

## TECHNICAL MEMORANDUM

To: Microsoft Word Users/Developers

Project: General Information

From: Dave Ihnat

CC:

Date: 08 Oct 98

Last Modified:

22 Apr 03

Subject: A Quick Tutorial on Template and Form Creation  
and Use

---

### **1. INTRODUCTION**

Microsoft Word (MSWord) offers a rich variety of features for the use of templates. Microsoft Excel (MSEcel) also offers template support.<sup>1</sup> However, there isn't any simple, clearly-spelled out entry in the online documentation, and third-party manuals have proven to be spotty.

This memorandum is intended as a quick primer to explain the issues involved in creation of both simple and complex templates as forms in MSWord. It will also discuss the management of templates in both MSWord and MSEcel.

With respect to distribution, this document is copyright © 1998 by David M. Ihnat. Permission is given to distribute freely, and to use the information in this document for any purpose, commercial or private, provided the author's attribution and copyright notice remain intact.

### **2. OVERVIEW**

This memorandum is presented in several major sections; each discusses a different topic related to template creation and management.

### **3. BOILERPLATE VS TEMPLATE**

You will run into two common terms when dealing with this topic—*Boilerplate* and *Template*. Let's define them for our purposes.

#### **3.1 BOILERPLATE**

*Boilerplate* is commonly used to refer to a document that contains stock text, formatting, and graphics. You make use of a boilerplate document by first copying it, then modifying the copy to suit your current needs. This is easy to manage and set up, but has several disadvantages.

Microsoft Word (MSWord) provides no direct support for or protection of boilerplate source documents. A common problem with procedures that rely on using boilerplates is that people open and then forget to save the document as a new file. Instead, they directly modify the boilerplate, thereby destroying it for others.

---

<sup>1</sup> Actually, so does PowerPoint—but we've gotta draw the line somewhere on what we're going to try and discuss here.

---

This can be overcome by using operating system protections (e.g., set the file read-only, make the server sharing it protect the share from writes by unprivileged users, etc.) but all of these are more complex than just using Word templates—and complexity means mistakes are more likely to happen.

Frankly, once you master templates, there is no advantage to boilerplates over templates.

## 3.2 **TEMPLATE**

A *template* is directly supported by MSWord and MSEXcel. Word offers a rich set of management options that integrate your template into the types of documents offered in the *File/New* choices in such a way that they're indistinguishable from the ones distributed by Microsoft.

All that differentiates a template from a normal Word or Excel document is two attributes:

- It is saved as *filename.dot* instead of *filename.doc* (MSWord), or *filename.xlt* instead of *filename.xls* (MSEXcel).
- It is in a special locations that are known by MSWord and MSEXcel as sources for templates.

Since MSWord and MSEXcel differ in how they manage templates, they will be discussed separately in a section dedicated to that topic.

In both cases, the advantage of a template is that you can customize basic settings for *all* documents, if you wish. You then apply further customization to each individual template. This can include simple or very complex features such as graphics, fill-in boxes, tables, etc., all the way up to writing powerful Visual Basic macros.

Now, all this can be done in boilerplate files, too. But once you copy a boilerplate file, there is no further relationship between the copied document and the boilerplate. This isn't true with a template—the related document will always be tied to the template. The advantage to this is that if you change something in the template—say, make the default font size 11 instead of 10—this change will be propagated to the derived document the next time it's opened.<sup>1</sup>

A full copy of the template is embedded in your document, however, which is why you can send a copy of a customized document to another system that doesn't have your template and it will still display and print, and can be edited. Changes to *your* template on your original system that are made subsequent to such distribution, of course, won't be in documents on such "foreign" systems.<sup>2</sup>

MSWord or MSEXcel will then offer the template as a document type from the Toolbar under *File/New*. When selected, it automatically creates a new document with all the characteristics of the template, without changing the template itself. This alone prevents the corruption that is endemic with boilerplates.

### 3.2.1 **Visual Basic—Not**

I mentioned Visual Basic a bit ago. For the purposes of this document, Visual Basic is far beyond the scope of what we can discuss; but we will describe some of the more elegant things that can

---

<sup>1</sup> Unfortunately, this isn't true for *all* attributes of a document; but it is true for the most common and useful ones.

<sup>2</sup> No, I don't know if you bring that memo back to your system and then open it, if it will import those changes. I think it should, but haven't tried it...

be done in MSWord with tables and fill-in fields. Just using these features can provide very complex and useful forms for a wide range of applications.

### **3.2.2 MS Excel—Almost Not**

I will admit to a very definite bias—I'm not commonly a MSExcel user, and prefer more robust applications for number crunching.<sup>1</sup> I know it, but don't prefer to do a lot of work in it. Therefore, there is a definite bias toward the creation of MSWord templates in this memorandum—like, I'm only going to discuss specific techniques for creating MSWord templates. For Excel, you're on your own. However, the key item to remember is that anything you can do in a normal spreadsheet can be done in a template.

## **4. TEMPLATE MANAGEMENT**

This is important, since where you put your templates affects how they are presented to you in MSWord or MSExcel. It's not hard or complicated—you just have to know the magic invocations.

### **4.1 MICROSOFT WORD**

Microsoft word will offer you a normal set of templates delivered with the word processor—just go to the Toolbar and open *File/New*. Every one of those file types under the different tabs is nothing more than a template.<sup>2</sup>

You can add your own templates to this list—in fact, you can create your own “tabs” that show up in the top of that dialog window.

#### **4.1.1 Locations**

Microsoft Word “knows” about two locations for templates. Have any document open, and from the Toolbar open *Tools/Options* (for some reason, they won't let you change options unless you've an open file.) In the resulting *Options* dialog box, look for the tab entitled *File Locations* and select it.

You'll see a number of locations offered—some of them are interesting and useful, and you can investigate them later—but the two we're interested in are labeled *User Templates* and *Workgroup Templates*.

A note here—when you fill in either or both of these locations, *don't* use mapped drive letters. It's not needed, and wastes them for programs that might really need them. Not to mention that you have to maintain them on the workstation, and if they get “broken” you break this, too. Just use the full network address, i.e., `\\TKSERVER1\templates`.

---

<sup>1</sup> What do I mean by this? Well, what worries me most about spreadsheets in general is that they try so darn hard to recover from bad numbers and equations that they can hide things like circular dependencies. There is also little in the way of debugging or diagnostic tools for spreadsheets. All in all, it's too easy to get bad numbers that are presented as if they're good.

<sup>2</sup> In fact, you can change any one of them, or all of them. I don't think this is a good idea, in general, since people expect the regular templates to work/look the way they were shipped by Microsoft; but you *could* do it.

#### 4.1.1.1 User Templates

*User Templates* is usually set to point to your local copy of the templates distributed with Microsoft Word—and this is normally **C:\Program Files\Microsoft Office\templates**.<sup>1</sup> You don't normally get to play with either this path, or the files and directories there (unless you're the network administrator.) Play with the path, you may find nice things like the normal Blank document disappearing. Change the templates at that location, and you'll either get what you want, if it's your own computer, or get clobbered by the network administrator for screwing up the templates for everyone else. (Unless you rapidly ask pre-clobbering just why they were left unprotected so you *could* change them...)

#### 4.1.1.2 Workgroup Templates

*Workgroup Templates* is unset as installed by MSWord. This means that any value you see here had to have been put there by the network administrator, another user, or you. It almost invariably points to the group of templates created by your office or group.

I said almost. Even with two choices here, there aren't enough places to look for templates. Sometimes, individuals have templates that they don't want to share with the group (or the group doesn't want them to share...), but they'd still like to use as templates. In this case, as long as you remember the group-approved value for *Workgroup Templates*, it's easy to set up your own directory and just change this value whenever you're feeling template-antisocial and want to play in your own sandbox.

#### 4.1.1.3 Tabs and Directories

Do look in the directory pointed to by either entry, though—you'll notice several subdirectories. An astute observer will also notice that each subdirectory matches a tab in your *File/New* dialog box! It's that simple to create a new tab there. Obviously, there will be a limit to how many you can have—either very long directory names, or a *lot* of directories, will be too big to hold in that dialog box header—but it's pretty reasonable.

This particularly means that you want to create subdirectories in the Workgroup Template directory to break up that big list of templates into groups related to function.

Unfortunately, this only works one level down—any more directories, and they all get lumped under the top-level tab when displayed as choices by Word. You can have almost any number or depth of subdirectories, though, and Word will run down them to collect the choices. So, even though you won't see this internal structure, you can use it to break down the glop of template files in the directory for purposes of managing them.

You may notice that, if you create an empty directory, you might expect to see a tab with no choices in it. Nope. If a directory is empty, its tab will never be displayed. The first time any template file is found in the directory, any level down, and it'll show up.

### 4.1.2 Normal.dot

There's one, particular, special template file that you should know about. It's distributed by Microsoft, but it violates my suggestion that you *not* change such files. Entitled *normal.dot*, it should be found in the directory of the *User Templates* directory.

---

<sup>1</sup> And we immediately break this rule on most networks. You always want people in a workgroup using the same set of templates—including those from Microsoft. It's especially important, since *Normal.dot* is commonly changed for a particular workgroup, that this be the case. SO, if you look at your value for this and it's pointing off to some network directory on your server—that's why!

It's the underlying base for all templates—the Mother of Templates, so to speak. Change something here, and you're probably going to change a lot of things. SO, on the Good Side, you can make sure that Ariel 11-point is *the* default font and size in your office. On the Bad Side, if you really trash this template, I hope you've got a backup. Type once, think twice before saving. Maybe three times if you've pointed everyone on the network to the same template directory.

## 4.2 MICROSOFT EXCEL

Microsoft Excel offers pretty much the same capabilities as Microsoft Word, but it's more confirmed in its ways—you can't change the "normal" template location.

There are far fewer templates delivered with Excel, and it doesn't come with its equivalent to the *normal.dot* templates created for you. Instead, if these files—documented below—aren't already created, Excel will use "compiled in defaults"; that is, whatever the programmer(s) decided you want. (Sounds kind of crude, but note that most people never create the default files—so it must work to some extent.)

### 4.2.1 Locations

Microsoft Excel also "knows" about two locations for files. Unfortunately, one of them is hard-coded into the program. It also treats these directories as much more than just template folders—it will try to open *any* file it finds in these directories, except for templates. This means if you point to a directory with some worksheets in it, they'll automatically be opened. If you want to work on them every time you start Excel, this is great. Otherwise, it's a pain. But worse—if you have other files, such as MSWord documents there, Excel will blindly try to open *them*. Or even program (.exe) files. This is a Bad Thing. Be careful with what you put in these directories.

Also, don't just save a template by renaming its suffix to *.xlt*—you **must** save it by using *File/Save As*. Don't ask me, I didn't write the rules.

#### 4.2.1.1 XLStart Directory

This is the "hard-coded" directory. It will be wherever is the "Microsoft Excel Directory"—usually **C:\Program Files\Microsoft Office\Office\XLStart**. I haven't checked, but I expect that when you install Excel, this gets recorded, so you can't really move it, either. (But then, you can't just move installed programs under Win95/98 or NT, anyway...)

This is specifically where *book.xlt* and *sheet.xlt* (next sub-section) are supposed to go, too. It says that if you put them in the Alternate Startup Folder, that'll work, too, though.

#### 4.2.1.2 Alternate Startup Folder

This is where you put all your own template files. As mentioned, when you create them, do a *Save As* and tell Excel they're templates. After that, you can move the file anywhere you want.

Once you've picked a location—I suggest Excel directories in the same directory you pointed to for MSWord, to make it easy to maintain—move the files there, then from the Excel Toolbar, pick *Tools/Options*. The *General* tab will offer the *Alternate Startup Folder* location. No, you can't browse as you could in MSWord—you have to type the entire location. Again, don't use mapped drive letters; instead, use the full network path, i.e., `\\TKSERVER1\templates`.

### 4.2.2 Book.xlt

If you save a template with the name *book.xlt* in the *XLStart* or *Alternate Startup* folder, Excel will use it as the default for all formatting, content, etc. for every new book you open. If you don't

create this, Excel guesses what you want from whatever defaults the programmer decided upon, and/or whatever is set in your options.

### 4.2.3 Sheet.xlt

Similarly, if you save a template with the name *sheet.xlt* in the *XLStart* or Alternate Startup folder, whenever you do an *Insert/Insert Worksheet* from the Toolbar, it will use the formatting, content, etc. saved in this file when it creates the new sheet.

## 5. STORAGE ISSUES AND HINTS

Before we start talking about what you can put in templates, please remember that Microsoft tools in general make “fat files”. If you include objects, such as photos or bitmap images, in the file, it gets all that much fatter. A lot of these can eat a lot of disk space—heck, the way MSWord works, only a few of them can eat a lot of disk space. And it’s nastier than you can imagine, thanks to Word—unless you do it right.

For instance, I selected a JPEG file (*.jpg* extension), of the type created by a digital camera. The original is only 63Kb—about 64,000 characters, if you want to think of it that way.<sup>1</sup>

You include a photo in a MSWord file by using *Insert/Picture/From File* from the Toolbar. Ok, I did that—went to where this file was stored on disk, and selected it. There are two innocuous-looking buttons in the dialog that lets you do this. One says, *Link to File*; the other, *Save with Document*. Both are off by default. This Is Bad.

I created one file just using the default settings for these options. The photo was included, and looks great. How big was the saved file? 499K. That was, exactly 510,976 bytes. Why? Because MSWord converted the photo from the relatively compact JPEG format to a bitmap—much larger—then stuffed the whole thing into the file.

Ok, pick *Link to File*—meaning that you just *point* to the file on disk. Right? Well, not exactly. Because when you do that, Word automatically clicks on the *Save with Document* checkbox. And guess what happens? Right. It stuffs a copy of the picture into the file, and we’re back to 499K.

The only choice that makes sense is to turn on *Link to File*, and turn off *Save with Document*. In that case, the saved file is only 19K—this, we can live with. You have similar choices for virtually anything you include in a MSWord document. (Or Excel spreadsheet.)

So, the point is—check what you’re creating. Don’t include that huge graphic as a letterhead in your template unless you do it the right way, or you’ll find that your 3Gb disk is a lot smaller than you thought.

Incidentally, you might ask why *does* Microsoft save these files so inefficiently, if they have a better way? Well, there are uses for each of those huge files. If you include the photo in the file, it can be sent to someone else on a floppy or over a network, and not lose the picture; if it’s just linked, take the memo away from your network or disk, and it can’t show the picture. (How do you give away a good copy of a document that has just links? Look at *Edit/Links* from the

---

<sup>1</sup> In computerese, a “K” is 1,024 bytes. So, a 63K file is  $63 * 1024 = 64,512$  bytes—maybe with up to 1023 more bytes, since we’re just giving an approximate size to the nearest K. Actually, Microsoft reports to the nearest whole K when you look at a Windows 95 display; if you select the file and look at its *Properties*, you get a closer approximation (in this case, it then reported the size was actually 62.3 K), and an exact size (63,876 bytes). And for visualizing what this means to normal humans, a byte is what’s needed to store a single character of text (at least, in most cases. Don’t ask about internationalization...)

Toolbar. You have the choice of telling it then to include the objects in that copy of the document, then save it on the outbound floppy, or Zip, or E-mail, or whatever.)

If it's linked and saved with the file, then it'll be updated every time the picture is changed, so any copy you send out is the latest version. Microsoft decided to use as the default the one that will guarantee that any copy of the document will always show the included object.

## **6. FORMS CREATION**

Given all this, one of the most useful purposes for a template is to create a protected form with fill-in fields and checkboxes. We'll discuss how to design one from a blank form, and then how to take a boilerplate and turn it into a form that matches the original in appearance, but has "live" fill-in fields.

Another very useful characteristic of tables is that you can do simple calculations on entries in a table. For instance, you can sum all the entries in a column, or calculate a percentage.

### **6.1 GENERAL APPROACH**

The problem most people encounter when trying to create forms is that virtually all fonts in use today are variable pitch—that is, the letters take differing amounts of space based on size, font, etc. This makes fill-in lines almost impossible to anchor, and they change length based on what's typed in them. Boxes are very hard to draw using the Draw option of Word, and don't fit in well with the text.

The solution is a combination of Tables and objects from the Form Toolbar, along with Document Protection when you're done.

Tables are easy to create, and provide the fixed fields and location needed for such forms. Using the *Format/Borders and Shading* option, you can create underlined frames for the fill-in fields.

The Form Toolbar offers three data fields that you can simply drop in place, and with no sophisticated programming, you get fill-in text with formatting; checkboxes; or drop-down lists.

Finally, you protect the document, and any text and formatting you've selected is protected from changes, while the fill-in Form objects can now accept data.

This is the simple explanation; following, we'll create a simple piece of a form to show how you accomplish this.

#### **6.1.1 Table and Format Manipulation**

In the following, there will be somewhat abbreviated discussion of how to use Microsoft Word features such as Tables and Formatting. In most cases, there will be many more options available to you in the dialogs offered by Word than we discuss here. In those cases, assume that any value or default offered by MSWord in such entries is what we want.

In cases where I think some explanation may be useful, but isn't necessary to carry out the step being described, it'll be tagged as a footnote.

### **6.2 ACTIVE FILL-IN FORM**

Let's assume you want something that looks like:

Name: \_\_\_\_\_ Age: \_\_\_\_\_

Address: \_\_\_\_\_ Phone: \_\_\_\_\_

Options:  Full  Normal  Minimum

First, count how many columns you'd need in a table. Remember you can merge or "join" columns to change the number of entries available. (You can split them, too, but it's usually easier to merge.) In the above, the line with the most items is the last—7. (The first line has 4, as does the second.)

So, select from the Toolbar *Table/Insert Table*. You'll get a dialog box asking how large to make the table; define 3 rows, 7 columns. Then choose AutoFormat, Grid 1<sup>1</sup>. Turn off all special format checkboxes; hit *Ok*, then *Ok* again to actually create the table. You'll get the following:


Now it's time to actually start configuring the table. First, select the entire table—click in any one cell, then go to *Table/Select Table*. The whole table will highlight.

Then go back to *Table/Cell Height and Width*. Turn off the selection, *Allow row to break across pages* in the **Row** tab. Make sure to leave the default alignment as *Left*. Select the **Column** tab; set the *Space between columns* to zero.

If you're happy with the font type and size, OK. Otherwise, while the whole table is selected, go to *Format/Font* and set that to what you would like. Later on, we can change individual cells from this default—for instance, static text can be made bold.

In any case, go to *Format/Paragraph* and make sure all *Indentation* and *Spacing* values are set to zero—this can make things difficult to align if they're not.

Finally, again while the whole table is selected, select *Format/Borders and Shading*. If it's not already on the **Borders** tab, select that. You'll see a series of choices for border styles; under *Settings* on the left-hand side, choose *None*. (Right now, we don't want anything but the controls that table cells give us.) Now strike the *Ok* button on the bottom of the dialog box.

After all this, on-screen you should see a pale grid outlining the table and its cells. If you don't, go to *Table/Show Grid Lines*, since someone turned them off. I'd show you here how this looks, but those grid lines are only on-screen—they don't print. So please remember, in the rest of this description, the lines you see are here because I *didn't* get rid of the borders, so I can print this document. I did set them to the finest lines I could choose.

Now populate the table; the first cut will look like:

Name:					Age:	
Address:					Phone:	
Options:		Full		Normal		Minimum

---

<sup>1</sup> AutofORMAT provides a large number of pre-defined tables. Grid 1 is the simplest predefined table with tops & sides on all cells.

This is closer, but still doesn't look like what we want. Where are the check boxes? Let's drop those in now. First, look at *View/Toolbars*. If *Forms* isn't checked, please do so. You'll now see—or already saw, if it was checked—a toolbar at the top of your screen offering a highlighted **ab**, a white checkbox with a check, and a stylized (and almost illegible) graphic representing a dropdown box. (It also has several other boxes for controlling form fields and tables.)

Check the box just after the *Options:* column, and set the paragraph to center. You can either click the center icon on the *Formatting* toolbar, or you can go to *Format/Paragraph* and set *Alignment* to *Center*. In either case, now click on the check box. A black checkbox should appear in the center of the column. Do the same for each of the other checkbox columns; it should look like this when you're finished:

Name:					Age:	
Address:					Phone:	
Options:	<input type="checkbox"/>	Full	<input type="checkbox"/>	Normal	<input type="checkbox"/>	Minimum

Closer...but how about all those extra columns in the first two rows? Simple. Select the four columns after *Name:*. (Click in the first one, then drag the cursor across all four so they're highlighted. Then select *Table/Merge Cells*. Do the same in the *Address:* line. The table now will look like:

Name:					Age:	
Address:					Phone:	
Options:	<input type="checkbox"/>	Full	<input type="checkbox"/>	Normal	<input type="checkbox"/>	Minimum

Much closer, but we don't have underlines on your screen—remember they're just here for display purposes—it really would print like:

Name: \_\_\_\_\_ Age: \_\_\_\_\_  
 Address: \_\_\_\_\_ Phone: \_\_\_\_\_  
 Options:  Full  Normal  Minimum

And the columns aren't spaced right. How to fix this? Again, simple—just select a column and drag it. You can easily drag an entire column to resize it. While we're at it, select the cells that hold the *Name:*, *Address:*, *Options:*, *Age:* and *Phone:* labels and right-justify them so they'll line up with their underlined next entry:

Name:					Age:	
Address:					Phone:	
Options:	<input type="checkbox"/>	Full	<input type="checkbox"/>	Normal	<input type="checkbox"/>	Minimum

Good so far, but that last line is pretty far off. Again, just drag the columns to make everything the right length:

Name:					Age:	
Address:					Phone:	
Options:	<input type="checkbox"/>	Full	<input type="checkbox"/>	Normal	<input type="checkbox"/>	Minimum

A hint here—you'd probably like all the checkbox columns the same width; that can be hard with the mouse. If you click on the cell such that it's selected<sup>1</sup> then go to *Table/Cell Height and Width*, choose the **Column** tab, you will see that you can set the width very precisely—to the 100<sup>th</sup> inch, allegedly. Set the first column by eye, then read its size. Now go set all the remaining columns to that value.

Ok, we're almost done. Oh, yeah—we need those fill-in fields for everything that's not an *Option*, right? Again, back on the *Forms* toolbar, this time we'll use the first choice labeled **ab**. Click in the table cell next to *Name:*, then click the **ab** entry. Do this for each of those fields; you'll now have a table that *looks* the same right now, but is different—if you click on the beginning of each of those fields, you'll see a highlighted box that indicates there's a form field there. It will, when the file is protected, accept data and show up as a highlighted field in the table. Trust me.

The last thing we'll do now is add in the underlines. Select the field after the *Name:* label—the one you just put that form into—and drag down to get the *Address:* field, too.<sup>2</sup> Then, in *Format/Borders and Shading*, turn on the line for just the middle and bottom—which are the two lines in the part of the table we're playing with. Do the same for the *Age:* and *Phone:* data field cells; now, with all my temporary borders turned off and those turned on, we'll see on the screen:

Name:	_____	Age:	_____
Address:	_____	Phone:	_____
Options:	<input type="checkbox"/> Full	<input type="checkbox"/> Normal	<input type="checkbox"/> Minimum

The last thing to do is to turn off those grid lines—*Table/Hide Grid Lines*—and protect the document. If you don't protect it, then those active fill-in cells won't work. You do this by going to *Tools/Protect Document*. You'll be offered a dialog box; select protection for *Forms*. (There are choices for a password—if you wish, but don't forget it!—and *Sections*. This is a bit more than we'll go into here.)

It's done! Save it in one of the template directories discussed earlier—making sure it's saved as a *.dot* file—and then test it with a *File/New*.

You can see that you could—and usually do—create more than one table on a form, so you can logically group items that belong on different places on the page. Text and other items on the page—or in the file—are also protected when you do the form protection.

### 6.3 CREATING AN ACTIVE FORM FROM A “PASSIVE” FILE

Often, you'll find that someone has created a boilerplate, or even a template, but it's just a “passive” file. There are no fill-ins, and while it may be protected by being a template, the fun and games of alignment and filling in data are still in your hands.

The usual approach to converting this to a form is to identify the areas that would be in the fill-in table, and then drop a table under the block. Then look at the old example while you create and

<sup>1</sup> The cell must be all highlighted. This, too, can be tricky—there are at least three ways to select that cell. One, the way we want it. Two, just clicking *in* the cell, as if you were going to type in it or insert another item—in this case, you'll just see your cursor. And three, you can select the checkbox itself—it should be “grey highlighted” if you do this. This is useful and necessary sometimes—but not now.

<sup>2</sup> This can be tricky—for instance, try selecting *just* that one cell. It really can't be done. Sometimes, if you find funny problems selecting a cell or cells in a table, you may have to select more than you want, then clean it up. Or, probably better, select the individual cells and set/unset borders before you do all the fancy resizing and merging of cells. There's more than one way to do all of this.

modify the table—in fact, copy and paste their label text as needed. Finally, when it looks right, delete the original text.

## 6.4 USING TABLES AND BOOKMARKS FOR CALCULATIONS

There's a bit of clarification needed here, since this can be confusing. First thing—many people don't know you can do calculations in a Word document, or in a Word table. They're very similar, yet different.

### 6.4.1 Calculations in a Word Table

Without using any fields, calculations in a MSWord table are pretty straightforward—if the table is regular in shape and number of cells. For instance, let's take a small, simple table:

	A	B	C
1	12	24	36
2		33	69

If the table is three columns wide and two rows deep<sup>1</sup>, then addressing is as shown above, e.g., A1, B1, C1, A2, B2, C2. If you go into *Table/Formula*, you can insert a pretty wide range of function calls into a table cell—something like 18 operations such as SUM, ABS, AVERAGE, etc. And, of course, the usual four, add (+), subtract (-), multiply (\*) and divide (/). So, if you simply entered numbers in A1 and B1 above, C1 could have =SUM(LEFT) inserted from the *Formula* entry, and you'd get the sum shown in C1.<sup>2</sup>

### 6.4.2 Bookmarks and Calculations

Bookmarks are used throughout MSWord to reference items in the document. One use of a bookmark is to provide the contents for a calculation.

Suppose you have typed in a number elsewhere in your document and would like to use it in the table (or, for that matter, anywhere else.)

Select the text—or number, including the dollar sign, if used. Then, from *Insert/Bookmark*, type a name you'll want to use for this—pick something that makes sense to you!—and *Add*. From now on, anywhere in the document—before or after this bookmark's location—you can retrieve the contents by simply doing *Insert/Cross-Reference*. When the dialog box appears, select *Reference Type* as *Bookmark*. Then simply select the *Insert* button.

For calculations outside a table, use the bookmark's name. Within a table, you can use the cell number into which you inserted the bookmark.

**WARNING:** Even though there are calculations going on here, MSWord is *not* a spreadsheet. In particular, unless you tell it to do so, it won't know when it should run through its calculation list

---

<sup>1</sup> If you're reading this on a printed page, then I'm OK—you don't know. If you're looking at this from within Word, you're probably wondering what I'm talking about—it *shows* you there are actually 4 columns and 3 rows. Well, it was the easiest way to label the table, OK? Just make believe you can't really see those columns/rows, and don't bother looking at the formulas in the cells—they're not going to match the text.

<sup>2</sup> How do you know what can be used in the function calls—like the **LEFT** for the **SUM** function? Go to Help for the different function names offered in the *Table/Formula* dialog.

and re-calculate values. You do this by striking the **F9** key. *If you don't do this, and change a field used elsewhere in a calculation, the numbers will be wrong!* (Another plus for forms.)

## **6.5 USING TABLES AND BOOKMARKS IN FORMS FOR CALCULATIONS**

There are real problems, however, if you want to try and turn a memo with calculations into a template or use it as a boilerplate. The user can destroy or corrupt the formulas in the table just by typing over them, since you can't tell there's an equation there unless you turn on viewing of field codes.

Worse, as mentioned earlier, if you go back and change one of the inputs to a formula, it *doesn't* automatically get updated! You have to select the cells that are formulas and strike the F9 function key to force an update.

This may be fine for a memo you're writing for yourself, but it just doesn't work in a template.

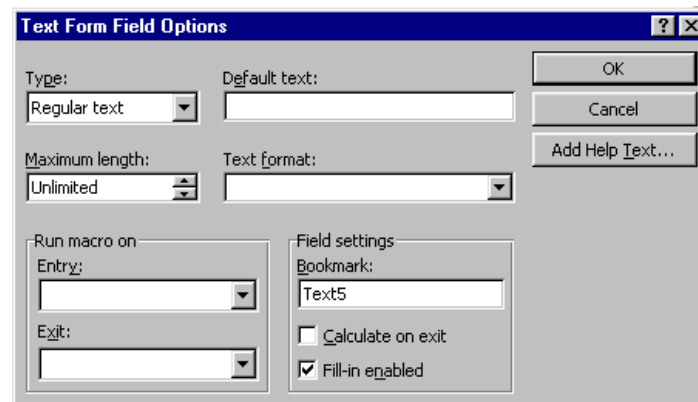
### **6.5.1 Calculations in a Form**

Ok, so you'll protect the template and treat it like a form—uh, you can't enter anything in the table; it's protected just as the rest of the form is (or, at least, should be.)

Fine, insert a Text Form Field. But to use this, now you need to look a little more closely at that beast. So we'll detour here and have you examine that form field a bit more closely.

#### **6.5.1.1 Form Field Properties**

First, insert a text field, then select it—click on it until it's highlighted—then right-click on it and choose *Properties*. You'll get the following dialog box.



The important item to notice is that there are a *lot* of features you can control for this object, and in fact, for every one of the form objects. We'll discuss what you can do in this dialog, and then you can take it further by examining the *Properties* of other form objects such as check boxes and dropdown lists.

The first thing to notice here is that you can change the *Type* of text; this has a major effect on exactly what this field is and does. Your choices in the *Type* box are:

- Regular Text
- Number
- Date
- Current Date
- Current Time
- Calculation

For each of these, you will find that the choices offered in the *Text Format* field change. (Well, actually, they're identical for *Date* and *Current Date*.) The formats for some of these look ugly, but you can pretty well figure them out by inspection—for instance, if for a number, you pick `##,###0.00;($#, #0.00)` you may not know exactly *how* this format works, but you can pretty easily figure out it's what you want for currency display. Play with it a bit—it'll sort itself out quickly.

Calculations can be a bit complex, so we'll save that for a separate discussion following this overview.

The *Maximum Length* field lies, sort of—but not really. The longest string you can put into a field is 32,767 characters, and I believe that's the quiet limit on *Unlimited*, too. But then, I don't feel like typing a 32-thousand-character field to test that. Generally, you'll want to put limits on a string when you really don't want someone to be able to type enough in a field that's in a table cell to cause the table to resize itself. (Didn't know it would do that? You do now.) And even though you can set it to 32,767 for a number, I'm not sure what that would buy you.

*Default Text* is pretty obvious, too—type something in there, and that's what will be in the field when it's first displayed; or nothing, if you put nothing here.

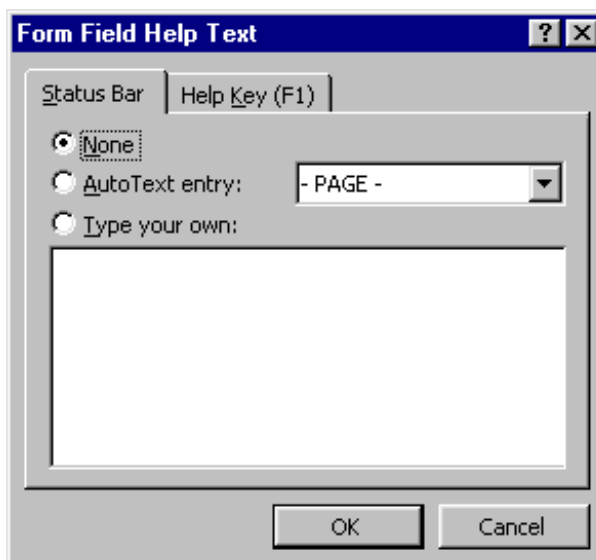
*Field Settings* are important in a form, particularly if you have a calculation somewhere on the form that uses this field. You'll have to check *Fill-in enabled*, of course, if the user is to put

something in this field. If the field is used in a calculation elsewhere, you *must* also check *Calculate on exit*—which doesn't mean to calculate when you exit the field that does the calculation<sup>1</sup>, but when you exit a field in which a user can enter data.

The *Bookmark* is also important if you're going to do any calculations with this field. It's the label, or identifier, you'll use to refer to this field. MSWord will assign a unique bookmark whenever you create a field—although you can corrupt that, for instance if you're tempted to cut then paste a field you've formatted. It will copy all the formats, *and* the same bookmark name; don't forget to change them to be unique after the paste!

*Run Macro on Entry/Exit* is a whole world beyond where we can go in a quick overview. Particularly as of Word97, macros are written in Visual Basic. This is an entire programming language in its own right, and not well documented. Certainly not for the faint of heart. If you want to play with these—and you *can* do some extremely sophisticated things here—you've graduated beyond this memorandum.

Finally, *Add Help Text* will give you this screen:



The **Status Bar** tab lets you select one of your existing AutoText entries—generally not too useful, and if you're not sure what this is, don't worry. See Help if you're curious. The most common use is to *Type your own*. Then, whenever you're in this field, that status bar on the bottom of the screen will show this text.

If you have anything complex to say, most people put something like "Strike F1 for help" here, and then go to the **Help Key (F1)** tab. The same choices exist here, but if you *Type your own*, it will then be displayed whenever they strike the Help Key. Pretty useful, actually—except you can't write War and Peace in this box. It has a limit, so be concise.

---

<sup>1</sup> Even though you can set that button for a calculated field, it doesn't do anything useful, since you can never *enter* a protected, calculated field in a form. Bug. Microsoft knows about it; Microsoft Knowledge Base Article ID Q163158, dated June 11, 1998.

### 6.5.1.2 Calculations in a Form (For Real, This Time)

Now we're to the point of really considering how to do calculations. First, notice that it's incidental that you're in a table if you're following our method of building forms; you could scatter named fields throughout your form, and refer to them by the bookmark name. Secondly, you don't want to refer to fields in a table by the table cell address as we did for a "bare" table. Finally, experience has shown that using calculated fields in other calculations can provide results that are...questionable.

So, what does this mean to you? Well, let's assume your table from the example above has form fields in it; I'll type in the names that I might give the fields when I created them:

	A	B	C
1	Input1	Input2	Sum1
2		Input3	Sum2

I pay no attention whatsoever to cell addressing (A1, B2, etc.). Instead, in the *Properties* for *Input1* and *Input2*, I certainly make sure that *Fill-in enabled* and *Calculate on exit* are checked. Then, for *Sum1*, I would select a type of *Calculation*, and in the *Expression* field (which is what *Default Text* becomes), I'd enter **=(Input1 + Input2)**. That'd be it for that calculation! You may try doing something like **=SUM(LEFT)**, but I'd surely test it before trusting it.

Now, recall that *Sum2* added the value in *Sum1* and *Input3*. Again, make sure the properties for *Input3* are set—but using the calculation result from *Sum1* doesn't give you what you'd think. (Try it!). What happened? Microsoft goofed (and admitted it in writing.)<sup>1</sup> But that's OK, they figure, because they've told us how to fix it. Fortunately, this is only necessary for calculations that use other calculations:

1. Go ahead and create all your calculations. Try it with numbers you trust and see if they come out OK. If so, you're done. Else, continue.
2. Turn on *Field Code View* (while the document is unprotected) by pressing **ALT + F9**.
3. Select the first Calculation form field that is referenced in your formula field; for instance, in the example above, you'd select all of **Sum1**. Of course, it's going to look something like "{ **FORMTEXT { =Input1+Input2}** " (That's what you *really* create when you insert a calculated text field.) Note that there is a space at the end of this thing—select it all, including the space.
4. From the Toolbar, select *Insert/Bookmark*. You should see the name of the field; in our example, **Sum1**. Change that to something different but obviously related—I'd suggest **Sum1calc**. Click *Add*. (You're actually creating a new bookmark to refer to the field—for whatever reason, this fixes Word. **Sigh**.)
5. Turn off the Field Code View by pressing **ALT + F9** again.

---

<sup>1</sup> Microsoft Knowledge Base Article ID Q110656, dated May 7, 1998.

6. In the second calculation field (and/or any others using the “bad” name), click on the field, select *Properties*, and replace the “bad” name with the “good” name you just created.

It took longer to type and read than to do, trust me. Hopefully they'll have this fixed in the next release, but it's not bad enough to ruin the convenience of this feature (the calculation, not the bug.) Especially since these bookmarked fields can be referenced *anywhere* in the document, not just in tables.

## **7. ADVANCED TOPICS**

Ok, so we lied. There are a couple of annoying problems with templates that it would be useful to know how to correct, but they require some more advanced techniques. Again, while this won't tell you *every* way to use these features, it will let you know they exist and how to start to use them.

For instance, you can't apply spell checking or complex formatting within a form field. If you unprotect a document, then reprotect it, your data will be cleared. These two problems alone warrant a visit to these capabilities of Word.

### **7.1 SECTIONS**

Sometimes, you find that you need to change more than fields will permit; for instance, you've some tables at points in a form, but you need to permit large areas of text input, including complex formatting. Or you'd really like to allow input of additional section headers, and use of spell checking.

For these situations, you can insert a *Continuous Break* (Toolbar Insert/Break/Continuous). From this point on, you will notice that the lower left-hand corner of your screen will indicate, just after the current page, the current *Section* (Sec). Now, when you Protect the form, you can select which sections to protect for forms input! Bracket tables or field sections; protect major Headings while permitting insertion of sub-Headings.

A caveat here. I've found no way to display continuous breaks—they're not visible when you turn on viewing of Field Codes. You have to watch that section indicator. Or you can search for them by Find/More/Special.

### **7.2 PROTECT MACRO**

It's really annoying to lose everything you've typed by unprotecting a document, either deliberately or inadvertently. Once you've unprotected a document, you can't go back again. *Unless* you include this snippet of a macro into the document:

```
Sub ProtectForm()
'*****
' ProtectForm Macro.
' Toggles protection for the active document
' when the Protect Form button on the forms toolbar is clicked.
'*****
If ActiveDocument.ProtectionType = wdNoProtection Then
    ActiveDocument.Protect Type:=wdAllowOnlyFormFields, NoReset:=True
Else
    ActiveDocument.Unprotect Password:=""
End If
End Sub
```

```

Sub ToolsProtectUnprotectDocument()
' *****
' ToolsProtectUnprotectDocument Macro
' Sets protection for the active document when Protect Document or Unprotect
' Document is clicked on Tools menu
' *****
If ActiveDocument.ProtectionType = wdNoProtection Then
    ActiveDocument.Protect Type:=wdAllowOnlyFormFields, NoReset:=True
Else
    ActiveDocument.Unprotect Password:=""
End If
End Sub

```

How, you ask? Well, I can't make you a Visual Basic programmer here—but this is pretty straightforward. From the Toolbar, select *Tools/Macro/Visual Basic Editor*. You'll get a pretty exciting new screen; disregard all of it except its Toolbar, where you want *Insert/Module*. An empty window will open; either type in the above, or, if you've an electronic form of this document, you can open it, sweep out the above text, *Copy* to the clipboard, and then *Paste* into that window! From this point on, you can switch between protected and unprotected mode to your heart's content.

As usual, a caveat for this—you will be warned, when you open a form containing such a macro, that there are live macros that could hurt your system. You know what this one is—permit Word to activate the macro.

## **8. CONCLUSION**

We haven't fully covered all the options and features of form and template creation in MSWord, let alone all the other Microsoft programs such as Excel, Access, etc. But you now know that these type of things can be done; and you know to look for them in the other programs. Don't forget to use the Help facility to look up some of the other possibilities, and search the Internet for hints and suggestions from other users. Have fun!