9/1/2020 -- Installing into a Windows 10 VM prepped to handle Docker   (currently updated to v2004)

1. ref: https://github.com/conioh/Docker-Windows-Linux
    1. **Install Ubuntu in WSL**   ref: https://youtu.be/xzgwDbe7foQ
        1. 'Turn Windows Features On or Off': Check 'Windows Subsystem for Linux'
        2. https://aka.ms/wslstore and install Ubuntu   9/2/2020 : installed Ubuntu 20.04.1 LTS , *see https://github.com/MicrosoftDocs/WSL/issues/662 saying 20.04 does not work with WSL1*
        3. Launch Ubuntu now and let it finish installing : **user/pw: uuuu / pppp**
        4. Can add an alias to the Windows user folder:
            ```
            $ cd
            $ nano ~/.bashrc
            $ alias winhome='cd "/mnt/c/Users/My Name/"'
            $ source ~/.bashrc      // Really needed?
            ```
        5. `$ sudo apt-get update`
        6. `$ sudo apt-get upgrade`

    2. **Install Docker Engine on Windows**   *https://github.com/conioh/Docker-Windows-Linux/blob/master/Install-Docker-Windows.md*
        1. Download https://repos.mirantis.com/win/static/stable/x86_64/docker-latest.zip (or a specific version if you prefer).
        2. Extract to `%ProgramFiles%\Docker`
        3. Add `%ProgramFiles%\Docker` to `%PATH%`.
        4. Run (asAdmin) `dockerd.exe --register-service` (can add `-G <somegroup>` so non-admins can use Docker too)
        5. Optional: Add a `C:\ProgramData\Docker\config\daemon.json` file. Perhaps at least add `"exec-opts":["isolation=process"]`. Example:
            ```
            {
                "debug": true,
                "exec-opts":["isolation=process"],
                "experimental": true
             }
            ```

    3. **Install Docker into WSL**   https://github.com/conioh/Docker-Windows-Linux/blob/master/Install-Docker-WSL.md
        1. ref: *https://docs.docker.com/engine/install/ubuntu/*   see also (older/newer version?) https://medium.com/@sh.tsang/installation-of-docker-3b18d9e70bea
            1. Update the apt package index and install packages to allow apt to use a repository over HTTPS
                1. `$ sudo apt-get update`
                2. `$ sudo apt-get install apt-transport-https ca-certificates curl gnupg-agent software-properties-common`
            2. ~~see issue WSL Ubuntu 20.04 - gpg: can't connect to the agent: IPC connect call failed~~
                1. ~~$ sudo add-apt-repository ppa:rafaeldtinoco/lp1871129~~
                2. ~~$ sudo apt update~~
                3. ~~$ sudo apt install libc6=2.31-0ubuntu8+lp1871129~1 libc6-dev=2.31-0ubuntu8+lp1871129~1 libc-dev-bin=2.31-0ubuntu8+lp1871129~1 -y --allow-downgrades --allow-change-held-packages~~
                4. ~~$ sudo apt-mark hold libc6~~
            3. (better workaround?) https://github.com/MicrosoftDocs/WSL/issues/662
                1. `$ sudo apt-get remove gpg`
                2. `$ sudo apt-get install gnupg1`
            4. Add Docker's official GPG key
                `$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -`
            5. Check if it worked
                `$ sudo apt-key fingerprint 0EBFCD88`
            6. Set up the stable repository.
                1. `$ sudo apt-get install software-properties-common`
                2. `$ sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"`
            7. Install Docker Engine
                1. `$ sudo apt-get update`
                2. `$ sudo apt-get install docker-ce`
                    ```
                    DID IT WORK??   output:
                    Creating config file /etc/default/grub with new version
                    Setting up grub-gfxpayload-lists (0.7) ...
                    Processing triggers for install-info (6.7.0.dfsg.2-5) ...
                    Processing triggers for libc-bin (2.31-0ubuntu9) ...
                    Processing triggers for systemd (245.4-4ubuntu3.2) ...
                    Processing triggers for man-db (2.9.1-1) ...
                    Processing triggers for linux-image-unsigned-5.6.0-1023-oem (5.6.0-1023.23) ...
                    /etc/kernel/postinst.d/initramfs-tools:
                    update-initramfs: Generating /boot/initrd.img-5.6.0-1023-oem
                    cryptsetup: ERROR: Couldn't resolve device rootfs
                    cryptsetup: WARNING: Couldn't determine root device
                    grep: /proc/swaps: No such file or directory
                    W: mkconf: MD subsystem is not loaded, thus I cannot scan for arrays.
                    W: mdadm: failed to auto-generate temporary mdadm.conf file.
                    ```
                3. `$ sudo docker -v`   // Docker version 19.03.12, build 48a66213fe
            8. Verify that Docker Engine is installed correctly
                `$ sudo docker run hello-world`

        2. Expose the Docker daemon to the Windows host
            1. `$ sudo mkdir /etc/docker`
            2. Create a *`/etc/docker/daemon.json`* and put in:
                ```
                {
                    "debug": true,
                    "hosts": ["unix://", "tcp://127.0.0.1:2375"],
                    "experimental": true
                 }
                ```

        3. Start Docker Daemon in Ubuntu during Windows Host Startup
            1. Scheduled Task: ***`wsl.exe -u root -- service docker start`***

    4. **Docker Context**
        1. ***`docker context create wsl --docker host=tcp://localhost:2375`***
        2. NOTE: will now need to add `` `-c wsl` `` to most all docker commands

2. Starting Info for the daemon (??)
    1. The default location of the configuration file on Windows is *`%programdata%\docker\config\daemon.json`*.
       The ***`--config-file`*** flag can be used to specify a non-default location.
    2. use single global utility VM (better performance)
       `--storage-opt lcow.globalmode=false`
    3. size to use when creating the sandbox which is used for containers. Defaults to 20G
       `--storage-opt size=190G`
    4. This is a full example of the allowed configuration options on Windows:
        ```
        {
            "authorization-plugins": [],
            "data-root": "",
            "dns": [],
            "dns-opts": [],
            "dns-search": [],
            "exec-opts": [],
            "experimental": false,
            "features":{},
            "storage-driver": "",
            "storage-opts": [],
            "labels": [],
            "log-driver": "",
            "mtu": 0,
            "pidfile": "",
            "cluster-store": "",
            "cluster-advertise": "",
            "max-concurrent-downloads": 3,
            "max-concurrent-uploads": 5,
            "shutdown-timeout": 15,
            "debug": true,
            "hosts": [],
            "log-level": "",
            "tlsverify": true,
            "tlscacert": "",
            "tlscert": "",
            "tlskey": "",
            "swarm-default-advertise-addr": "",
            "group": "",
            "default-ulimits": {},
            "bridge": "",
            "fixed-cidr": "",
            "raw-logs": false,
            "allow-nondistributable-artifacts": [],
            "registry-mirrors": [],
            "insecure-registries":
        }
        ```
    5. .

3. References
    1. See here for the background on the trouble of running Linux and Windows Docker modes together like the Experimental Feature promised....
       https://github.com/docker/roadmap/issues/78
    2.