Adventures in Groovy – Part 49: Unable To Retrieve Variable Deployed In Application

Have you ever gotten an error that reads, “Unable to retrieve variable *variable name* deployed in application?” When you look at the error, the variable name in the error message doesn’t exist and you aren’t trying to retrieve such a variable? If you spent enough time, or your script was small enough, you might have recognized that the variable name that it can’t find is a Groovy variable.

The Issue

The following script is an example, and likely a piece of a larger script. All this does is create a string variable named uda and sets it equal to Locked. It is then used in a metadata query that returns a list of the name of all the members with a UDA of Locked.

```
String uda = "Locked"
List<String> lockedAccts = operation.application.getDimension('Account',rule.cube).getMembersWithUda("${uda}",false,rule.cube)*.name
```

When this is executed, it returns an error that reads *Unable to retrieve variable uda deployed in application*.

Why is the script looking for a substitution variable of uda? You might be thinking, I know I have put Groovy variables in strings and this hasn’t happened, so what the heck? Can I now use Groovy variables?

Struggling through this, once again the Oracle development team, and you know who you are, helped me figure this out. And, here is the reason this is happening, how to fix it, or use it to your advantage.

The Why

This is actually a bug of sorts. As Oracle has added functionality, this cropped up as an unintended issue. Originally, anything in squiggly brackets is interpreted as a variable, like in an Essbase calculation. This was built to interpret this. When Groovy was added, and using the same syntax to reference Groovy variables, not substitution variables, the interpreter of the Groovy script kind of tripped over itself. The way it is built is to look at these in the native Essbase variable ONLY IF there are no run time prompts identified. If there are RTPs in the script, the interpreter changes and interprets them as Groovy variables. Since so many clients use this for the unintended purpose of identifying substitution variables, Oracle can't "fix" it as it would completely hose a lot of people.

The Fix

The fix is simple, add run time prompts! Seriously, I have to add an RTP even though I don't need one? No, all you have to do is enter /*RTPS:*/ ANYWHERE in the script. I normally add it to the top and the script above would now look like this. When I add the RTP indicator, even with no run time prompts, the interpreter doesn't look for anything in squiggly brackets to variables, but the more likely expectation, a Groovy variable.

```
/*RTPS:*/
String uda = "Locked"
List<String>                lockedAccts                =
operation.application.getDimension('Account',rule.cube).getMem
```

```
bersWithUda("${uda}",false,rule.cube)*.name
```

When I run this script, I get a successful message.

That’s a Wrap

This post was a long time coming. I had somebody ask me about it and it made me realize that I haven’t shared this information. I make it a practice to always add /*RTPS:*/ to every Groovy script. If you want to get more information like this, become an industry leader, check out Groovy For Planning.