64-bit or 32-bit machine / ps host / process

Various ways of determining if the system is 64-bit or 32-bit

[Source]

```
# Get the path where powershell resides.  If the caller passes -use32 then
# make sure we are returning back a 32 bit version of powershell regardless
# of the current machine architecture
function Get-PowerShellPath() {
    param ( [switch]$use32=$false,
            [string]$version="1.0" )

    if ( $use32 -and (test-win64machine) ) {
        return (join-path $env:windir "syswow64\WindowsPowerShell\v$version\powershell.exe")
    }

    return (join-path $env:windir "System32\WindowsPowerShell\v$version\powershell.exe")
}



# Is this a Win64 machine regardless of whether or not we are currently
# running in a 64 bit mode
function Test-Win64Machine() {
    return test-path (join-path $env:WinDir "SysWow64")
}

# Is this a Wow64 powershell host
function Test-Wow64() {
    return (Test-Win32) -and (test-path env:\PROCESSOR_ARCHITEW6432)
```

```
}

# Is this a 64 bit process
function Test-Win64() {
    return [IntPtr]::size -eq 8
}

# Is this a 32 bit process
function Test-Win32() {
    return [IntPtr]::size -eq 4
}

function Get-ProgramFiles32() {
    if (Test-Win64 ) {
        return ${env:ProgramFiles(x86)}
    }

    return $env:ProgramFiles
}

function Invoke-Admin() {
    param ( [string]$program = $(throw "Please specify a program" ),
            [string]$argumentString = "",
            [switch]$waitForExit )

    $psi = new-object "Diagnostics.ProcessStartInfo"
    $psi.FileName = $program
    $psi.Arguments = $argumentString
    $psi.Verb = "runas"
    $proc = [Diagnostics.Process]::Start($psi)
    if ( $waitForExit ) {
        $proc.WaitForExit();
    }
}

# Run the specified script as an administrator
function Invoke-ScriptAdmin() {
    param ( [string]$scriptPath = $(throw "Please specify a script"),
            [switch]$waitForExit,
            [switch]$use32=$false )
    $argString = ""
    for ( $i = 0; $i -lt $args.Length; $i++ ) {
        $argString += $args[$i]
        if ( ($i + 1) -lt $args.Length ) {
            $argString += " "
        }
    }

    $p = "-Command & "
    $p += resolve-path($scriptPath)
    $p += " $argString"

    $psPath = Get-PowershellPath -use32:$use32
    write-debug ("Running: $psPath $p")
    Invoke-Admin $psPath $p -waitForExit:$waitForExit
}
```

Revision #2
Created 26 August 2024 15:53:14 by bluecrow76
Updated 26 August 2024 15:55:51 by bluecrow76