20th September 2007

## [**11 Visual Studio 2005 IDE Tips and Tricks to Make You a More Productive Developer**](http://www.chinhdo.com/20070920/top-11-visual-studio-2005-ide-tips-and-tricks-to-make-you-a-more-productive-developer/)

posted in [Dotnet/.NET - C#](http://www.chinhdo.com/category/programming/dotnet/), [Programming](http://www.chinhdo.com/category/programming/) |

Here are my top 11 tips and tricks for getting things done faster with the Visual Studio 2005 IDE (without using third-party add-ins or upgrading hardware… that’s another article). Yes, some of these tips and tricks may fall into the "obvious" category, but I think they are worth repeating. I continue to see too many .NET developers not taking advantage of even the basic time-saving techniques.

I work mostly with C# so some of these tips may not apply to, or work differently with other Visual Studio languages such as Visual Basic.NET.

### (1) Express Yourself with Regular Expressions

Regular Expressions is a powerful and portable text search/replace/transformation language. Learning basic Regular Expressions will immediately make you a more productive developer/power user. Regular Expressions is supported in Visual Studio’s various Search/Replace dialogs. Any Regular Expressions skill you learn will also be useful in numerous other applications and settings: other text editors, unix shell/egrep, PowerShell, input validation, and Google search (heh, just kidding on that last item).

You can also use Regular Expressions with macros and automation via the Regex class.

Here’s an example of how you can save time with Regular Expressions in Visual Studio. Say, you just wrote and tested a SQL in a Query Tool and you want to turn it into a  string variable in your C# class? Here’s how:

First, paste the SQL text into the editor. Make sure to remove any unwanted indentation on the left side of the text (**SHIFT-Tab**):

Then hit **CTRL+H** to bring up the Find and Replace Dialog and fill it out like this:

Note: For VB.NET, in the “Find and Replace Dialog Window”, change the value in

“Replace With” box: from {[ + “\1″ ]}… to {[ & “\1″ \_ ]}. Thanks to berkx for this info.

Choose Replace All. Fix a couple of lines, sit back and admire your beautiful work:

Explanation? Basically, the "Find what" expression above matches the content of each line and give it a numbered "tag". The "Replace with" expression then replaces each line with the first tagged value (\1), wrapped around in + " ". Click the fly-out (triangle) button next to each box to display a cheat-sheet of frequently used expressions. Oh, and don’t worry about the "+" string concatenations in the example, the [compiler knows to optimize that syntax](http://www.chinhdo.com/chinh/blog/20070224/stringbuilder-is-not-always-faster/).

Once you’ve created a few Search/Replace expressions like the above, create macros out of them and assign to shortcuts.

Here are some of the Regex transformations I use most often when writing code:

* Surrounds each line with (example above).
* Transform a list of values separated by newlines into a coma-delimited list (used in array initializers or SQL where clause).
* Put double quotes around each value in a coma-separated list.

Tools to help you write/test Regular Expressions:

* [The Regulator](http://tools.osherove.com/CoolTools/Regulator/tabid/185/Default.aspx) is a free GUI to compose/test your Regular Expressions.
* [Nregex](http://www.nregex.com/) – AJAX web site.
* [MSDN – How to: Search with Regular Expressions](http://msdn2.microsoft.com/en-us/library/ms174214.aspx).

### (2) Take (Keyboard) Shortcuts

Using keyboard shortcuts is the best way to get things done faster in Visual Studio (and most other computer applications for that matter).

Below are my favorite Visual Studio keyboard shortcuts (I am leaving out the really obvious ones like **F5**).

* **CTRL+ALT+L**: View Solution Explorer. I use Auto Hide for all of my tool windows to maximize screen real estate. Whenever I need to open the Solution Explorer, it’s just a shortcut away. Related shortcuts: **CTRL+ALT+X** (Toolbox), **F4** (Properties), **CTRL+ALT+O** (Output), **CTRL+\, E** (Error List), **CTRL+\, T** (Task List).
* **F12**: Go to definition of a variable, object, or function.
* **SHIFT+F12**: Find all references of a function or variable.
* **F7**: Toggle between Designer and Source views.
* **CTRL+PgDn**: Toggle between Design and Source View in HTML editor.
* **F10**: Debug – step over. Related debugging shortcuts: **F11** (debug – step into), **SHIFT-F11** (debug – step out), **CTRL-F10** (debug – run to cursor). **F9** (toggle breakpoint).
* **CTRL+D** or **CTRL+/**: Find combo (see section on Find Combo below).
* **CTRL+M, O**: Collapse to Definitions. This is usually the first thing I do when opening up a new class.
* **CTRL+K, CTRL+C**: Comment block. **CTRL+K, CTRL-U** (uncomment selected block).
* **CTRL+-**: Go back to the previous location in the navigation history.
* **ALT+B, B**: Build Solution. Related shortcuts: **ALT+B, U** (build selected Project), **ALT+B, R** (rebuild Solution).
* **CTRL+ALT+Down Arrow**: Show dropdown of currently open files. Type the first few letters of the file you want to select.
* **CTRL+K, CTRL+D**: Format code.
* **CTRL+L**: Delete entire line.
* **CTRL+G**: Go to line number. This is useful when you are looking at an exception stack trace and want to go to the offending line number.
* **SHIFT+ALT+Enter**: Toggle full screen mode. This is especially useful if you have a small monitor. Since I upgraded to dual 17" monitors, I no longer needed to use full screen mode.
* **CTRL+K, X**: Insert "surrounds with" code snippet. See Snippets tip below.
* **CTRL+B, T**: Toggle bookmark. Related: **CTRL+B**, **N** (next bookmark), **CTRL+B**, **P** (prev bookmark).

The complete list of default shortcuts is available from VS 2005 Documentation. You can also download/print reference posters from Microsoft: [C# Keyboard Reference Poster](http://www.microsoft.com/downloads/details.aspx?FamilyID=c15d210d-a926-46a8-a586-31f8a2e576fe&DisplayLang=en), [VB.NET Keyboard Reference Poster](http://www.microsoft.com/downloads/details.aspx?FamilyID=6bb41456-9378-4746-b502-b4c5f7182203&DisplayLang=en).

### (3) Make New Shortcuts

There is something you do a lot in Visual Studio and there is no shortcut for it? Create one. Here’s how:

* Choose Tools/Options and select Environment/Keyboard.
* Type in something into "Show commands containing" to get a list of matching commands. If there is already a shortcut for the selected command, it’ll be displayed in "Shortcuts for selected command".
* To assign a new shortcut to the selected command, put the cursor in "Press shortcut keys" and press the shortcut key or key combinations desired.

Have a custom Macro that you run often? Assign it to a keyboard shortcut. Here are some of my custom keyboard shortcuts:

* **CTRL+Num, T**: Show the Test View.
* **CTRL+Num, D**: Start debugging the selected Unit Test in Test View.
* **CTRL+’, L**: "Collapse all in Solution Explorer " macro (see Macros section below).
* **CTRL+’, S**: "Surrounds each line with" Macro.
* **CTRL+’, C**: Compare with previous Source Control version.

### (4) Use Code Snippets

Save time typing repetitive code by using Code Snippets. There are two types of Snippets in Visual Studio 2005: Expansion and SurroundsWith. To use Expansion Snippets, type the Snippet shortcut (not to be confused with keyboard shortcuts), and press Tab twice.

For example, the "for" Snippet is an Expansion Snippet. To use it, type "for"…

Then press Tab, Tab:

I find SurroundsWith Snippets more useful though. An example SurroundsWith Snippet is "#region". First, select a block of code:

Then, type **CTRL+K, CTRL+S** and "**#re**":

Then hit **Enter**:

Here are my favorite Snippets:

* #region: Regions is a great way to organize your code.
* using: If you create an IDisposable object, you should use the "using" pattern. In addition to the basic "using" Snippet, I also created several variations for TransactionScope, and IDataReader.
* try/catch
* {}
* /// <summary>$end$</summary>

More info:

* [GotCodeSnippets.NET](http://www.gotcodesnippets.com/default.aspx)
* [Code Snippets – the whole enchilada](http://jens-schaller.de/blog/post/154)

### (5) State Your Preferences

Find yourself constantly switching to the Design view every time you create/open an ASPX page? Cannot locate your current file in the Solution Explorer? Easy… just change the right settings and never think about it again.

Here are the some settings in Visual Studio that can save you time:

* Open HTML pages in Source View: Tools/Options/HTML Designer/Start pages in.
* Track the current file in Solution Explorer: Tools/Options/Projects and Solutions/Track Active Item in Solution Explorer.
* Change the Start page or get rid of it: Tools/Options/Environment/Startup.
* Change the default font-size to a smaller size so you can see more code. My editor font setting is [ProFontWindows](http://www.google.com/url?sa=t&ct=res&cd=1&url=http://www.tobias-jung.de/seekingprofont/&ei=e4jyRuzSCYzkggSWyZ34Dg&usg=AFQjCNGApSH0SXG0c_FFQGmguWkSe2QfRg&sig2=Rbi_01MyiW6mQLylDyoFXw) at size 9.
* Turn of animations: Uncheck Tools/Options/Environment/Animate environment tools.

### (6) "Attach to Process" to Start Debugging ASP.NET

Most ASP.NET developers use the standard **F5** (Debug/Start Debugging) to start debugging from Visual Studio. However, there is a much faster way to start debugging if you already have an instance of your web application running. Just attach to it instead:

* Choose Debug/Attach to Process.
* Select the "aspnet\_wp.exe" process and choose Attach.

Or, for keyboarders:

* **ALT+D**, **P**, "**as**", **Enter**.

Debugging this way is faster because you skip the often-lengthy compilation step, and you don’t have to navigate from the start page to the actual page that you want to debug.

### (7) Stop Conditionally (Conditional Breakpoints)

How often have you found yourself repeatedly stepping through a loop while debugging, waiting to get to a specific loop value (because the bug only occurs with that specific value)? With Conditional Breakpoints, you don’t have to do that. Just set a Breakpoint Condition.

Set the Breakpoint. Right click on the Breakpoint indicator (red circle), and choose **Condition:**

Set the condition (any valid C# expression):

Another debugging productivity trick I use is to override ToString() to return a useful summary of your objects. The Debugger uses the value returned by ToString in various Debug windows such as Watch Window. You can also use the [DebuggerDisplay attribute](http://msdn2.microsoft.com/en-us/library/ms228992.aspx).

### (8) Employ Task List Tokens

Use Task List tokens such as TODO and HACK to quickly mark incomplete code or code that requires further attention. This allows you to keep flowing and skip over the details, but at the same time ensures that you will not forget to go back and finish up.

A shortcoming with Visual Studio 2005’s Task List is that it only shows the items in the current file. You can get around this by using the Find in Files feature and search for "// TODO".

### (9) Go Directly to Any File with the Find Combo Box

This is the Find dropdown that is on the Standard Toolbar, not the Find dialog. Use the shortcut **CTRL+D** to activate the Find *dropdown* in normal mode. Use **CTRL+/** to activate the Find dropdown in command mode (with ">" prepended… this doesn’t work sometimes for me).

To quickly go to a file, type **CTRL+D**, >**open**  <start of file name>. Intellisense works here just like in the Command Window. "**of"** (short for "open file") can be used instead of **open**. Compare this with opening Solution Explorer, expand the correct folder/project, and visually hunt for the file you need.

With the Find Combo, you can also execute commands, macros, find text, etc. More info:

* [Fun with the Visual Studio Find Combo Box](http://blogs.msdn.com/shawnfa/archive/2004/02/27/81338.aspx)
* Default Visual Studio Command Aliases

### (10) Type Ahead (Incremental Search) in Lists

Type-ahead search works in many Visual Studio lists such as Solution Explorer, Active Files Combo (CTRL+ALT+Down Arrow), Add References, Class View, Attach to Process, Test View, etc.

To see how it works, it’s best to try it yourself. Open the Solution Explorer and start typing the first few letters of a visible file.

### (11) Automate with Macros and Visual Studio Automation

I save this for last because I think macros and Automation have the potential to give you the biggest productivity booster, but also require the most initial time investment.

For many developers, the most effective way to take advantage of macros is to find and use or customize someone else’s macros.

If you want to get started with writing your own macros, the first feature you should get familiarize yourself with is the Macro Recording feature (shortcut **CTRL+SHIFT+R**).

For more info:

* [Automating Repetitive Tasks in Visual Studio](http://www.developer.com/net/cplus/article.php/856631)

Here are some useful macros for you to start with:

* [Collaps All Projects in Solution Explorer](http://www.cptloadtest.com/?p=56).
* [Better Keyboard Support for #region … #endregion](http://weblogs.asp.net/rweigelt/archive/2003/07/06/9741.aspx)
* [Productivity macros for C# in Visual Studio 2005](http://www.codeproject.com/csharp/C__productivity_tools.asp)
* [Visual Studio 2005 Automation Samples (spellcheck)](http://www.microsoft.com/downloads/details.aspx?FamilyID=79C7E038-8768-4E1E-87AE-5BBBE3886DE8&displaylang=en)

### Additional reading

* [Visual Studio 2005 IDE Tips and Tricks](http://msdn2.microsoft.com/en-us/library/bb245788%28VS.80%29.aspx), MSDN
* [Sara Ford’s WebLog](http://blogs.msdn.com/saraford/)