Subject: : JAmiga
Topic: : ARexxability - but how?
ARexxability - but how?
Author: : jaokim
Date: : 2014/8/16 11:37:26
URL:

I've always had the aim to add ARexx support for JAmiga. I however haven't decided on how it should look. How would one want to call Java code from an ARexx script?

I've been looking at the twitter4j API, and how that could be utilised from ARexx.
For instance how should this Java code be written in ARexx?

```
01   import twitter4j.TwitterFactory;
02   import twitter4j.Twitter;
03   import twitter4j.Status;
04   import java.util.List;
   ...
10   Twitter twitter = TwitterFactory.getSingleton();
11   List<Status> statuses = twitter.getHomeTimeline();
12   System.out.println("Showing home timeline.");
13   for (Status status : statuses) {
14       System.out.println(status.getUser().getName() + ":" +
15                          status.getText());
16   }
```

My initial aim was to not require any Java code to be written to utilise any arbitrary Java JAR. But this might prove too difficult. The main aim should be an easy to use approach.

One option is to expose the JNI layer using ADDLIB("jni"), creating something like:

```
01  CALL ADDLIB("jamiga", -20)    /* expose JNI layer*/
02  factory = GetStaticFieldID("twitter4j.TwitterFactory")
03  twitter = CallObjectMethod(factory, "getSingleton")
04  say "Showing home timeline."
05  statuses = CallObjectMethod(twitter, "getHomeTimeline")
06  DO i=0 TO CallIntMethod(statuses, "size")
07     user = CallObjectMethod(statuses, "getUser")
08     name = CallObjectMethod(user, "getName")
09     text = CallObjectMethod(user, "getStatus")
10     say name ":" text
```

```
11  END
```

As you see, this gets very chatty and clunky. It is however extremely powerful, as it allows you do much anything. However, you need to know your JNI, and if you do, you're probably better off just writing it in Java. There are also lots of stuff not covered, for isntance exceptions. This can probably be a real nightmare to use.

So, what I would like to achieve is something more in the lines of:

```
 01  ADDRESS JAMIGA
02  CALL ADDLIB("jamiga", -20) /* add JAmiga functions, like JCALL */
03
04  twitter = JCALL("twitter4j.TwitterFactory", "getSingleton")
05  statuses = twitter.homeTimeline    /* translates to twitter.getHomeTimeline() */
06  DO i=0 TO statuses.size
07     say statuses.i.user.name ":" status.i.text
08  END
```

This is much more convenient to write, and even makes sense if you just know ARexx. It is however very likely a can of worms for me, implementation wise. For instance, in order to get the homeTimeline stem variable on row 05, the underlying code on row 04, woudl have to fully traverse the Twitter object structure, which does hold a lot of other methods, where some most likely would throw exceptions.

I guess I could add a few more helper functions, to export Java bean objects to stem variables, forming something like:

```
 01  ADDRESS JAMIGA
02  CALL ADDLIB("jamiga", -20)
03
04  twitter = JCALL("twitter4j.TwitterFactory", "getSingleton")
05  statuses = JBEAN(twitter, "getHomeTimeline")  /* makes the list of statuses as a stem var */
06  DO i=0 TO statuses.size
07     status = JBEAN(statuses, "get", i)      /* gets the i:th item */
08     say status.user.name ":" status.text
09  END
```

Now, I haven't done much ARexx host programming so I don't know all ways how callbacks from ARexx to C is actually made, but I don't think that accessing stem variables creates a callback, right? AFAIU, a stem variable is just a variable with dots in the name...?

Does anyone have any input to give, on how you would like to call Java from ARexx?