Improving Memory Usage

SoapUI is rather memory intensive, especially with default settings for logging, and so on. If you are running more lengthy functional tests or load tests you might well bump into an _OutOfMemory_ error. Fortunately, there are several things that can be done to minimize the risk for running into this error.

**Adjust Memory Settings**

This doesn't really solve the underlying problem, but if you have large WSDLs, requests, attachments, and so on the default settings in `bin\soapui.bat`, `bin\soapui.sh`, `bin\vmoptions` for Windows or `Contents/vmoptions.txt` for Mac won't be sufficient.

Open the corresponding file in a text editor and change the line

```
set JAVA_OPTS=%JAVA_OPTS% -Xms128m -Xmx256m -Dsoapui.properties=soapui.properties
```

to use higher values for the max allocated size, for example:

```
set JAVA_OPTS=%JAVA_OPTS% -Xms128m -Xmx768m -Dsoapui.properties=soapui.properties
```

The exact value is hard to say, it depends of course on your system setup, other applications running, and so on. You can set it to half the amount of available memory, for example on a machine with 2 GB of ram that would be 1024. Remember that if you are running a 32-bit operating system (for example Windows XP), then you wont be able to allocate more than approximately 1.5Gb, so there is no use in setting the value higher then that.

If you run into PermGen errors (for example if you are using a lot of groovy scripts), then you might need to adjust that setting as well, add a `-XX:MaxPermSize` setting to the above line:

```
set JAVA_OPTS=%JAVA_OPTS% -Xms128m -Xmx768m -XX:MaxPermSize=128m -Dsoapui.properties=soapui.properties
```

**TestCase Memory Usage**

One common memory-hog is a long-running functional TestCase, usually data-driven and thus running through thousands of requests. By default SoapUI automatically saves the entire request-response message exchange for each request so it can be viewed by double-clicking the corresponding entry in the TestCase log. Obviously this will fill up memory over time no matter how much you have allocated, but fortunately there are ways to discard old results from memory if not needed.

**TestCase Log**
The TestCase Log Options available via the toolbar on top of the Transaction Log:

The Log Options settings define aspects of the transaction log:

Log Options
Set options for the run log below

- Disable logging: Disables the log, if checked nothing will be written in the window.
- Errors Only: Logs only TestStep errors in the log
- Follow: Follow log content

These log options control what is visible in the log. This affects memory to a certain extent (for display purposes), but something not shown in the log can still affect memory. You might still want to access events or results from a groovy script or for reports, in which case you would still need access to all results, these would then still be in memory.

Note: Removing things from the log does not necessarily mean they are removed from memory.

TestCase Options

TestCase options are available from the toolbar:

There are two settings in the TestCase Options dialog that affect memory usage.

- Discard OK Results: Discards successful TestStep results to preserve memory
- Max Results: 0
The options at play are:

1. **Discard OK Results**: Selecting this will discard the content of any successful TestStep Result that would otherwise have been viewable by double-clicking the corresponding result in the TestCase Log.

2. **Max Results**: Controls how many TestStep Results are kept in memory in total, regardless of their state. Setting this to 0 will keep all in memory but allow LoadTests to discard them if possible. If you don't want LoadTests to discard them, set this to a very high value.

Be aware that if a TestStep Result is discarded (either because it is ok and should be discarded, or is beyond the limit set in the second option above), it will still be held in memory if it is visible in the log. Thus the Log Options and TestCase Options need to harmonize so that they don't hold on to old Results when they shouldn't.

When running from the command-line there is no Log, so in that scenario only the TestCase Options are applicable.

**Note**: Always select the **Discard OK Results** and set the **Max Results** to a reasonable value when running tests in production, depending on your specific requirements.