An Introduction to PEBL: The Psychology Experiment Building Language

Shane T. Mueller, Ph.D.
Indiana University Dept. of Psychology
1101 East 10th Street
Bloomington, IN 48109
smueller@obereed.net

Abstract

PEBL (Psychology Experiment Building Language) is a new programming language developed especially for building psychology experiments involving human participants interacting with a computer. PEBL source code and compiled binary executables are available via the internet at http://pebl.sf.net.

Technical Details

PEBL is an interpreter/compiler written in C++, and uses a parser designed with the flex and yacc compiler design tools (Levine et al., 1992). PEBL makes heavy use of data structures from the C++ Standard Template Library, and uses a counted-reference pointer system to enable automatic memory management. It is designed with several principles in mind: (1) The source code for important scientific tools should be freely available and modifiable; PEBL is Free software, allowing experimenters complete control over the system. (2) PEBL is designed to be cross-platform, allowing a single program to run on whatever computer system the interpreter is compiled for, and to allow today's experiments to be used on tomorrow's computer systems. (3) PEBL is designed to be used by psychologists, who are rarely skilled programmers. Consequently, aspects of the language are designed to enable common experiment idioms to be used easily, while having a forgiving syntax that encourages the creation of maintainable and readable programs.

PEBL is Free Software

PEBL is designed with the hope to providing a lingua franca within which psychology experiments can be created and exchanged freely, to be used without the need to license or purchase other software. The source code for PEBL is licensed under the GNU General Public License (Stallman & Moglen, 1989), which allows others to use and modify the source code as long as any changes are made available to others. The license covers distribution of the executable binaries and source code of the PEBL system, and does not include user-created experiments that can be run with PEBL. Psychologists who design experiments using PEBL are encouraged to redistribute their PEBL programs freely, to foster easier exchange of scientific information.

PEBL is Cross-Platform

PEBL was designed to allow it to be easily ported to different computing platforms. Together with the fact that Current versions are written using the Simple DirectMedia Layer Library (SDL; Lantiga, 2004) an open source cross-platform game library. The use of this library enables multiple platforms to be targeted easily, but PEBL
can be ported directly to other platforms, potentially improving performance by taking advantage of platform-specific idiosyncrasies. Currently, compiled versions exist for recent versions of Linux and Microsoft Windows using the SDL, and a version for Macintosh OSX will be possible with very little modification.

**PEBL is for Psychologists.**

PEBL is designed to be used by psychologists who have modest programming skills. Its design takes the challenges faced by the novice programmer to heart, and attempts to help programmers avoid common mistakes made in other languages. The syntax is relatively simple, yet it attempts to enforce a readable and maintainable programming style through several mechanisms. Additionally, PEBL provides many useful functions tailored to experimental design. Complex stimuli are managed through PEBL's object system, which attempts to hide complexity from the user whenever possible.

**Syntax.** The syntax of PEBL is fairly simple. A sample PEBL function ("Add") is shown below:

```plaintext
1 define Add (number1, number2)
2 {
3   # This function adds two
4   # numbers together
5   sum <- number1 + number2
6   Print(number1 + " plus " + number2 + " = " + sum)
7   return sum
8 }
```

When this function is invoked with two numbers as arguments (e.g., 3 and 5), it calculates the sum of two numbers, prints out the message "3 + 5 = 8", and then returns the sum to the calling function. In the PEBL syntax, blocks of code are set apart using curly brackets, and newline characters signify the end to complete expressions. All functions begin with a capital letter (cf. Lines 1 and 6), and all variables begin with a small letter (e.g., Line 5); this naming convention enables novice users to more easily identify which type of object they are dealing with. Comments (Lines 3 and 4) can be added by preceding them with a #, a typical comment character in many languages. Variables do not need to be defined or declared prior to use (Line 5), nor do they have an explicit type (all data is stored as a "variant"), removing a substantial source of error for programmers of all abilities. The '+' operator serves to add numbers together (Line 5), as well as to concatenate strings (Lines 6-7). PEBL uses the "<->" combination as an assignment symbol (Line 5) to avoid ambiguities arising in the meaning of the traditional assignment symbol "="; which is often confused with a test for equality. The function returns a value specified by the optional return keyword (Line 8), which can only occur at the end of the function, promoting easier debugging because each function has only one exit point. To further promote the ability to trace execution paths through code, there are no 'go to' statements in the PEBL syntax (Dijkstra, 1968), and no means of breaking out of loops aside from their invoking tests.

**Function Library.** PEBL provides more than 160 built-in functions, many of which implement commonly-used experimental idioms. For example, PEBL has a function GetNIMHDemographics(), which will gather demographic information required by NIMH from participants and write it to a text file. Several functions enable experimental design, such as DesignLatinSquare() and DesignFullCounterbalance(), and others enable easy stimulus selection and randomization: Shuffle(), SampleN(), SampleNWithReplacement(). Random numbers can be generated according to a variety of distributions with functions such...
as RandomDiscrete, RandomNormal, and RandomExponential. PEBL also includes a full complement of mathematical functions.

Object System. Experimental stimuli and other visual and auditory objects are handled via PEBL’s object system. These include images (PEBL can display most common computer bitmap image formats directly, such as .bmp, .gif, .jpg and .png), fonts, colors, text labels (single lines of text), text boxes (boxes of text that wrap and are optionally editable by the participant), file streams, and audio streams. These are created and modified with PEBL functions.

Examples of simple and complex experiments are available in the PEBL source tree, under the demo subdirectory. Complete documentation of the syntax and functions is available in the PEBL manual.

References:


