30th August 2025

By: John Goacher

## Developer Notes

These notes are intended as a brief guide to anyone wishing to to access the lower level BSOL api from software. It does not provide a definition of the api but provides information about where to look in the code to see how it is used.

BSOL was not originally written with the expectation that the code would be publicly released, so is not as well commented as it could be. It was originally developed as a client/server application, with all double dummy calculations performed by a CGI module on the server. The standalone version now carries out all such operations in user's browser. The server is only required in order to host the files. However, the comments in the code still refer to the server and to CGI modules in some places – ignore this !

The double dummy calculations use Bo Haglund's well known double dummy solver. I have compiled this into a WebAssembly module using the Emscripten SDK for Windows, which is freely available. The source files and the build file for the WebAssembly (wasm) module are in the directory WebAssembly_src. The output files from the build are dds.wasm and a module dds.js which provides a javascript interface to it. If building this module note that the build will generate a number of warnings, mainly about non-compliance with ISO C++11. These can be safely ignored.

BSOL creates a number of Web Workers to interact with the wasm module. Using multiple Web Workers allows operations to run in parallel (assuming multiple CPUs/threads), and means that operations can run in the background without affecting UI responsiveness. The Web workers run a module callds.js. BSOL sends messages to the workers using the postMessage function and attaches a function "listener" to the web worker in order to receive responses.

callds.js uses the statement "importScripts("dds.js");" to import the javascript interface to the wasm module and uses the Module.cwrap function to generate a reference to the "handleDDSRequest" function in that module.

I do not have documentation of the api but by examining the code in callds.js it is possible to see how handleDDSRequest is invoked. Refer to the file DDummy.cpp in the WebAssembly_src directory to see how the requests are processed. DDummy.cpp makes calls to Bo Haglund's dds.cpp module and documentation for these functions is on his website. Note that the last 3 parameters to handleDDSRequest are a hangover from the client/server implementation of BSOL cand can safely just be set to null.