more. A summary of what is left:

For REST:

- some edge cases when the type doesn't match (like a date for year 1450 gets a 7 days difference in FWD, or some cases as memptr vs raw). Or the decimal with 15 precision instead of 10. I think we can consider these low priority.
- a configured program in the .paar no longer exists on disk (FWD can't configure this 'leftover' and will generate a 404 instead of 500 error)
- some other small issues related to STOP/ERROR - I'll try to fix these with #3310 work.

The first two can be 'postponed'.

For WebHandler:

- FwdHandler.cls: do we add web-context handle support? Is used in this line: addJson(jsonRsp, 'parsed', jsonParser:Parse(web-context)). in FwdHandler.cls
- WebResponseWriter: we need tests for all methods... currently the implementation 'guesses' how it should work, but it's weird because you can call write(memptr) and the write(character). Which has precedence? How do you build the body actually?
- RemoteWebRequest: this class translates a Servlet HTTP request to 4GL (currently this is read-only; I don't know if 4GL can alter the cookie/headers in the request)
  - contentMd5, some headers-related APIs, defaultCookieDomain, defaultCookiePath, pathParameterNames, uriTemplate, some cookie-related APIs,
  - all the setters (for cookie, header, etc) are marked no-op; we need tests to see if these actually are no-op or do anything (like perform some validation or actually alter the field).
- implementation for net.Uri, net.http.Cookie, web.WebResponse and all their super-classes

Here, the work should start with the Uri, Cookie and WebResponse implementations, and RemoteWebRequest. After that, WebResponseWriter (but again, we need tests).

In total, if there are proper tests, I don't think this work will take more than ~2 days. Here I exclude the web-context system handle which ... is a black box at this time.

#103 - 04/27/2020 12:56 PM - Constantin Asofiei

I've configured the testcases project to be able to run the appsrv and rest tests:

- the appserver is state-free. If you want stateless, change the directory and run the tests individually. Look for sessionFree (set it to false) and state-free (set it to stateless) in directory.xml.
- FwdHandler.cls needs to exist in the root project folder
- Added appsrv/runall.p to run appsrv related tests
- configured REST and WebHandler - SoapUI can be used to run these in FWD, if you set the proper endpoint (https://localhost:7443) and path ("rest" for REST or "web" for WebHandler)
- file-cvt-list.txt.rest_and_appserver contains the list of files to convert, to be able to run the REST and appserver tests. Rename this to file-cvt-list.txt and run ant deploy. This contains more files than necessary, it can be stripped down. It takes ~20 minutes to convert everything.

Another issue in the testcases is a test.p program which collides with InternalEntry.Type enum (used by LegacySignature). I haven't fixed this.

To execute the runall.p suite, go to deploy/client-chui and run ./client-terminal.sh client:cmd-line-option:startup-procedure=appsrv/runall.p.

#104 - 04/27/2020 01:00 PM - Constantin Asofiei

I've edited #3855-102 with more details about the work required.

#105 - 04/27/2020 01:32 PM - Marian Edu

Constantin Asofiei wrote:

> I've edited #3855-102 with more details about the work required.

Constantin, we're working on URI and rest of classes in Net package, for Web I do not see what tests we can write that we could run from a regular client - all those need some web 'context', this is much like the old wrap-cgi Webspeed interface. Basically the WebRequest works with whatever you receive in the servlet and can pass down to the 4gl/fwd side, those are mainly CGI variables and data sent from the client for POST, PUT.

#106 - 04/27/2020 01:47 PM - Constantin Asofiei

Marian Edu wrote:

> Constantin Asofiei wrote:
>
> > I've edited #3855-102 with more details about the work required.
>
> Constantin, we're working on URI and rest of classes in Net package, for Web I do not see what tests we can write that we could run from a regular client - all those need some web 'context', this is much like the old wrap-cgi Webspeed interface. Basically the WebRequest works with whatever you receive in the servlet and can pass down to the 4gl/fwd side, those are mainly CGI variables and data sent from the client for POST, PUT.

I don't mean to test the web-context system handle. I refer to the properties as defined in the OpenEdge.Web.IWebRequest interface (especially those mentioned in #3855-102). RemoteWebRequest is a FWD-specific implementation of this interface, which relies on the Servlet request to get some data; but not all APIs are implemented yet (I think priority should have the missing parts for cookies and headers). Also, I think SoapUI can be configured to send cookies and different headers.

For WebResponseWriter the tests should cover the write APIs (except the WebStream property); I'm interested in what happens if you mixed these, or call flush before close, or before write, multiple open calls, and such. And also what happens if these APIs are mixed with WebResponse.

#108 - 06/08/2020 07:02 AM - Constantin Asofiei

For this what I'm missing are tests with extent arguments (input/output) and tables with extent fields.

#109 - 06/08/2020 11:18 AM - Marian Edu

Constantin Asofiei wrote:

> For this what I'm missing are tests with extent arguments (input/output) and tables with extent fields.

We can add those of course but for REST is all JSON, extents are sent as JSON arrays regardless if fixed or indeterminate size. For output those are sent as an array as well, nothing special about it.

#110 - 06/08/2020 11:25 AM - Constantin Asofiei

Marian Edu wrote:

> Constantin Asofiei wrote:
>
> > For this what I'm missing are tests with extent arguments (input/output) and tables with extent fields.
>
> We can add those of course but for REST is all JSON, extents are sent as JSON arrays regardless if fixed or indeterminate size. For output those are sent as an array as well, nothing special about it.

Yes, but I do need them to validate my changes.

#111 - 06/20/2020 11:55 AM - Greg Shah

- % Done changed from 0 to 90

Task branch 4231b has been merged to trunk as revision 11347.

#112 - 06/23/2020 02:02 PM - Marian Edu

Greg Shah wrote:

> Task branch 4231b has been merged to trunk as revision 11347.

Constantin, do you still need those extra tests for REST? I'm having troubles getting the deployment on PASOE to work but you were right, there are quite a few cases that are not covered in that service definition - mappings for headers, cookies, server context...

#113 - 06/23/2020 02:41 PM - Constantin Asofiei

Marian Edu wrote:

> Constantin, do you still need those extra tests for REST? I'm having troubles getting the deployment on PASOE to work but you were right, there are quite a few cases that are not covered in that service definition - mappings for headers, cookies, server context...

Yes, please add these, and the ones in #3855-108.

#114 - 07/25/2020 05:13 PM - Constantin Asofiei

Marian, looks like INPUT-OUTPUT was missing, too. Please add a couple of tests for these, too - I just want to make sure we got the matching/mapping right, for the conversion and runtime.

#115 - 07/25/2020 05:14 PM - Constantin Asofiei

And also double-check if raw, com-handle, object, rowid, recid are supported types, for both SOAP and REST. We found that at least HANDLE is supported.

#116 - 09/10/2020 05:58 AM - Constantin Asofiei

Runtime for multiple .paar support for legacy REST services is in 3821c rev 11512

#117 - 12/11/2020 09:56 AM - Greg Shah

> > Constantin, do you still need those extra tests for REST? I'm having troubles getting the deployment on PASOE to work but you were right, there are quite a few cases that are not covered in that service definition - mappings for headers, cookies, server context...
>
> Yes, please add these, and the ones in #3855-108.
>
> Marian, looks like INPUT-OUTPUT was missing, too. Please add a couple of tests for these, too - I just want to make sure we got the matching/mapping right, for the conversion and runtime.
>
> And also double-check if raw, com-handle, object, rowid, recid are supported types, for both SOAP and REST. We found that at least HANDLE is supported.

Did any of these tests get written?

#118 - 12/11/2020 09:59 AM - Marian Edu

Greg Shah wrote:

> Did any of these tests get written?

Some tests were added and updated the soapui results, I think only for PASOE not classical though. If anything missing please let me know, might have just fallen out of radar as we've put the focus on OO implementation :(

#119 - 10/09/2023 02:42 PM - Greg Shah

What parts of #3855-102 remain to be completed?

Where are we with the testcases?

#120 - 10/10/2023 02:28 AM - Marian Edu

Greg Shah wrote:

> What parts of #3855-102 remain to be completed?
>
> Where are we with the testcases?

I think we're missing tests for input-output parameters, we can update the supporting api to include all modes for passing parameters and update the SoapUI tests. The 4GL unit tests (calling in from the 4GL client) were not migrated to ABLUnit as far as I can see, we need to make them somehow configurable in order to be able to connect to the appsrv hosting the api. I will see if we can still somehow install the 'classical appsrv', this is no longer available in OE12.2 so we've let with PASOE now :(

There is quite a bit of setup needed for this so I will probably have to deal with it myself, is this one a higher priority compared with what was left to implement for OO support?

#121 - 10/10/2023 08:33 AM - Greg Shah

We have customer test environments that are 11.x so we can still access classic appserver for testing. Would that help?

> is this one a higher priority compared with what was left to implement for OO support?

No, the OO support is higher priority.