An Interactive Development Environment for CHDK uBasic Scripts

UBDebug is a PC-based interactive environment for developing for CHDK ubasic scripts. It lets you step through scripts, inspecting and setting variables. Here's what it looks like when it's running:

Click this to browse directories and select a script to run

You can alter the values of these

Click this to edit the current script

You can 'drag and drop' .bas files here

You can set the values returned by these functions …

This log is used to report errors as well as output from the current script

… and the value of any property

Use these buttons to execute the script

Inspecting and setting Parameters, Variables, Functions and Properties

Clicking **Set** will set the value of **i** to 5

Clicking **Set** will set the value returned by **get_usb_power** to 12

Clicking **Set** will set the value of property 206 to 4

This panel lets you alter the values of variables and the value returned by certain CHDK functions. Note that the list of parameters is initialised when a script is *loaded* but variables do not appear in the list until the script has been *started* and a line setting the variable executed. The function list is not populated until a script is *started*.

Double clicking on a variable line opens up a text field above the list in which a new value can be entered. Doubling clicking on one of the functions does the same thing. Property values can be set and altered too – clicking on the **Set** button will either update a value or set a new one if the property is not already in the list.

Running or Stepping a Script
Once a script is loaded, clicking the **Start** button starts interactive debugging. The **Start** button is renamed **Step** and the **Run** and **Stop** buttons are enabled. The script itself is disabled, so it's no longer possible to edit it or set a breakpoint. If the **Step** button is then clicked, the current highlighted script line will be executed and the debugger will wait. If **Run** is clicked the rest of the script will be executed (unless and breakpoint has been set, in which case execution will stop at that line). If **Stop** is clicked, or the script ends, then the **Run** and **Stop** buttons are disabled again, **Step** is renamed **Start** ready for the script to be run again or another script to be loaded and the script is re-enabled.

Editing a Script
While a script is running you can't change it, but before you start it and after you stop it you can edit it by clicking the edit button. A simple editor panel appears which lets you make changes and then update the debugger's script. You can also save the edited script to a file.

Breakpoints
The current version only supports one breakpoint at a time. Before a script is run you can set a breakpoint by double-clicking on a script line. The script line then has the prefix '**Break –**'. Double-clicking the same line will clear the breakpoint, while double-clicking another line will set the breakpoint there instead. If one is set then when the **Run** button is clicked the script will run up to the breakpoint. You can see a breakpoint (set on line 18) in the picture on the previous page.

Note that once the script is running the script list is disabled and hence a breakpoint can neither be set nor cleared until the script is stopped or ends.

The Current Package
UBDebug is mostly written in java, but it uses a slightly modified version of Adam Dunkels' **ubasic** interpreter that's part of CHDK. The interpreter, together with the java "native methods" which interface the debugger to the interpreter, is written in C and has been compiled as a DLL (for Windows) and a DYLIB (for Mac OSX). The ZIP file contains the following files:

UBDB.jar	this contains the five java classes which make up the debugger
ubengine.dll	the native methods and ubasic interpreter (Windows version)
libubengine.jnilib	the native methods and ubasic interpreter (Mac OSX version)
test.bas	a simple test script

Simply unzip these files into any convenient directory and double-click the jar file to start the debugger.

Mac OSX users may prefer to download the App version (packaged as a disk image).

Note that the debugger stores the current script and directory names in a properties file, so the next time you run it the script last loaded will be loaded again.

Comments, suggestions and bug reports are welcome.

Dave Mitchell (dave@zenoshrdlu.com)