Easily Automate Any Application

Simple, Free Tool Yields 2× Productivity Increase

First There is a Mountain – A client recently tasked me with consolidating and formatting eleven PowerPoint decks into a single, 700+ slide master deck. The decks were created by several authors having varied familiarity with the application.

My job was to provide an attractive, uniform look—using a new design template—that would enhance readability and my client's professional image. All the while I had to keep an eye on the clock to help minimize my client's costs.

It was estimated to take six minutes per slide based on data I included with the first few completed decks. But partway through the project I conjured a way to cut that time in half—in addition to being able to share my simple-to-use, no-cost productivity enhancement process with the client (who is not familiar with computer coding).

Mac OS users can look to Keyboard Maestro and the native Automator to provide similar functionality. But what I'm about to describe can be used anywhere in Windows 7 or later, and in any Microsoft or third-party application that follows long-established Windows user interface conventions.

"Oh, the Tedium" – Bullet use was a mishmash throughout the client's disparate slide decks. There was no consensus as to when an en- or em dash should be used—never mind corresponding spacing considerations.

Adding to my budding carpel tunnel fear, not all Microsoft Office apps are created equal.

Referencing the numeric keypad while using Word, why can you press [**Ctrl**]+[**-**] to insert an en dash and [**Ctrl**]+[**Alt**]+[**-**] for an em dash, but the same hotkey combos don't work in PowerPoint? (Scratches head.) Yet pressing [**F4**], which repeats the last action, is common to both applications. At least M$ got that right.

Consider the dashes: yes, I could click the **Insert** tab, click **Symbol**, locate one, select it, click **Insert**, then click **Close**. That's five mouse clicks. Instead I initially chose to manually enter their ASCII equivalents (long etched into my memory) from the keyboard. First I had to engage [**NumLock**], then hold [**Alt**] while quickly pressing **0150** (en) or **0151** (em) in succession. But then I'd forget [**NumLock**] remained engaged while I made some keyboard goof a few moments later. *Aargh!*

In Word (2013), it's easy to record a simple macro and not have any idea that Visual Basic for Applications (VBA, its macro underpinnings) exists. But try that in PowerPoint, where there is no facility to record steps as you manipulate dialog boxes. Yes, one can create and use PowerPoint macros—if you're a fluent VBA coder.

Third-tier items required a square bullet 80% of its initial size—that's five mouse clicks and two key presses. And having to click my mouse six times to place a blue, checkmark-style bullet in front of a selected paragraph(s) on.every.single.slide got old very quickly. There were numerous other repetitive formatting minutiae to wrestle with.

I was well into the project when the menial nature of repetitious keystrokes and mouse clicks began to *really* annoy me. Meanwhile the clock continued to run.

Do You Know the Macro Man? – Then I remembered WinBatch, a powerful, easy-to-use scripting tool I used often in years past. Its SendKey function lets mere mortals make almost any window or dialog box perform a series of hoops, jumps, and somersets—through hogsheads of real fire—sans VBA.

I unearthed my WinBatch 2006 disc from the catacombs and promptly installed it. Now it was a matter of figuring out the keystroke sequence for the first macro.

Shortcut Key History – Rather than roll its own application from scratch, Microsoft bought PowerPoint, retooled it, then offered it for Windows 3.0 (!) users in 1990. In those days few Windows users existed, let alone PC owners who owned a mouse. It made sense that Windows, and any Windows application, would have to be capable of running 100% from the keyboard by way of "shortcut keys."

This remains true today. No doubt you've seen the underlined shortcut keys in Windows menus and dialog boxes and have wondered what they're about. Dropping down the **Paragraph** menu in Word, for example, you see the **Indentation – Right:** shortcut key is **R**, **Spacing – After:** is **f**, and so on. Pressing [**Tab**] several times lets you move the focus within a dialog box and, once it's on [**OK**], pressing [**Enter**] executes all changes and closes the dialog box.

But wait! What are the shortcut keys for the various Ribbon tabs, e.g., **HOME**, **INSERT**, **DESIGN**? Press [**Alt**]. Each tab has a letter assigned in reverse type, e.g., H, N, G, respectively.

I took several minutes to manually work out the PowerPoint key sequence, using Notepad (a plain text editor) to note each keypress as I went. Still in Notepad, ultimately I added the SendKey command, polished the syntax, then tested the macro. It worked straight away.

Productivity Value-Add – My website states, *"…I'm often able to suggest process improvements that may result in cost reductions or gains in workflow efficiency."* I desired to share my discovery with my client so, going forward, it can also benefit from efficiencies offered by simple macros.

But WinBatch is overkill. It has also become pricey for anyone who only wants a simple SendKey function. Looking around on the web, I found a couple of free alternatives I thought might work. AutoIT looked like it might be terrific, but I arbitrarily chose AutoHotkey instead.

In under ten minutes I had downloaded AutoHotkey, used Notepad to convert my plain text WinBatch file (this time saving it with an .AHK file extension), and tested the macro. *Perfeto!*

Magic Decoder Ring – Here is what my AutoHotkey (.AHK) file initially looked like in Notepad:

```
^!k::
Send,!HUNB{TAB}{DOWN}{RIGHT 3}!C{TAB}{RIGHT 4}{ENTER}{TAB 3}{ENTER}
Return
```

Pretty simple, huh? It may look daunting at first, but by following along you'll soon have the hang of it.

The first line declares [**Ctrl**]+[**Alt**]+[**K**] as my desired macro hotkey sequence. I chose this because it doesn't do anything natively in PowerPoint, i.e., it's unassigned. In AutoHotkey—like WinBatch—a caret (^) signifies the [**Ctrl**] key, while an exclamation point (!) denotes the [**Alt**] key. Thinking mnemonically, I chose "k" because my mind associates it with "chec*k*mar*k*," the type of bullet I wanted this macro to create.

Instead of *SendKey* (a WinBatch function), AutoHotkey simplifies that as *Send* on the second line. `!HUNB` starts at the **Home** tab (H), opens a basic **Bullets** dialog box (U), then a more definitive **Bullets and Numbering** dialog box (N), followed by placing the focus specifically on the **Bulleted** tab (B). A single `{TAB}`places the focus on None (the first item in the dialog box), while `{DOWN}{RIGHT 3}` selects the ✓ bullet style.

Next, `!C` opens the **Color** dialog, a `{TAB}` sets the focus on **Theme Colors**, `{RIGHT 4}`selects blue, `{ENTER}` confirms that choice, `{TAB 3}` moves the focus to [**OK**], and `{ENTER}` closes the sequence.

One form of coding in AutoHotkey calls for a `Return` on the last line. You'll see why this is useful in a moment.

Well-Tempered Macro File – Stated earlier, in PowerPoint I miss being able to press [**Ctrl**]+[-] and [**Ctrl**]+[**Alt**]+[-] to create an en- and em dash, respectively. The good news: AutoHotkey lets you place several macros in a single .AHK file, such that an entire set of hotkeys is readily accessible during any given session. Each is contained within the hotkey designation on the first line and its corresponding `Return` .

Let's have a look at the modified macro file (with the initial macro now at the bottom for reference):

```
$^NumpadSub::     ;NumpadSub is the [–] key on the numeric keypad
   Send,{SPACE}{-}{SPACE}
Return

$^!NumpadSub::
   Send,{–}
Return

$^!k::
   Send,!HUNB{TAB}{DOWN}{RIGHT 3}!C{TAB}{RIGHT 4}{ENTER}{TAB 3}{ENTER}
Return
```

The optional $ that begins the first line can be used to prevent a hotkey from triggering itself when Send is used (without it, it's possible to create an endless loop). NumpadSub is AutoHotkey's way of specifying the *sub*traction key on the *num*eric key*pad*. Preceded by a semi-colon, the second instance is a note I added for future reference.

My client's style calls for a space on each side of an en dash, and I frequently came across an entry such as *Some item- more words* (using a hyphen and incorrect spacing). On the Send line, then, I sandwiched the en dash (which I physically pasted between the squiggly brackets) between two spaces. In using this macro, I'd simply select everything between the two italicized 'm's in the example, then press [**Ctrl**]+[**Subtraction key**] to get *Some item – more words*.

For the next macro, I pasted an em dash between the brackets. Pressing [**Ctrl**]+[**Alt**]+[**Subtraction key**] generates the — character with no spaces, just as it does in Word.

> **Note:** You can find other special characters in Word by selecting **Insert** > **Symbol** and locating them there. You can also find them in Windows' Character Map applet (CHARMAP.EXE), which you can run from the **Start** > **Run** command box. Yet another way is to visit alt-codes.net, note the numerical value of the symbol you want, then insert it in your work using the [**Alt**]+*<Numpad numerical sequence>* method, e.g., [**Alt**]+**0150** for an en dash.

Let's have a look at one last macro:

```
^!q::
   Send, !HFS18{ENTER}!HUNB{TAB}{RIGHT 2}!S80{ENTER}!HPG{TAB 2}1{TAB}H{ENTER}{TAB}.25{ENTER}
Return
```

This one is a bit different, in that a single flood of keys accesses items from the **Home** tab three times. I'll let you guess which hotkey sequence I assigned it. Pressing it once takes the place of eight mouse clicks, three text wipes, and six keypresses.

For whatever text is selected in PowerPoint, the first pass sets the font size (FS) to 18 pt. The second (HUNB…) selects a square bullet and reduces its size to 80% (!S80) of the original. The third pass accesses the Paragraph dialog box (PG), then indents the selected text 1" with a hanging indent (H) of .25".

Your Saving Grace – Taking a few minutes to find a free tool online, then work out a few key sequences saved my sanity during the remainder of my client's project. Now I could zip through each slide in no time. And having eliminated the price barrier, now I had a great productivity tool I could share with my client—and use myself.