

A Guide to the CS 110 Library

The point of Lab #0 is to get you used to the programming environment, not to make you experts on dealing with the PalmOS API (application programmers' interface). In order to allow you to write programs for the Palm without knowing these details, we have provided you with a shell to deal with initialization / cleanup and a library of routines to work with that shell.

As you will notice from looking at the "src" folder that comes with the lab 0 distribution, the shell and library are divided into quite a few files. Here's the lowdown on what's there.

- [CS110Shell.c/h/.rsrc/Rsc.h](#) - This contains the code and resources for the CS 110 shell. This is mostly just the starter code given by Palm with a few modifications. This is C with a little bit of assembly.
- [CS110Lib.c/h](#) - This is a library of routines to act on the CS 110 shell. The .h file is reprinted with this handout. **In order to write Lab #0, the only thing you need to understand is this .h.** This file is written as a C / assembly mix (mostly assembly).
- [A5Utils.c/h](#) - Routines to save/restore the register A5. You will find out why this is important later in the quarter. These routines are called by CS110Shell.c and CS110Lib.c. This file is written in assembly.
- [setjmp.palm.c/h](#) - An implementation of the ANSI C setjmp routine used by the CS 110 shell and library. This is really neat. Implemented in assembly.
- [HostControl.h](#) - This is a small header file used by the CS 110 shell to tell whether it's running under the emulator or not. You need not concern yourself with it.

You do not need to look at any of these files except CS110Lib.h. However, if you're interested, feel free to take a look at them. They are all well documented with comments—however we will not go over the topics used to implement them until later in the quarter.

```

/* File: CS110Lib.h
-----
This file contains a few routines that can be used to write some simple
assembly language routines for the Palm.  These routines are meant for use
with the CS 110 Starter project (CS110Shell.c and CS110Shell.rsrc)--they
probably aren't very useful otherwise.

This interface is based on the old CS110 shell for Cöder.  Many thanks to
previous TAs and instructors for their work.  Note that some functions from the
original interface weren't needed for our purposes and weren't written.

Note: none of these routines trash any registers (unless they use a register
for output).

by Douglas Anderson (dianders@cs.stanford.edu)
for CS 110, Spring 1999

Modification History:
    03/25/99 - Initial revision.
*/
#ifndef _cs110lib_h
#define _cs110lib_h

/* Function: SetTitle()
-----
This function sets the title of the console to the string indicated by A0.

Input:
    A0.L - Input string.  Should point to first character in null-terminated string.
    NOTE: the string passed in is NOT copied--it is actually stored by the
    OS for future use.  Strings that have been allocated with DC are fine
    to use, since those are never freed and shouldn't ever change.
*/

asm SetTitle();

/* Function: ClearScreen()
-----
Clears the screen.  Basically, just deletes everything in the field.
*/

asm ClearScreen();

/* Function: NewLine()
-----
Moves to the next line.  This is equivalent to writing a '\n' character with
WriteChar().
*/

asm NewLine();

/* Function: WriteChar()
-----
Writes the character stored in the lowest byte of D0 to the console.

Input:
    D0.B - Input character.  The lowest byte is the byte written.
*/

asm WriteChar();

```

```

/* Function: WriteString()
-----
Writes the C string (null terminated string) stored at the location
indicated by A0.

Input:
  A0.L - Input string.  Should point to first character in null-terminated string.
*/

asm WriteString();

/* Function: WriteNum()
-----
Writes the number stored in D0 to the console.

Known bugs:
- There appears to be a bug with the Palm libraries which makes it fail when
  it is supposed to convert -2147483648.  As far as I can tell, all other
  numbers work just fine.

Input:
  D0.L - Input number.
*/

asm WriteNum();

/* Function:ReadChar()
-----
Reads a character and returns it in the lowest byte of D0.  The high bytes of D0
are set to 0.  The character is not passed onto the system and is not printed to
the console.

Output:
  D0.L - The low byte of D0 gets the character.  The upper bytes are cleared.
*/

asm ReadChar();

/* Function: MouseClick()
-----
Waits for a mouse click and returns when it gets it.  Very simple form of flow
control.  The mouse click is not passed onto the system.
*/

asm MouseClick();

/* Function: Beep()
-----
Does a simple little beep.
*/

asm Beep();

#endif // _cs110lib_h

```