WinDocks

Working with and supporting Windocks

Windocks is a modern, open data delivery platform based on an independent port of Docker's source to Windows and supports all editions of Windows 8, Windows 10, Windows Server 2012, and Windows Server 2016, on public clouds or private infrastructure. Windocks supports containers with .NET 4.5 (with IIS), all versions and editions of SQL Server 2008 onward, and Java and other open source projects.

In addition to being a container engine, Windocks is also a general purpose SQL Server database cloning solution. Windocks SQL Server clones support use with any SQL Server application environment, including Windocks SQL Server containers, as well as Microsoft's SQL Server containers, and conventional servers and workstations. Inquire with support@windocks.com for more information on use of database clones with Microsoft containers and conventional instances.

Docker Containers and Images

Docker combines application packaging with support for application multi-tenancy. As a port of Docker's source to Windows, Windocks delivers a faithful port that operates with subset of standard Docker commands. The technology involves the following terms:

Container: is an application process (or processes), that can include data.

Dockerfile: is a plain text configuration file that defines a container and associated image

Docker engine or daemon: is the supervisory privileged process that manages containers. Windocks runs as a Windows Service.

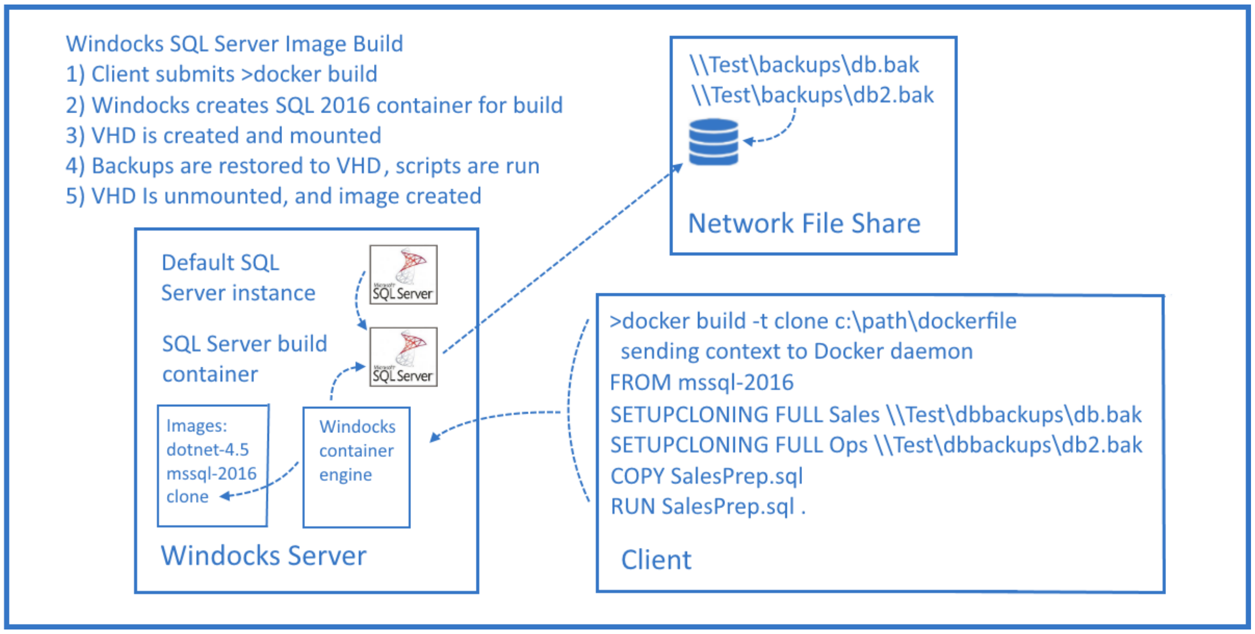
Image: is an immutable definition of a container. Windocks supports a number of standard "base" images, and the creation of custom images.

Image Registry: a cloud or server hosted storage and distribution service for images. Windocks does not support the Dockerhub registry, or a private registry at this time. Images are supported on each Windocks host, but are easily portable.

Docker Toolbox: refers to the Docker clients for Mac, Linux, and Windows. Windocks does not support the most current Docker Toolbox, but provides standard Docker clients (details below).

The Docker workflow involves use of Dockerfiles which reside on a client. During a docker build, the dockerfile **and all files in the same directory** are copied to the host for processing. On the host, files identified in the Dockerfile are copied into the container and run. The Dockerfile may also specify network resources that are used to build SQL Server clone images.

In the example below, a client submits a **>docker build** with a Dockerfile that specifies use of SQL Server 2016, network hosted backups, and a SQL Server script that is run on the sales database. Windocks builds the custom image by first creating a container, while also creating and mounting a Virtual Hard Drive (VHD). The backups are restored to the VHD, and the scripts are run, and then the VHD is un-mounted, and the custom SQL Server image is created.


Windocks SQL Server Image Build
1) Client submits >docker build
2) Windocks creates SQL 2016 container for build
3) VHD is created and mounted
4) Backups are restored to VHD, scripts are run
5) VHD Is unmounted, and image created
\\Test\backups\db.bak
\\Test\backups\db2.bak
Network File Share
Default SQL Server instance
SQL Server build container
Images: dotnet-4.5 mssql-2016 clone
Windocks container engine
Windocks Server
>docker build -t clone c:\path\dockerfile
sending context to Docker daemon
FROM mssql-2016
SETUPCLONING FULL Sales \\Test\dbbackups\db.bak
SETUPCLONING FULL Ops \\Test\dbbackups\db2.bak
COPY SalesPrep.sql
RUN SalesPrep.sql .
Client

Note: the syntax for remote Docker client command is simplified for readability. A remote Docker build command illustrated would be properly be:

```
>docker –H=tcp://windocks.host.ip.address:2375 build –t clone c:\path\dockerfile
```

Windocks Overview:

Windocks uses local host resources to deliver containers, including a locally installed default or named SQL Server instance for creation of SQL Server containers. SQL Server licenses allow unlimited named instances on licensed SQL Server installations, thus allowing use of SQL Server containers without additional SQL Server licenses.

Windocks uses standard Docker client software, and installs the Docker client on the system path of the Windocks host. The Windows client is located in the \windocks\client directory, and can be copied to other Windows systems as desired. Mac and Linux clients are available at:

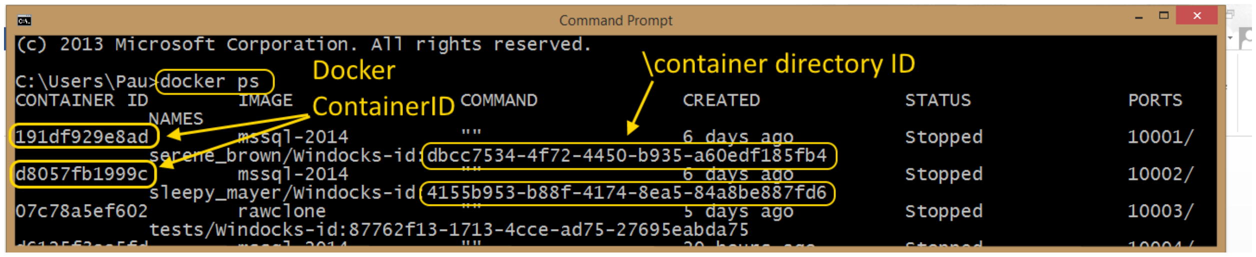
Docker Mac client

Docker Linux client

Windocks includes a web application that simplifies the use of containers for Developers and Testers. The Windocks installation includes a number of directories:

\config: System level configuration options are set in the node file. Refer to the **Windocks Installation and Configuration Guide** for a detailed review of configuration options.

\containers: includes a list of containers on the host, as well as a list of assigned ports, and a mapping file that maps Windocks containers to the Docker assigned container ID. The container IDs presented are easily found by examining the details presented on the **>docker ps** command. Docker containerIDs are used to start, stop or commit containers. The Windocks container IDs are used to examine navigate container logs and file directories. Note, the directory used to store containers is configurable, refer to the **Windocks Installation and Configuration** guide for details.



```
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Pau>docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
          NAMES
191df929e8ad        mssql-2014          ""                  6 days ago          Stopped             10001/
          serene_brown/Windocks-id:dbcc7534-4f72-4450-b935-a60edf185fb4
d8057fb1999c        mssql-2014          ""                  6 days ago          Stopped             10002/
          sleepy_mayer/Windocks-id:4155b953-b88f-4174-8ea5-84a8be887fd6
07c78a5ef602        rawclone            ""                  5 days ago          Stopped             10003/
          tests/Windocks-id:87762f13-1713-4cce-ad75-27695eabda75
```

\data: lists the mount points generated for SQL Server containers created with database clones. This directory can also be used by conventional SQL Server instances when mounting the cloned database files.

\dbbackups: includes sample database backups for building SQL Server images with cloned databases. VHD's built with the backups are co-located in this directory. Most users will choose to locate a similar directory on a separate drive on the Windocks host, or use network attached file shares. The VHDs will be built in the first directory referenced in the Dockerfile. See **Getting Started with SQL Server Containers and Database Clones** for further reading.

\log: includes installation related logs, as well as a Platform log file that is used in delivery of technical support. See the **Technical Support** section of this document for further information.

\samples: includes a set of Powershell scripts, and a number of sample Dockerfile samples including .NET applications and databases. These examples can be modified to suit your needs.

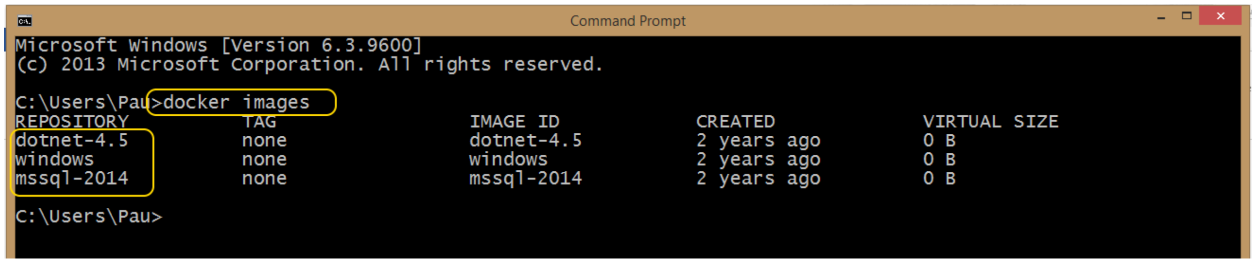
Version: records the Windocks release and version installed.

Key.txt: for Windocks monthly subscribers, the license key is included in the key.txt file located in the Windocks root directory. The Windocks installation and associated license key can be moved to new systems as needed, but only the latest installation will operate with the license key.

Windocks SQL Server containers:

Windocks runs as a Windows Service. During installation Windocks configures a SQL Server default or named instance for container support. The selected instance will be set to a "manual – off" following the installation. Windocks starts and stops the default instance as needed during the container creation and management process. The instance can be turned on at any time, without impacting existing containers, <u>but should only be used if the Windocks Service is turned off</u>. Additional SQL Server instances of the same release can co-exist on the host without issue, and Windocks can support multiple SQL Server images on the same host without issue (ie., configure support for containers using multiple SQL Server installations on the host).

Following installation the Windocks normally presents three base images, including **dotnet-4.5, windows,** and **mssql-20XX** (depending on what release of SQL Server is present on the host). This is confirmed by running a **>docker images** command (refer to the **Windocks Command Line Reference** for a complete review of Docker commands supported by Windocks).



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Pau>docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
dotnet-4.5          none                dotnet-4.5          2 years ago         0 B
windows             none                windows             2 years ago         0 B
mssql-2014          none                mssql-2014          2 years ago         0 B

C:\Users\Pau>
```

The base SQL Server image delivers containers that include only system databases from the default instance.   This behavior can be adjusted by modifying the node file entry: **COPY_DEFAULT_INSTANCE_DATABASES=0**. The node file is located in \windocks\config directory.   See the **Windocks Installation and Configuration Guide** for a complete review of configuration options.

Each SQL Server container includes a copy of the default instance Master database, so user logins configured for the default instance are inherited by containers. Workloads that require custom configuration of the Master database, should include SQL Server scripts included in the Dockerfile that defines the custom image.

Windocks SQL Server containers are clones of the default instance, and operate as a standard SQL Server named instance with a few exceptions:

- SQL Server containers do not operate as a Windows Service.   As a result, following a reboot of the host, the containers will be in an “off” state. They can be easily restarted.
- SQL Server containers are delivered with Named Pipes support disabled.   As a result, the instances are accessed using a local loopback address (127.0.0.1,1000X), with a comma separator for the port, or from a remote system the full IP address.
- SQL Server containers are not supported with SQL Agent at this time, work is underway to support SQL Agent in the future.

SQL Server containers support two methods for working with data, including databases located in the container’s private file system. Alternatively, databases can be mounted on external locations. Refer to other **Windocks Guides** for detailed instructions on the use of these methods. The trade-offs between these methods is easily summarized:

In-container data: works well for data sets of up to 2 or 3 GB, as each container requires a full byte copy of the data when built, and the data resides in the container’s private file system. Data does not persist when the container is deleted, but this approach does support easy creation of custom images.   See **Getting Started with SQL Server Containers and in-container data** for further information.

Mounted data: data can be mounted that is local to the Windocks host, or over network attached file shares, or Storage Array Networks.   SANs are popular for fast provisioning of snapshots and clones.   Data persists when containers are removed, but mount points do not persist in custom images.   As a result, containers are built individually using Dockerfiles.

Database Clones: Windocks includes Windows based SQL Server database cloning. A custom image is built with a full byte copy of the data environment, and containers are delivered with cloned databases. Clones are writable, and a Terabyte class database is delivered in 30 seconds. The cloned databases can be used by conventional SQL Server instances (they are not limited to use by only containers). See **Getting Started with SQL Server Containers and Database Clones** for more information.

Working with SQL Server containers can involve many topics, such as integration with DNS. Refer to https://windocks.com/blog-2/docker-windows-containers-and-DNS Applying SQL client aliases is another alternative.

Windocks dotnet-4.5 Image:

Windocks includes a standard image that includes .NET 4.5 with IIS.

Windocks windows Image:

Windocks includes an empty "container" designed to support server-side executables. This is the base image used to support creation of Java with Tomcat, Java with Jetty, as well as Nginx, and node.js images. Samples are available, but do not ship as part of the standard install to keep the basic Windocks install package to a manageable size.

Trouble Shooting Tips

The most common problem associated with use of Windocks is the use of incorrect Docker client syntax. Please refer to the detailed instructions provided in the **Windocks 2.2 Command Line Reference.**

Problems with remote access? Ensure the Windows Firewall is not preventing inbound traffic to the container ports, and the Docker daemon port of 2375.

Confirm the Windocks service is running, by using >docker ps, or open "local services" to check the status of the Windocks Service.

Working with Windocks Technical Support

Windocks support is available by emailing support@windocks.com during standard business hours, 9 am to 5 pm, Pacific time. We commonly meet online with screen sharing enabled to help resolve problems. Please provide the following information to assist in the resolution of your problem.

If a problem occurs during installation, provide the log files in \windocks\log, and everything in in the \windocks\install directory.

For problems involved in using Windocks, provide the \windocks\log\platform log file, and a screen shot of the Docker client command prompt.

For problems associated with SQL Server containers, please note the specific behavior, navigate to the SQL Server container involved and it's associated log directory (\mssql\log), and attach the controllog, stdout, and stderr logs. Also attach a screen shot of SQL Server Management Studio when trying to connect to the container.

For problems with .NET containers, navigate to the container, and to the dotnet\log and attach the controllog, stdout, and stderr logs.

Windocks Release Notes

General notes:

1) Windocks assigns container ports within a range based on a configuration setting. Users, however, can assign ports outside of this assigned range.
2) Windocks 2.2 does not currently support the most current Docker Toolbox client. Use the Docker clients identified in this and other Windocks docs.
3) The Windocks web UI allows image names with Caps, while standard Docker clients do not. As a result, images can be created that are distinguished by the presence of absence of a capitalized letter.
4) The Windocks web UI supports Chrome and FireFox browsers, but not IE.
5) Users can work with containers using 2 or 3 digits of the DockerID, which is sufficient to achieve a unique match. Windocks will occasionally return an error indicating no such container ID exists. Repeat the command using the full DockerID and the command will succeed.
6) Windocks support the use of Docker exec, providing the ability to execute a command inside of the container, but Windocks currently lacks the ability to write the output to a shell.
7) The Windocks 2.2 web UI lacks meaningful error reporting. A user, for example, can assign a duplicate container name, and the container will fail to be delivered without feedback as to why the command failed.
8) Windocks 2.2 runs as a Windows Service and has changed the path to the local Docker client. As a result, existing PowerShell scripts will need to be updated to reflect the updated path to the local Docker client (\windocks\client).
9) Many organizations ask about methods for DNS integration. Several strategies are available, including IP port forwarding and SQL client aliases. Inquire with support@windocks.com for more information.
10) Windocks 2.2 includes a web UI file selector tool. This tool is only supported for use in building SQL Server images with cloned databases.
11) Sample PowerShell scripts are included in the \windocks\samples directory. The scripts should be checked to confirm that they reference the correct version of SQL Server.

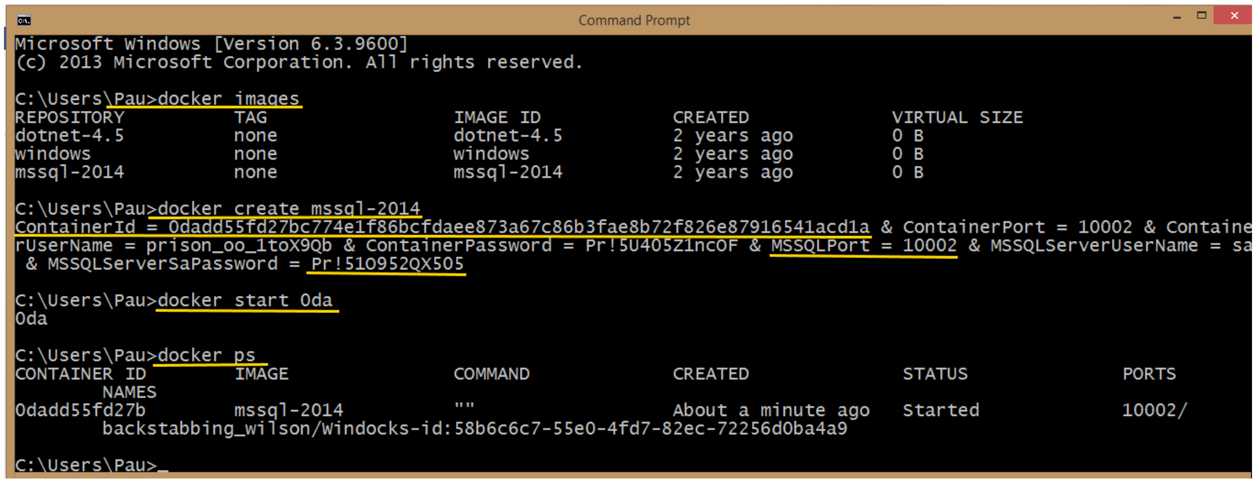
SQL Server notes:

1) The Windocks Service employs a default or named instance on the host for creation and management of SQL Server containers. DO NOT use this instance for other purposes without ensuring the Windocks Service is turned off.
2) Named pipes is disabled on SQL Server containers. Access containers using either the local loopback address (127.0.0.1), or the Windocks host IP address. In both cases the IP address is separated from the port with a comma. (127.0.0.1,1000X).
3) Windocks supports user assigned SQL sa passwords (>docker create –e SA_PASSWORD=”Pa44word!!”). A sa password that fails the password complexity requirement results in delivery of a non-functional container (the container will not start).

4) Following a forced system reboot all containers will be in a "stopped" state. Containers can be restarted, but SQL Server containers with cloned databases will start with the database in "recovery pending" mode.
5) Windocks become less responsive when building a large database clones. Large images should be built during off hours if possible.
6) The Windocks web UI includes a file sector tool for building SQL Server images with cloned databases. The tool is not supported for other purposes.
7) Windocks 2.2 supports user assigned SQL sa passwords, with docker create or run –d (>docker create –e SA_PASSWORD="Pa44word!!"). This environment variable, however, is not supported through Dockerfiles with the ENV command.
8) Do not use the special character "&" or "~" in assigned SQL sa passwords.
9) SQL Server images with cloned databases can be updated through use of Differential backups. The updated image depends on the continued presence of the parent image. Support for SQL log shipping and application of SQL Server scripts independently of the Differential backup is pending.

Get Started with Windocks using the CLI

With the Windocks service running, open a command prompt on the Windocks host and start using docker commands. The **>docker images** command provides a list of available images. Create your first container by using the **>docker create** command. Once the container is delivered, start the instance with **>docker start <containerid>.** The **>docker ps** command provides a list of the containers, and confirms the instance is running.



```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Pau>docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
dotnet-4.5          none                dotnet-4.5          2 years ago         0 B
windows             none                windows             2 years ago         0 B
mssql-2014          none                mssql-2014          2 years ago         0 B

C:\Users\Pau>docker create mssql-2014
ContainerId = 0dadd55fd27bc774e1f86bcfdaee873a67c86b3fae8b72f826e87916541acd1a & ContainerPort = 10002 & Containe
rUserName = prison_oo_1toX9Qb & ContainerPassword = Pr!5U405Z1ncOF & MSSQLPort = 10002 & MSSQLServerUserName = sa
 & MSSQLServerSaPassword = Pr!51O952QX5O5

C:\Users\Pau>docker start 0da
0da

C:\Users\Pau>docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS
        NAMES
0dadd55fd27b        mssql-2014          ""                  About a minute ago  Started             10002/
        backstabbing_wilson/Windocks-id:58b6c6c7-55e0-4fd7-82ec-72256d0ba4a9

C:\Users\Pau>_
```

The example illustrates the use of the docker commands on a local machine. For use on a remote client, the same commands are used, with some additional host details:

>docker -H=tcp://windocks.host.ip.address:2375 <command> <options> <args>

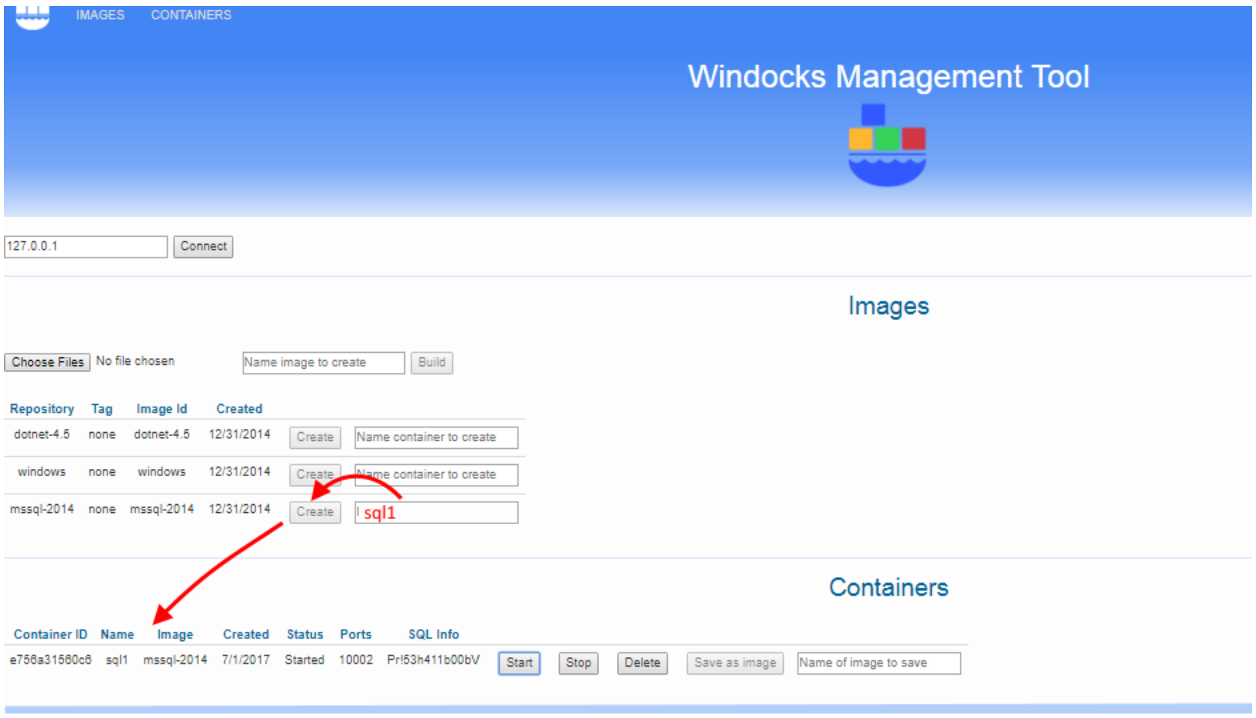
Be sure the host is configured to allow inbound traffic on ports 10000 – 10200, the Docker daemon port 2375, and the SQL Server port 1433.

Get started with the Windocks web UI

Windocks includes a web application that supports Chrome or Firefox browsers (IE is not supported). Open a Chrome or Firebox browser on the Windocks host, and direct the URL to “local host” Once the page resolves, enter the local loopback address (127.0.0.1), and “connect.” The available images (and containers) are presented.

Remote clients can use the web UI by opening a Chrome or Firefox browser, directing the URL to the IP address of the Windocks host, and once connected entering the IP address again in the IP address box, and “connecting.” Ensure the Windocks host firewall is configured to allow inbound traffic on the container ports (10000-10200), the daemon port of 2375, and the default SQL Server port of 1433.

Choose one of the SQLServer or dotnet images, assign a name, and “create.” The container will be presented below. Start the container, and it is ready for use.



Repository	Tag	Image Id	Created		
dotnet-4.5	none	dotnet-4.5	12/31/2014	Create	Name container to create
windows	none	windows	12/31/2014	Create	Name container to create
mssql-2014	none	mssql-2014	12/31/2014	Create	sql1

Container ID	Name	Image	Created	Status	Ports	SQL Info					
e756a31560c6	sql1	mssql-2014	7/1/2017	Started	10002	Pr!53h411b00bV	Start	Stop	Delete	Save as image	Name of image to save

WinDocks

Additional Resources:

1) For work with larger and more complex database environments, see the companion article on Getting Started with SQL Server containers with in container data
2) Windocks Command Line Reference
3) Windocks containers operate with DNS: https://windocks.com/blog-2/docker-windows-containers-and-DNS
4) To understand Windocks licensing options for organizations: https://windocks.com/files/WinDocks_Licensing_and_Support.pdf
5) Forthcoming Jenkins CI pipeline support?   Email: info@windocks.com
6) For information on working with multi-tier environments, including .NET see: https://windocks.com/lps/gitbuildtest
7) For technical support email: support@windocks.com
8) For information on how Windocks compares to Microsoft's new containers in Windows Server 2016: https://windocks.com/blog-2/Windows-Containers-Compared-Windocks-Microsoft