WinDbg Cheat Sheet (user mode only)

Help Commands

`?`	Help on Debugee commands
`.help`	Help on Debugger commands
`.hh command`	Open WinDbg's help for this command

Execution Control

`restart`	Stop and restart execution
`t (F11)`	Step into (trace)
`p [count] (F10)`	Step over
`gu (Shift-F11)`	Step return
`g (F5)`	Continue (go)
`pa address`	Run to address
`(Ctrl-Break)`	Break

Breakpoints

`bl`	List breakpoints
`bp [addr] ["script"]`	Set a breakpoint
`bp`	Set breakpoint at current instruction
`bp addr`	Set breakpoint at specified address
`bp addr "script"`	Set a breakpoint and run script when hit
	`bp 403250 ".echo BP hit;g"`
`bc #`	Clear a breakpoint
`bc *`	Clear all breakpoints
`bd #`	Disable a breakpoint
`bd *`	Disable all breakpoints
`be #`	Enable a breakpoint
`be *`	Enable all breakpoints
`ba [rwe] [size] addr`	Set a breakpoint on memory access
	Size can be 1, 2, or 4
`ba r addr`	Break on read acces
`ba w addr`	Break on write access
`ba e addr`	Break on execute access

Listing Modules

`lm [olfv]`	List all modules
`lm o`	List only loaded modules
`lm l`	List modules with symbol information
`lm f`	List all modules and their full image path
`lm v`	List all modules and be verbose
`lm a address`	Display the module that contains *address*
`lm m pattern`	Find module name, can contain wildcard
`lm M pattern`	Find image path, can contain wildcard

Symbols

`.reload /f`	Reload all symbols
`ld module`	Load symbols for a module
`ld *`	Load symbols for all modules
`ln address`	Find nearest symbol to address
`x module!symbol`	Display the symbols that match the specified pattern, can contain wildcard

Unassembly

`u[ub] address [L#]`	Unassemble from memory
`uu addr`	Disassembly continues past read error
`ub addr`	Determine range by counting backwards
`u addr L#`	Set the number of instructions to disassemble

Memory

`d* [/c#] addr [L#]`	Display the contents of memory
`db addr`	Byte values (1 byte) and ASCII characters
`dw addr`	WORD values (2 bytes)
`dW addr`	WORD values (2 bytes) and ASCII characters
`dd addr`	DWORD values (4 bytes)
`dc addr`	DWORD values (4 bytes) and ASCII characters
`dq addr`	QWORD values (8 bytes)
`da addr`	ASCII string up until first null byte
`du addr`	Unicode string up until first null byte
`df addr`	Single-precision float numbers (4 bytes)
`dD addr`	Double-precision float numbers (8 bytes)
`d* /c# addr`	Set the number of columns to use in the display
`d* addr L#`	Set the length of output

Type Information

`dt [-r] name`	Display variable or data type information
`dt -r name`	Recursively dump the subtype fields
`dt name addr`	Specify the address of the struct
`dt ntdll!_TEB @$teb`	Use @ to specify a register
`dt name field`	Specify the field to display

Evaluate Expressions

`? expr`	Evaluates an expression. Examples:
	`? 77269bc0 - 77231430`
	`? 77269bc0 >> 18`
	`? 41` (to see value in decimal)
`?? expr`	Evaluates C++ expression. Example:
	`?? sizeof(ntdll!_TEB)`
`.formats expr`	Evaluate and show in multiple formats

Registers

`r`	Display all registers and their values
`r reg`	Display a single register and it's value
`r reg=value`	Set the register to a specific value

Prefixes

`0x`	Hexadecimal (default)
`0n`	Decimal
`0y`	Binary