WinDbg Cheat sheet

Credits: Huge thanks to Xeno Kovah & taesazad

Setting up

- Download the SDK from here.
- Symbol Server Configuration
  - What is actually symbol server?
  - The Symbol Server is a file server that stores your debug symbols centrally, on a server, rather than on each developer's system. Then, you can point Windbg (or your debugger of choice) to the Symbol Server to resolve symbol names. Everyone can share the same server. Microsoft even makes a publically available server available for Windows symbols. (requires the debug build of Windows).
  - Source: S.O FTW!
  - How to configure Symbol Server ?
  - Go to "File" then you will find "Symbol File Path" then enter
  - cache*c:\SymbolCache;srv*https://msdl.microsoft.com/download/symbols
- Setting up the necessary Windows
  - Watch
    - This window is used to track those global variables, local variables and registers we are interested to look into and notice the change in values inside them, as we step in.
  - Registers
    - This window is used to only track registers, and change in values inside them as we continue our debugging process.
  - Memory
    - This window is used to view the memory ana access memory by virtual address and physical address.
  - Disassembly
    - This window is used to view the disassembly of the executable which you want to debug.
  - Call Stack
    - This window is used to view the call stack of the program, call stack in simple words is the view of the function that is currently on the stack.
  - Locals
    - This window is used to view the local variables and how they change during the entire debugging process.
  - Command Window
    - This window is used to display the results of execution of multiple debugger commands.
- Loading and unloading an executable
  - To load an executable inside the debugger, go to 'File' & select "Open Executable"
  - OR
  - CTRL + E
  - To unload an executable from the debugger click on the "stop debugging"
  - OR
  - Enter 'q' in the command window.
- Clear Screen & Help
  - .cls
  - .help

User mode Debugging

- Symbol Reload & Fix
  - !sym : Displays state if symbol reloading
  - !d : Loads symbols for modules
  - .sympath : Sets symbol search path
  - .reload : Reloads symbol info
  - .reload /f : Reloads symbol info with force
  - .reload /f /v : Reloads symbol info with force and with verbose mode
- Breakpoints
  - bp $exentry : Sets a breakpoint on the very first instruction of the executable.
  - bl : Lists all the breakpoints.
  - bc : Clear a breakpoint
  - bc* : Clear all breakpoints
  - be : Enable a breakpoint
  - be * : Enable all breakpoints
  - bd : Disable a breakpoint
  - bd * : Disable all breakpoints
  - bp [Address] : Set's a breakpoint at a certain address.
  - bp `sourcefile:linenumber` : Sets breakpoint at a specified source code
  - bp [symbolname] : Sets a breakpoint on a symbol name.
  - ba : Sets a processor breakpoint
  - br : Renames breakpoint number.
- Registers
  - r : Lists all the registers
  - r [regX] : List the register X and its value
  - r regX = Somvalue : Assigns "Somevalue" to registerX.
  - rF : Lists all floating point registers
  - rX: Lists all XMM registers
- Call Stack & Frame
  - k : Shows the call stack
  - !findstack SymbolX: Displays the stack which contain the symbol or module
  - .frame: Displays the current frame
- Memory (View | Modify)
  - d * (display memory *)
    - * = a (ASCII)
    - * = b (Byte)
    - *= d(DWORD)
    - *= f(Float)
    - *=q(QWORD)
    - * = u(UNICODE)
    - * = w(Word)
  - e * (edit memory)
- Stepping & Tracing
  - In: When the next statement to execute reaches a method call, don't execute the method as a whole, but rather, execute the first line of that method and stop
    - p
    - p [Count]
    - pt : Steps to the next return
    - pc : Steps to the next call
    - pa: Steps to a certain address
  - Over: When the next statement to execute reaches a method call, execute the method as a whole and stop.
    - g
    - gu
  - Out: Finish off executing the callee's code and stop when execution returns to the caller
  - t : Performs single trace.
- Run to cursor : Click onto the line in the source view > Select "Run to Cursor".
- Attach/Detach to/from Running Process
  - .attach [PID]
  - .detach
  - .restart