http://www.gsx.com/blog/bid/81096/Enhance-your-PowerShell-experience-by-automatically-loading-scripts

**Enhance your PowerShell experience by automatically loading scripts**

Posted by [Olivier Raynaut](http://www.gsx.com/blog/author/olivier-raynaut) on Wed, Dec 19, 2012

Ever wanted to customize your PowerShell terminal so you can have all your custom function or scripts loaded automatically? Although it is possible to create aliases, functions or variables, everything is only kept for the current PowerShell session, meaning that those changes will be lost after you close the shell.

In order to store the information PowerShell, you can use a “Profile” that is loaded at every session. We’ll now explain how to enable your PowerShell profile and customize it.

## The different Profiles

You can have four different profiles in Windows PowerShell, listed below in load order, the most specific having precedence over less specific profiles where they apply:

* **%windir%\system32\WindowsPowerShell\v1.0\profile.ps1**   
  This profile applies to all users and all shells.
* **%windir%\system32\WindowsPowerShell\v1.0\Microsoft.PowerShell\_profile.ps1**   
  This profile applies to all users, but only to the Microsoft.PowerShell shell.
* **%UserProfile%\My Documents\WindowsPowerShell\profile.ps1**   
  This profile applies only to the current user, but affects all shells.
* **%UserProfile%\My Documents\WindowsPowerShell\Microsoft.PowerShell\_profile.ps1**   
  This profile applies only to the current user and the Microsoft.PowerShell shell.

## Enabling a profile

Open a PowerShell terminal and type in

|  |
| --- |
| **$profile** |

This should display the path that would be used to store you profile. However it might not be created yet, so in order to find out, type the following:

|  |
| --- |
| **test-path $profile** |

If the result is true, you’re all set, else type the following to create the profile:

|  |
| --- |
| **new-item -path $profile -itemtype file -force** |

You’re now all set with your profile, and can customize it, by launching the PowerShell ISE:

|  |
| --- |
| **powershell\_ise $profile** |

## 

## Auto-load scripts on PowerShell startup

In the ps1 file of your profile, usually named “Microsoft.PowerShell\_profile.ps1”, you can enter the following code, so that the folder of your choice will contain scripts that will be executed on startup:

|  |
| --- |
| # directory where my scripts are stored  $psdir="D:\Documents\Powershell\Scripts\autoload"  # load all 'autoload' scripts  Get-ChildItem "${psdir}\\*.ps1" | %{.$\_}  Write-Host "Custom PowerShell Environment Loaded" |

This script will browse the folder given in input, and load all the files. The files I have in this folder are “.ps1” files that contains a PowerShell function.

In order for those scripts to be executed, the policy needs to be updated to something else than "restricted", at least “remotesigned” by running the following:

|  |
| --- |
| **Set-ExecutionPolicy remoteSigned** |

Here you go, you can now put as many “.ps1” files as you need in your “autoload” folder, and these will be executed everytime you launch your PowerShell session.

## Sources

* **Windows PowerShell Profiles (MSDN) –**<http://msdn.microsoft.com/en-us/library/bb613488%28VS.85,loband%29.aspx>
* **PowerShell Startup Auto-Load Scripts –**<http://noahcoad.com/post/66/powershell-startup-auto-load-scripts>

http://noahcoad.com/post/66/powershell-startup-auto-load-scripts

# PowerShell Startup Auto-Load Scripts

Posted on [August 21, 2011](http://noahcoad.com/post/66/powershell-startup-auto-load-scripts) by [Noah Coad](http://noahcoad.com/post/author/noahcoad)

It’s very handy to have a PowerShell script automatically load when PowerShell starts up to configure your environment, like an AutoExec.bat in the old DOS days.  Here’s how I have mine set up.

1. **Tools Dir & Env Var**  
   I keep a general Tools directory on my machine with command line tools, scripts, etc and so set a [system environment variable](http://www.itechtalk.com/thread3595.html) called tools that points to this directory.
2. **Run Unsigned Scripts**  
   Run set-executionpolicy bypass once in each environment (cmd shell and ISE) to allow easy execution of unsigned scripts.
3. **Profiles Auto-Load Script**  
   PowerShell supports [profiles](http://msdn.microsoft.com/en-us/library/bb613488%28VS.85,loband%29.aspx) with an auto-load script on PowerShell start up.  Simply put the files Microsoft.PowerShell\_profile.ps1 (for basic shell) and Microsoft.PowerShellISE\_profile.ps1 (for the GUI) in your %UserProfile%\Documents\WindowsPowerShell directory. These will be run on PowerShell startup.
4. **Consolidate Scripts**  
   The only line I have in these files is: ."${env:tools}\PSScripts\PowerShellEnvironment.ps1" to launch my ‘real’ script. This make it easier to manage, with only one file that’s near my other powershell scripts instead of the two separate scripts above.
5. **Chain ‘autoload’ Scripts**  
   Then it helps to separate into individual scripts various aspects of your environment setup.  Like a script for aliases, itunes, functions, regular expressions, favorite assemblies, etc.  So I have an ‘autoload’ directory with these scripts in them to chain load. Here are the contents of PowerShellEnvironment.ps1:

# directory where my scripts are stored

$psdir="${env:tools}\PSScripts"

# load all 'autoload' scripts

get-childitem "${psdir}\autoload\\*.ps1" | %{.$\_}

Write-Host "Custom PowerShell Environment Loaded"

1. **My ‘autoload’ Scripts**  
   Here is a snapshot of my autoload directory of scripts in case you’d be interested.  [PowerShellScripts-autoload.7z](http://coad.net/blog/resources/PowerShellScripts-autoload.7z)

That’s all there is to it.  One of the reasons for putting this out there is so that I can now reference it from other posts too. 

