The Bash Shell
Tour, Variables, Control Flow, Functions, Builtins, History, Error Handling, Aliases, Best Practices

Bash is a powerful shell environment that provides excellent interactive and scripting control of the underlying OS, installed software platforms and can be used to build and manage your own custom tools too. Bash is good for repetitive chores, setting up an environment, automating non-trivial workflows, software builds and product installations.

The goal of well-written shell scripts is to allow repeatable / configurable / reliable task execution. The overall aim of this course is to equip attendees with a clear understanding of how to achieve that using Bash.

Professionals skilled in the art of Bash scripting are significantly more productive. Knowledge of Bash is a "must-have" tool in the skillset of every heterogeneous system administrator or developer. Though initially popular on Unix and Linux platforms, Bash is now very widely available, including on macOS and Microsoft itself has added it to modern Windows (10 & Server 2019, as an optional install). With a little effort you can even get Bash to work on mobile OSes. This wide availability is important because it means the effort you invest to learn Bash well now will pay repeated dividends in future regardless of which OS you use.

Contents of One-Day Training Course		
	Overview	**Scripts**
	What is Bash good for?	Creating/calling/debugging Bash scripts
	Compliance with POSIX Shell Standard	Sourcing – external library of functions
	Feature tour	Job control
	Setting up Bash – for Linux/macOS,	History
Target Audience	Windows Subsystem For Linux and	Arranging larger blocks of script
Administrators, developers	Git Bash (limited)	**Builtin Functions**
and power users who wish	How Bash compares to PowerShell	Wide range of builtins
to use the command line to	**Control Flow**	Regular expressions
interact with the system	if	readline
and to create shell scripts	select / case	**Alias**
to automate such activities	for	Listing aliases
	break / continue	Role of aliases
	Variables	alias and unalias functions
	set and unset for local variables	**Error handling**
	env	Error handling in scripts
Prerequisites	Export to child processes	Signal handling
Some previous experience	**Processes and Command Line**	trap
of shell usage required.	Command pipeline	**File Handling**
	Command line arguments	How files are represented in Bash
General knowledge of the	Exit code and results	general file I/O
operating system	Process architecture of executing Bash	umask
command layout is useful.	exec command	read
	$$ - process id of process executing shell	**Scripting Best Practices**
	Functions	Small chunks of script
	Parameters to functions	Pay attention to debugging
	Viewing declared functions	Resilience to error situations
	Nested functions	Similar to/different from regular
	Job control	programming
	Configuring the Environment	**Project**
	.bashrc	Explores the scripting needed for
	Start up scripts - profiles	automated workflow for a non-trivial
	Login shell	enterprise scenario