

A Powerful Unit Conversion Macro  
WordPerfect Magazine  
By Roger Gagon; Macro by Robert Padgug

Let's face it – the measuring system we use here in the United States is pretty eclectic. We order 16-ounce steaks but measure the fat content in grams. Our cars have 10-gallon gas tanks and two-liter engines. We take 3" x 5" photos using 35-millimeter film.

This piecemeal transition into metrics is spurring many companies and industries into providing information in both U.S. (Imperial) measurements and metric equivalents. Fortunately, Robert Padgug's macro entry in this year's Best Shot contest makes short work of metric conversions as well as many other conversions. This article shows you how to use the included UNITCONV.WPM macro to convert hundreds of different measurements. You'll also learn about an advanced programming concept called "arrays" and how to use them in the macros you create.

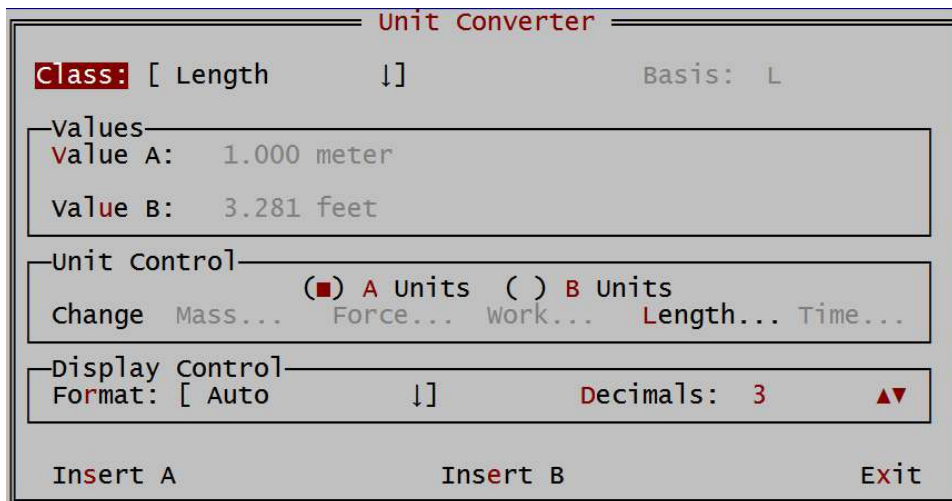


Figure 1. The Unit Converter dialog box lets you quickly and easily convert hundreds of measurements.

### Using the Macro

When you're converting volumes (such as cubic centimeters), the cubed symbol (<sup>3</sup>) displays as a box when you're in Text Mode. For this reason, you'll generally want to be in Graphics or Page Mode when you use the macro. To change modes, press Screen (Ctrl+F3), (3) Graphics Mode.

You can run the macro from any document screen. Press Play Macro (Alt+F10), type *unitconv*, press (Enter) and the Unit Converter dialog box shown in Figure 1 above appears on your screen. You'll first press (C) Class and choose one of the 12 available conversion types: Acceleration, Area, Density, Force, Length, Mass, Power, Pressure, Time, Velocity, Volume or Work. For now, highlight Area and press (Enter). Notice that the Basis text box now displays an L2 symbol, indicating "length squared."

The two values displayed in the Values group box are equal. If you chose Area, Value A should read 1.000 meter and Value B should show its equivalent in feet: 10.764 feet. Let's say you want to convert 35 square meters to the equivalent in square feet. Press (V) Value A, type 35 and press (Enter). The Value B text box now shows 376.737 feet.

What if you want to convert square kilometers to square miles? You can do this using the Unit Control group box. First, make sure A Units is selected, then press (L) Length to display the Change A Length Units dialog box (see Figure 2 below). Since Unit is already set to meters, just change the Prefix to kilo. Press (P) Prefix, highlight kilo and press (Enter) twice. Notice that Value A in the Values group box has been changed to kilometers. To change Value B to square miles, press (B) B Units to change the selection in the Unit Control group box. Press (L) Length to display the Change B Length Units dialog box. Press (U) Units, highlight miles and press (Enter) twice. This time, Value B in the Values group box has been updated and should indicate the equivalent of Value A in square miles.

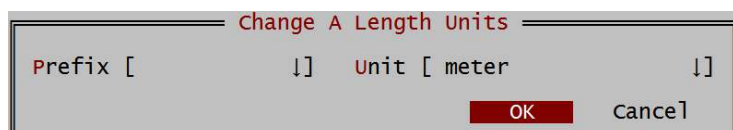


Figure 2. The Change A Length Units dialog box lets you choose a variety of length measures. If you're converting to metrics, you can assign over a dozen metric prefixes, such as nano-, micro-, milli-, centi- and deci-.

*Note: The macro does not discriminate when adding metric prefixes to units, even if the resulting combination isn't a legitimate word or measurement (e.g., picopound or megaweek). In other words, just because the macro lets you convert femtopoints to gigafeet doesn't mean these are valid measurements in the real world.*

You can change the decimal point to be as precise as you want. If you want to round the number to a single decimal point, press (D) Decimal, type 1 and press (Enter).

You can also change the format of the values from fixed to scientific. Press (R) Format and note the three options in the Format drop-down list – Scientific, Auto and Fixed. The Auto option generally displays your values in fixed format. If, however, the values ever get too large (greater than 10 digits to the left of the decimal) or too small (more decimal places are needed than are provided by the Decimal text box), then the format automatically changes to scientific. I'd recommend leaving the format set to Auto, but you can force it to always display one way or the other by choosing either Fixed or Scientific.

Finally, if you want to insert either value into your document, simply press (S) Insert A to insert the A value or press (E) Insert B to insert the B value.

## Understanding the Macro

UNITCONV.WPM is not a particularly easy macro to understand, so I'm going to spare you the laborious line-by-line exegesis. What may be beneficial, however, is to discuss how this macro uses arrays to create drop-down lists in a dialog box (see Figure 3 below). From the discussion, you'll learn how to add drop-down lists to macros of your own.

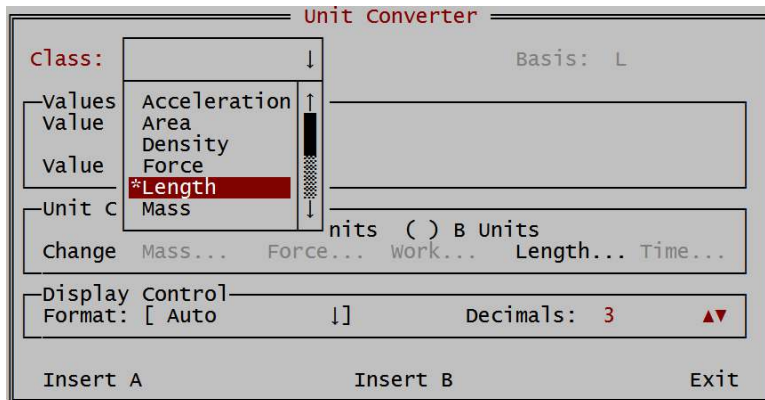


Figure 3. Variable arrays make it easy to create custom drop-down lists in dialog boxes, such as the Class drop-down list shown here.

First, you'll need to understand variables and arrays. You may already know that a variable is a unit of storage in the computer's memory – similar to a mailbox. Just like a mailbox, it has an address and a capacity. (WP6.x variables can hold approximately 256 bytes or characters.) Numbers, words, phrases and even some codes can be stored in variables, just as you can put several different things in a mailbox.

An *array* is a series of variables grouped under one name. To illustrate this idea, imagine an apartment complex named Hill Crest Apartments with a long row of mailboxes for the tenants. Each mailbox in the row or array is numbered sequentially beginning with the number 1. You could represent Mailbox #1 at Hill Crest Apartments as: *HillCrest[1]* and mailbox #2 as *HillCrest[2]* and so on. A macro with the code *HillCrest[4]:= "Package"* would assign the word *Package* to the fourth position of the HillCrest array. This method of assigning one variable works well for some purposes, but you can also assign an entire array at once.

### Using Arrays in Dialog Boxes

OK, so you've assigned your arrays, now how do you use them? Take a look at line 27 of the macro. This DLGCONTROL command uses a drop list as its control type (the first parameter). The second parameter on this line – Class – indicates the array name used in the drop list.

Whenever you use a CtrlDropList! or CtrlList! control type, you must always define the array before you can use it in the dialog box. In this case, the array was defined on line 4. So, when the dialog box displays (see Figure 1), if you choose the Class drop list, the macro displays the list of items in the Class array defined on line 4 (see Figure 3).

*Note: Notice that the list is now sorted in alphabetical order. The DLGCONTROL command automatically alphabetizes the array for you. If you don't want the list alphabetized, you can add the text +StyNoSort! immediately after the StyNoNumber! on line 27.*

The number of the option you choose in this drop list is stored in the initial "zero" position of the Class array – Class[0] or simply Class. (Remember, array numbers always begin with 1 – the zero position is reserved for the default number in the array.) If you chose Mass from the drop list in Figure 5, Class

would equal 1 since Mass is the first item in the array. Its location in the sorted drop list doesn't change its position in the array.

Arrays such as this are useful when you want to pair up lists of information. For example, compare the Class array on lines 4-5 with the ClassBasis array on line 6. The M corresponds with Mass, the F corresponds with Force and so on. If you chose Mass from the drop list in Figure 3, Class would equal 1. Therefore, *ClassBasis[Class]* (or *ClassBasis[1]*) would be M. Notice that this is exactly what the macro does on line 28 to update the class basis in the dialog box (compare Figures 1 and 3).