

## HeapLogger

Heaplogger version 1.

This lib is a heap investigation tool. It is designed to help track down memory leaks, to find out what is going on when you make an OS call, or? well that is about it really. It gives you the ability to log all the memory allocations and dump the heap into a log file. You can also make annotations so that you can tie the allocations back to your code. The functions are described in the header.

```
/**This function will go at the start of your app. It will give a config line line like
 * This is the Heap Logger Dump.
 * Old heap contains 000000 bytes in 000000 cells.
 * If there are any cells already allocated, the old heap should be switched back
 * in with User::SwitchHeap() before these cells are freed.
 *
 * @param aFileName is the name of the log file created
 * @return the old RHeap.
 */
IMPORT_C static RHeap* HeapLogger::UseHeapLogger(const TDesC& aFileName);

/**This sets what will actually be logged. Defaults to ELogNothing
 * @param aMode is an ored combination of THeapLoggerMode
 */
IMPORT_C static void HeapLogger::SetMode(TInt aMode);

/**When content is logged (on Free or a heap dump) this is the max
 * length of the buffer to be logged.
 * @param aMaxLen the max length to log.*/
IMPORT_C static void HeapLogger::SetMaxContentLength(TInt aMaxLen);

/**This dumps the current heap into the log file.*/
IMPORT_C static void HeapLogger::DumpHeap();

/**This allows you to put tags in the log file so that you can tie the
 * allocations to sections of code.
 * @param aLogStr is a tag for your code to annotate the log.
 * @param this is the file, normally given by "__FILE__"
 * @param this is the lin number in the source file. Given by "__LINE__"*/
IMPORT_C static void HeapLogger::Log(const TDesC8& aLogStr, const char* aFile,
TInt aLine);
```

To use the heaplogger, just include the header file and static lib in the mmp file, and then call the functions.

You can download a hello world type example at

<http://homepage.virgin.net/mr.nigel.brown/examples/Lumberjack1.zip>. Lumberjack just uses the logging tool to log some random things. The libs and the header file are included. The libs were compiled to Series 60 3rd MR, but should work on other Symbian OS V.9 platforms.

This works on the emulator, but there are better tools to use so this is only really useful on a target device. Also, it makes the application run so incredibly slowly when logging, that it is not useful for long term use. Just one off investigations. The log files get very big.

It all works by replacing the heap in the process with a custom one. I have used it a couple of times for real work, but it is not that well tested. It is here mostly for fun. I did have some issues with using this and target debugging on carbide, but they just went away.

Any comments, post them here.