```
function Set-DRFRemoteRegistry
{
    <#
        .SYNOPSIS
            Function to set registry items on the local machine, or on remote computers

        .DESCRIPTION
            This function uses the .Net Framework to set registry values of differing types.
            It includes a Force option to make it build the registry path in case it does not
            already exist.

        .PARAMETER  ComputerName
            This a list of ComputerNames that the registry value should be set.

        .PARAMETER  Hive
            This is the registry hive to open.  Acceptable parameters are:
            "ClassesRoot"       Represents the HKEY_CLASSES_ROOT base key
            "CurrentConfig"     Represents the HKEY_CURRENT_CONFIG base key (this a pointer to
            HKLM\SYSTEM\CurrentControlSet\CurrentControlSet\Hardware Profiles)
            "CurrentUser"       Represents the HKEY_CURRENT_USER base key
            "DynData"           Represents the HKEY_DYN_DATA base key  (this is only used with
            Windows 95, 98, 98SE)
            "LocalMachine"      Represents the HKEY_LOCAL_MACHINE base key
            "PerformanceData"   Represents the HKEY_PERFORMANCE_DATA base key
            "Users"             Represents the HKEY_USERS base key

        .PARAMETER  RegKeyPath
            This is the registry path in a string

        .PARAMETER ValueName
            This is the name of the registry value to be set

        .PARAMETER ValueData
            This is the data to be stored

        .PARAMETER ValueType
            This is the type of data to be stored
            "String"        A regular string value
            "ExpandString"  A string value containing replaceable parameters surrounded with %
            symbols, i.e. %USERNAME%
            "Binary"        Data that should be stored as a binary piece of data.
            "DWord"         A 32bit integer to be stored
            "MultiString"   This value contains one or more strings
            "QWord"         A 64bit integer to be stored
            "Default"       This is the default value of a registry key where the value does
            not have an actual name

        .PARAMETER Force
            This switch parameter causes the function to build a complete registry path if it
            does not exist.

        .EXAMPLE
            Get-Something -ParameterA 'One value' -ParameterB 32

        .EXAMPLE
```

```
        Get-Something 'One value' 32

    .INPUTS
        System.String,System.Int32

    .OUTPUTS
        System.String

    .NOTES
        Additional information about the function go here.

    .LINK
        about_functions_advanced

    .LINK
        about_comment_based_help

#>

[CmdletBinding(SupportsShouldProcess = $true)]
param
(
    [Parameter(Position = 0, Mandatory = $false)]
    [string[]]$ComputerName = $Env:COMPUTERNAME,

    [Parameter(Position = 1, Mandatory = $true)]
    [ValidateSet("ClassesRoot", "CurrentConfig", "CurrentUser", "DynData", "LocalMachine",
    "PerformanceData", "Users")]
    [string]$Hive = "LocalMachine",

    [Parameter(Position = 2, Mandatory = $true)]
    [ValidateNotNullOrEmpty()]
    [string]$RegKeyPath,

    [Parameter(Position = 3, Mandatory = $true)]
    [ValidateNotNullOrEmpty()]
    [string]$ValueName,

    [Parameter(Position = 4, Mandatory = $true)]
    [ValidateNotNullOrEmpty()]
    $ValueData,

    [Parameter(Position = 5, Mandatory = $true)]
    [ValidateSet("String", "ExpandString", "Binary", "DWord", "MultiString", "QWord",
    "Default")]
    [string]$ValueType,

    [Parameter(Position = 6, Mandatory = $false)]
    [Switch]$Force
)

Write-Debug -Message "Entering foreach (computer in computername) loop"
ForEach ($Computer in $ComputerName)
{
    Write-Debug -Message "ForEach ($Computer in ComputerName) pass"
```

```
try
{
    $Reg = [Microsoft.Win32.RegistryKey]::OpenRemoteBaseKey($Hive, $Computer)
    Write-Debug -Message "Successfully able to open $hive on $computer"

}
catch
{
    Write-Error -Message "Unable to open remote registry hive for $Computer.  Please
    verify connectivity and permissions to retry"
    continue
}


Write-Debug -Message "Entering foreach loop to create full registry path if $force is
enabled"
foreach ($SubKey in $RegKeyPath.Split('\'))
{
    $RegName = $Reg.Name
    Write-Debug -Message "SubKey is $subkey, and current registry path is $RegName"

    if ($Reg.GetSubKeyNames -contains $SubKey)
    {
        Write-Debug "\$Reg contains $SubKey, opening that key"
        $Reg = $Reg.OpenSubKey($SubKey)
    }
    else
    {
        Write-Debug -Message "\$Reg does not contain $subkey, checking for Force"
        if ($Force)
        {
            Write-Debug -Message "\$Force enabled, continuing"

            try
            {
                Write-Debug -Message "Trying to create SubKey ($SubKey)"
                $Reg = $Reg.CreateSubKey($SubKey)
            }
            catch
            {
                $Reg.Close()
                Write-Error -Message "Unable to create $subkey in $RegName, please
                check your permissions or the Remote Registry service may not be
                running" -ErrorAction 'Stop'
            }
        }
        else
        {
            $Reg.Close()
            Write-Error -Message "Force not specified, and subkey ($subkey) does not
            exist in $RegName" -ErrorAction 'Stop'
        }
    }
}
```

```
        Write-Debug -Message "Registry either exists, or has been created.  Proceeding with
        setting the value"
        try
        {
            $Reg.SetValue($ValueName, $ValueData, $ValueType)
            Write-Debug -Message "Succesfully set the value $ValueName on $Computer"
        }
        catch
        {
            Write-Debug -Message "Unable to set the value $ValueName in $RegName on $Computer;
            Please validate your access & permissions and try again"
        }
        finally
        {
            $Reg.Close()
        }

    }
}
```