Communication Intra-Steps Using TheLocker

Appvance IQ’s architecture allows you to integrate different testing tools and technologies into one single orchestration framework. Each one of these tools and technologies works independently, leaving some gaps that won’t let them work as a team with other tools. One gap Appvance IQ fills is the external data-driven testing, achieved with the DPLs. The other gap is the actual communication between tools. That is achieved with a special DPL called TheLocker.

By default, all DPLs are read-only. That means you can use them to obtain dynamic data, but not to push data to them. When orchestrating, one test step (performed using one tool, let’s say soapUI) may generate data that can be used as input to the next step (say a verification on a web page using Selenium). To communicate these two steps, one solution will be to have one DPL that allows step 1 to push its results into a named “space” where step 2 can read it later on. This is what TheLocker does.

TheLocker metaphor is a simple one. Any tool or step can request a locker space, name it and store any kind of data in it. The subsequent steps are able to retrieve that data using the name of the space. An example may help to understand the concept better.

The first step, as with all the DPLs, is to create an instance of TheLocker:

```
<DataSources><dpl name="myLocker" type="TheLocker"/></DataSources>
```

As special as TheLocker is, it behaves a little differently than regular DPLs. One difference is that a normal DPL instance shares all its data with all the user threads, thus one user changing its data-position will change the position for all the other threads. TheLocker has one copy of the instance for each user, so each one may add different values to the same space and this will not override their partner's data. Another difference is TheLocker always has one instance, no matter how many you define, it will be the same.

Now, to push data. To have data pushed into TheLocker, the testing tool in turn needs to offer a scripting facility. In the example mentioned above, soapUI is the tool that may push data into a namespace and this is done using the groovy scripting facility of soapUI. This is done by creating a Groovy step in soapUI and issuing some code similar to this:

```
def myLocker = new com.Appvance.tool.dpl.provided.TheLocker();
myLocker.put("paht1",firstXPath)
myLocker.put("path2",seconfXPath)
```

Let’s see the selenese commands inside our selenese file, to see how it will pass the data:

```
<tr><td>clickAndWait</td><td> paht1</td><td></td></tr><tr><td>clickAndWait</td><td> paht2</td><t
```

When running Selenese, the passed data from TheLocker will match the name of the pairs (path1 and path2) and replace those in the Selenese file with the data soapUI placed in TheLocker.

Note this is highly intuitive. Appvance IQ intelligently matches up and attends to the intention of the user to place the values in the correct place.

But the user may have even greater control of TheLocker in a scripting environment. For instance, if the receiving step is a Jython step, you may not need to pass the DPL values as above, you can simply request TheLocker instance and get data from it, like this:

```
from com.Appvance.tool.dpl.provided import TheLocker
…
myLocker = TheLocker()
lockerValues = myLocker .getData();
```

Last updated: June 02, 2023

If you have any feedback on this topic, please contact us at documentation@appvance.ai

---

Revision #11
Created 2026-07-31 15:09:37 UTC by Admin
Updated 2026-11-17 22:06:48 UTC by AI_Mananger