Presto User Guide

Last Updated: 9-Dec-2012

Quick Start Instructions

1. Copy the Presto Task Runner files (PrestoTaskRunner.zip) to a common location.
2. Install and run RavenDB (RavenDB-Build-960.zip).
3. Install and start the Self-updating service host (PrestoSelfUpdatingServiceHostInstaller.msi) on all servers that will be performing installations.
4. Install and run the Presto Dashboard (PrestoDashboardInstaller.msi)

Overview

What is Presto?

Presto is an automated deployment tool. It allows users to create deployment steps and then have servers update themselves when new versions of an app are ready. This allows for automated deployments to development, QA, and production environments.

The main reason to use Presto is when you have applications that need to be installed on many servers, in various environments, repeatedly, and each server requires a slightly different version of the app. That is Presto's sweet spot.

The Deployment Process

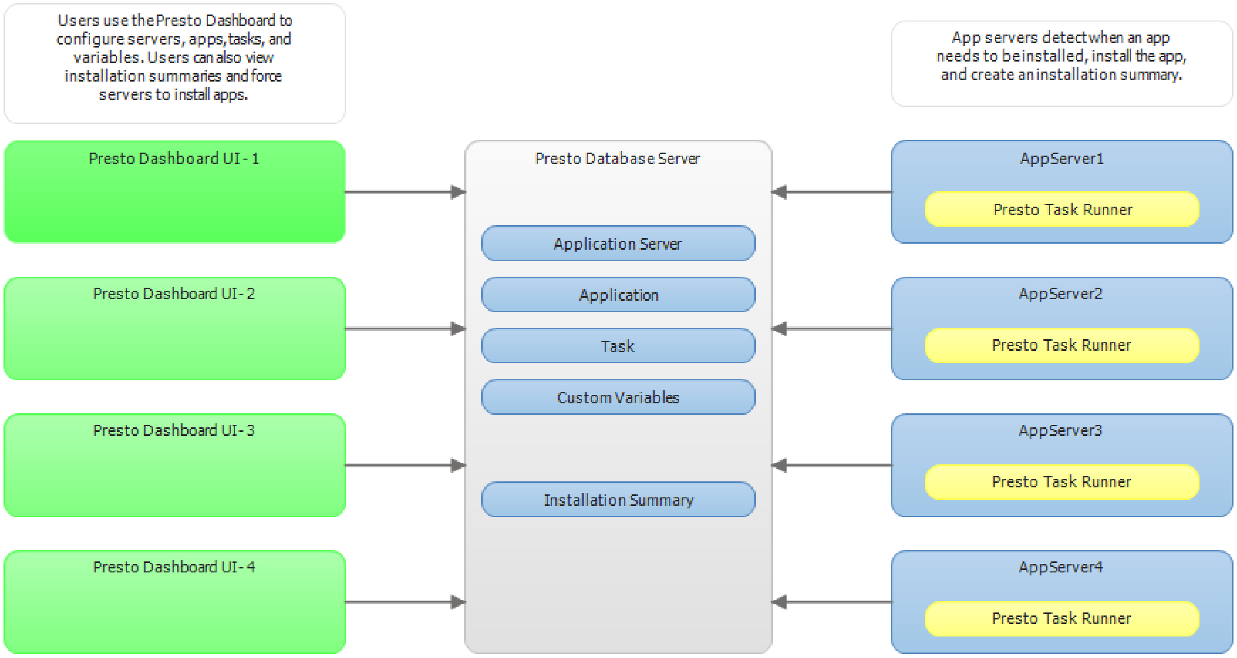
1. Within the Presto Dashboard, set up applications, servers, and variables.
2. When a new version of an application is ready to be deployed, copy the binaries to a common location.
3. Within the Presto Dashboard, initiate a deployment.
4. To deploy new versions of an application, go to step two.

What are the benefits of Presto?

- Deployment is automated; manual error is significantly reduced.
- Logging
    - When users add an application to a server
    - When users initiate a deployment
    - Each deployment: server, application, start time, end time, result/status
    - Installation details of each task when Presto installs an app
- Flexibility: Variables are created and used by applications and servers
- Ease of use
    - Deploy an application (or many apps) to a specific server
    - Deploy an application to all servers within an environment (i.e.: QA)
    - Presto will update itself on all servers when a new version is available

The Pieces

- **Presto Dashboard**: This is the application with which the user administers Presto. The user will create applications (and their tasks), set up servers, and create/use custom variables.
- **Presto Task Runner**: This is the actual deployment engine, running as a service on all servers. It polls the database to detect when an installation needs to occur, then performs the installation.
- **Database**: RavenDB (http://www.ravendb.net)



Users use the Presto Dashboard to configure servers, apps, tasks, and variables. Users can also view installation summaries and force servers to install apps.

App servers detect when an app needs to be installed, install the app, and create an installation summary.

What Presto Does

- Modify XML documents; usually configuration files
- Copy files
- Command prompt tasks. Examples:
    - Create a Windows service
    - Configure and start a windows service
    - Stop a clustered resource
    - Execute a batch file
    - Anything else that can be done at a command prompt

In a nutshell, Presto performs the three main tasks listed above. The power is in the third feature: command prompt tasks. Basically, anything that can be done at a command prompt can be automated with Presto.

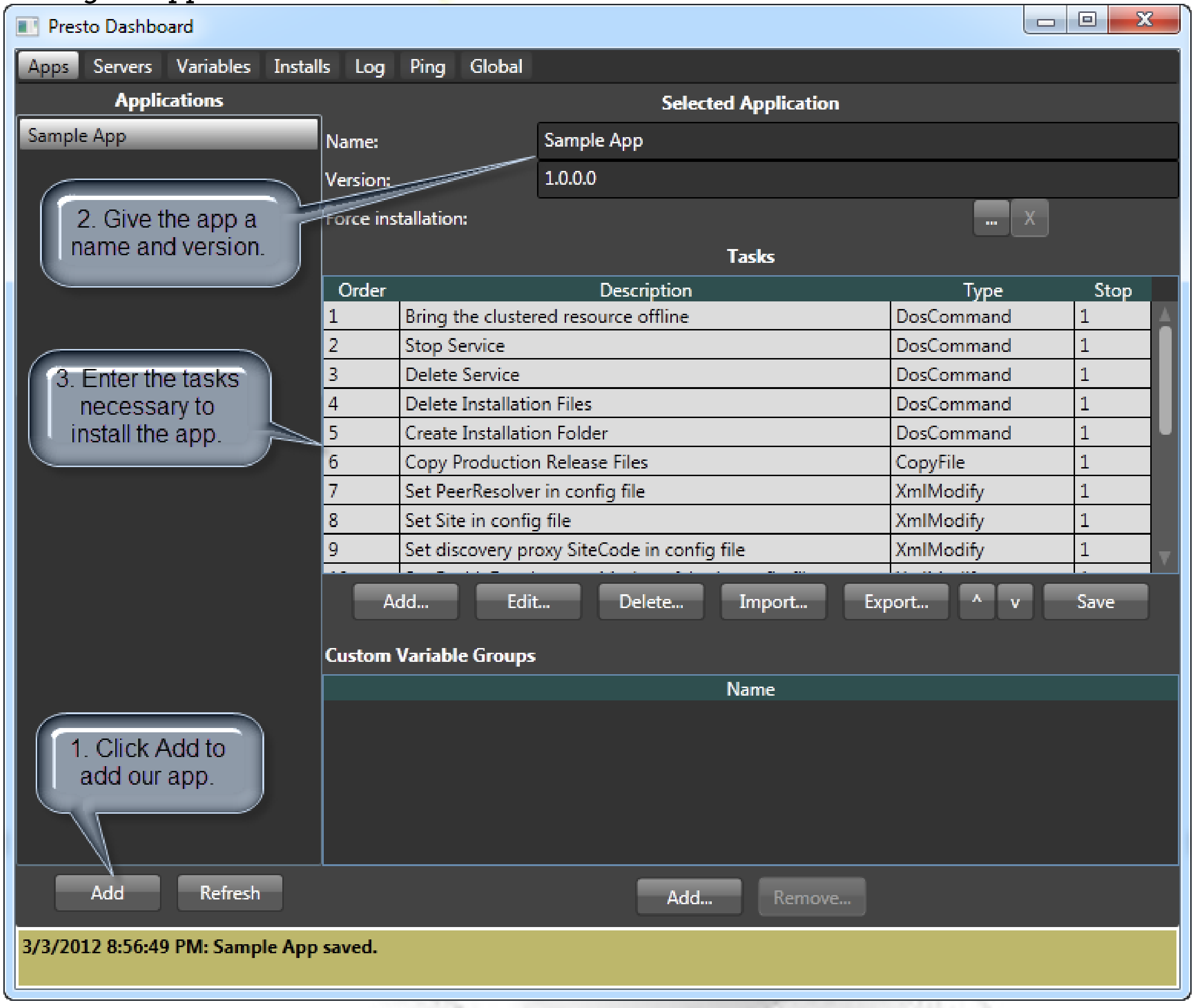
What Presto Doesn’t Do

- Deploy ClickOnce apps
- Deploy database schema and scripts

Sample Setup and Deployment – Screenshots

Note: These screenshots are here for overview reasons, to see how things come together. For a comprehensive explanation of each of the steps, see the remainder of this guide.

Adding the application in the Presto Dashboard:



Name: Sample App

Version: 1.0.0.0

Tasks

Order	Description	Type	Stop
1	Bring the clustered resource offline	DosCommand	1
2	Stop Service	DosCommand	1
3	Delete Service	DosCommand	1
4	Delete Installation Files	DosCommand	1
5	Create Installation Folder	DosCommand	1
6	Copy Production Release Files	CopyFile	1
7	Set PeerResolver in config file	XmlModify	1
8	Set Site in config file	XmlModify	1
9	Set discovery proxy SiteCode in config file	XmlModify	1

Custom Variable Groups

Name

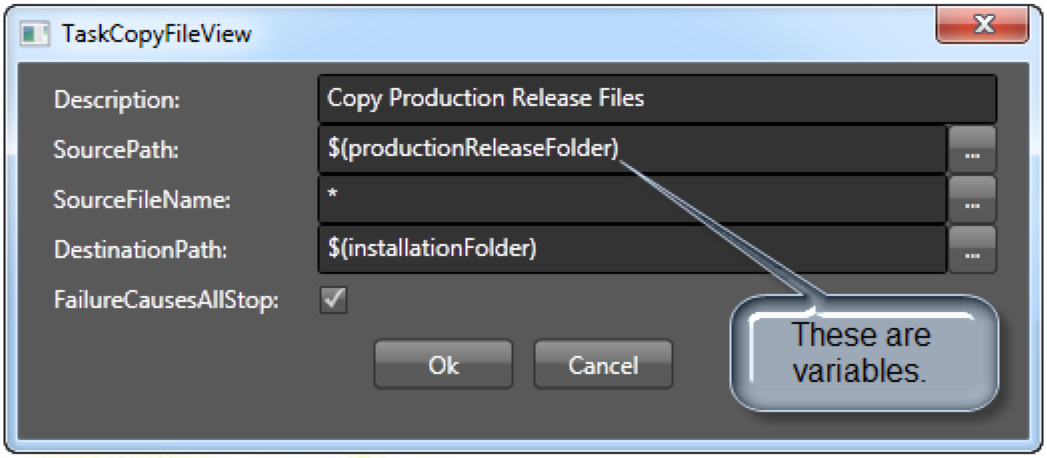
1. Click Add to add our app.

2. Give the app a name and version.

3. Enter the tasks necessary to install the app.

3/3/2012 8:56:49 PM: Sample App saved.

This is an example of one of the tasks (task 6 above, Copy Production Release Files). Here, we're simply copying the app's binaries from our production release folder to the installation folder:



Description: Copy Production Release Files

SourcePath: $(productionReleaseFolder)

SourceFileName: *

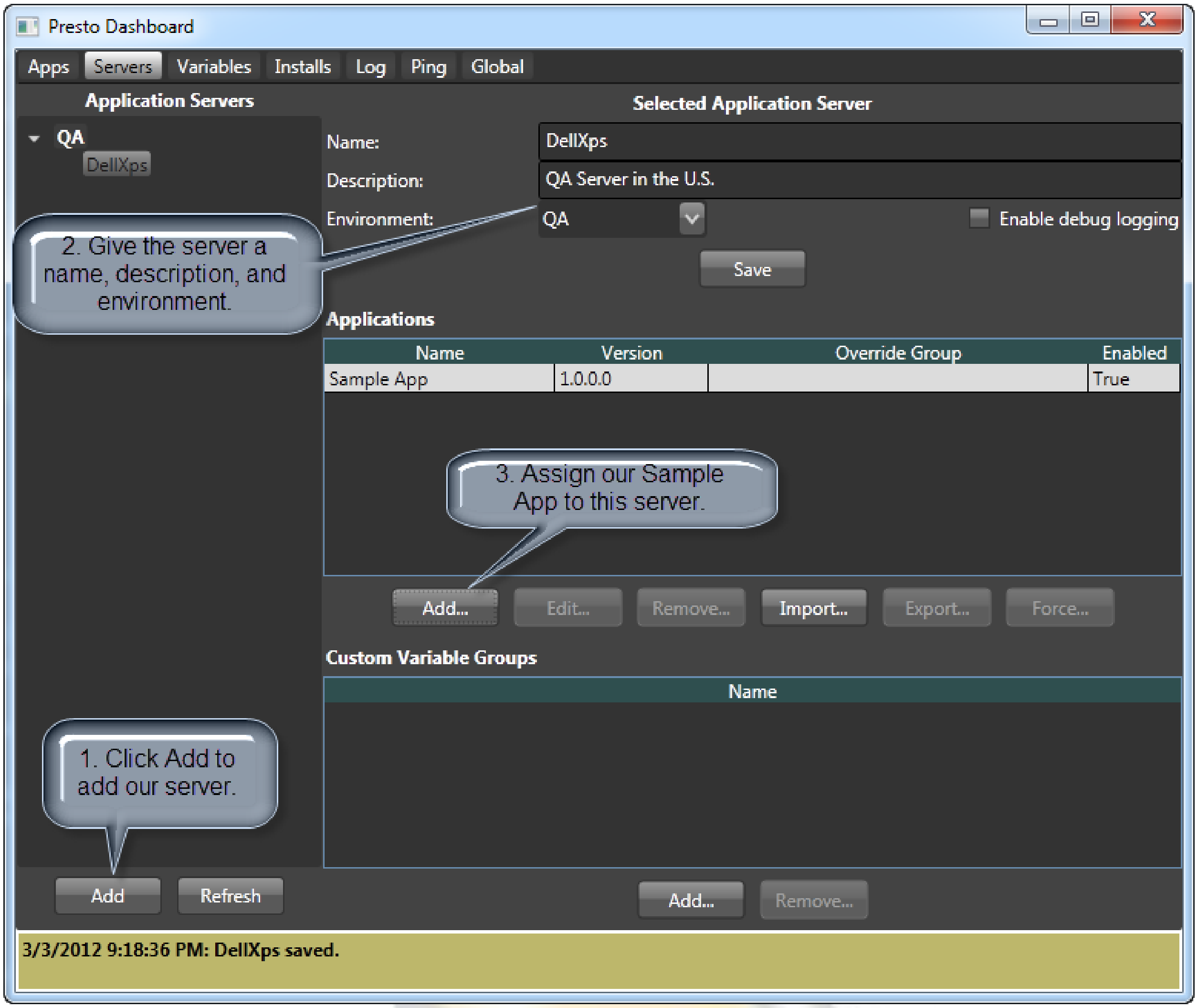
DestinationPath: $(installationFolder)

FailureCausesAllStop: ☑

These are variables.

Note: Don’t worry about the variables. We’ll get to those below. That variable will resolve to whatever is set as the production release folder; something like: \\SomeServer\ProductionBuilds\SampleApp\Release\.

Now that we have our application, let’s create a server on which to install it.



Name: DellXps

Description: QA Server in the U.S.

Environment: QA

□ Enable debug logging

Name	Version	Override Group	Enabled
Sample App	1.0.0.0		True

Name

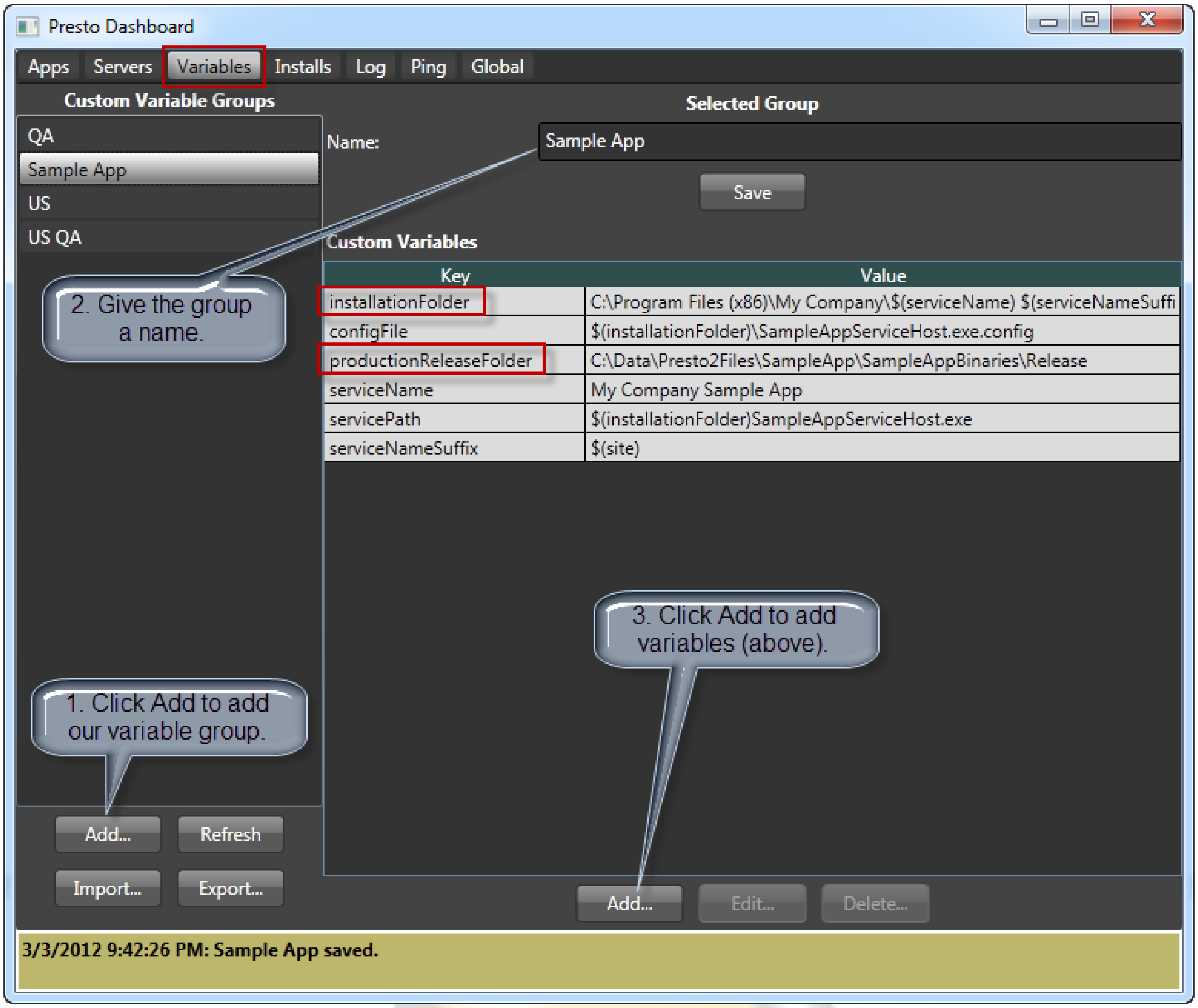
1. Click Add to add our server.

2. Give the server a name, description, and environment.

3. Assign our Sample App to this server.

3/3/2012 9:18:36 PM: DellXps saved.

Before being able to deploy our app, we need to add our custom variables.



Name: Sample App

Key	Value
installationFolder	C:\Program Files (x86)\My Company\$(serviceName) $(serviceNameSuffi
configFile	$(installationFolder)\SampleAppServiceHost.exe.config
productionReleaseFolder	C:\Data\Presto2Files\SampleApp\SampleAppBinaries\Release
serviceName	My Company Sample App
servicePath	$(installationFolder)SampleAppServiceHost.exe
serviceNameSuffix	$(site)

1. Click Add to add our variable group.

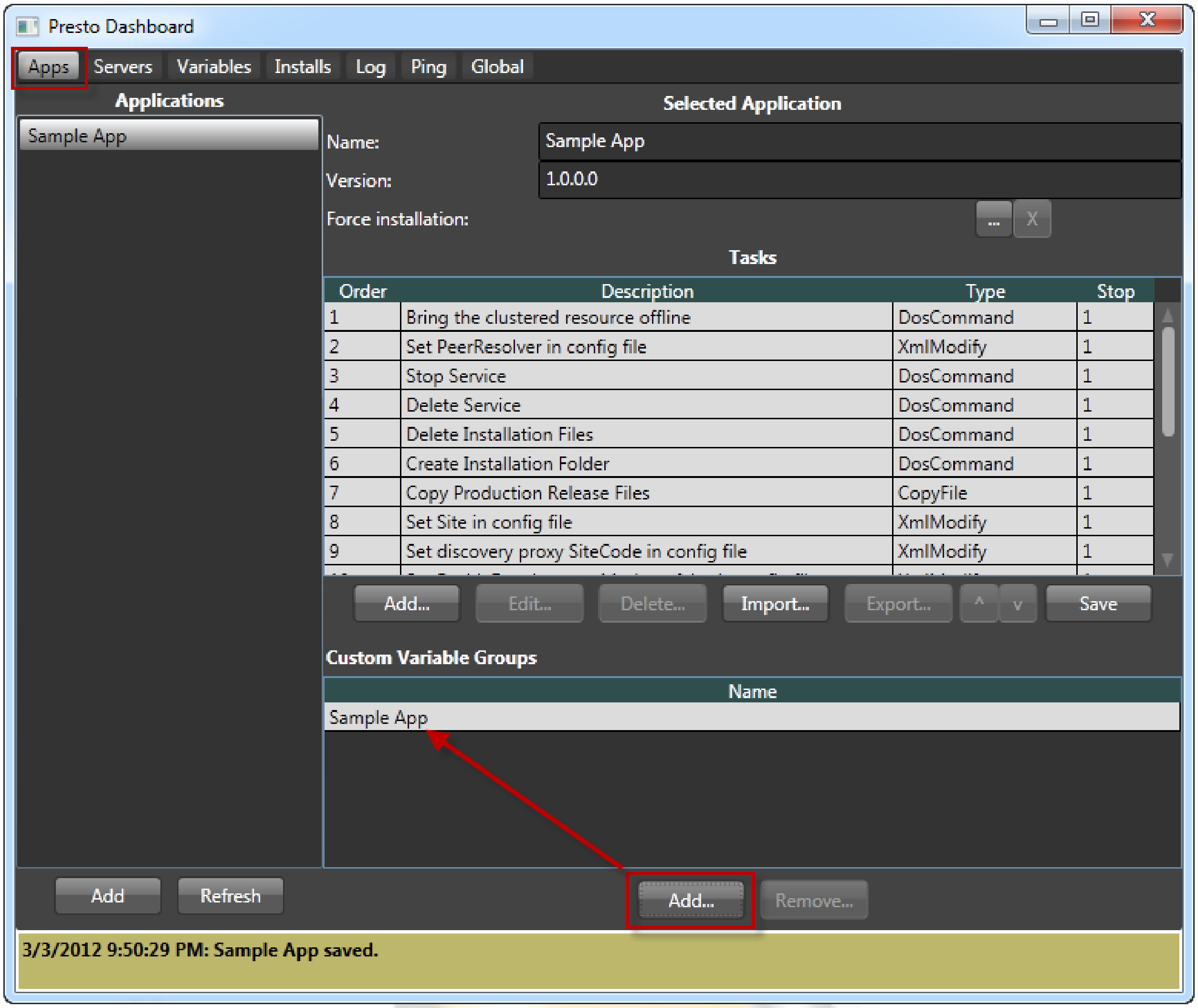
2. Give the group a name.

3. Click Add to add variables (above).

```
3/3/2012 9:42:26 PM: Sample App saved.
```

Remember our productionReleaseFolder and installationFolder variables from our task that we created a few minutes ago? This is where they're defined.

Now we just need to assign our custom variables to the app and server. Here's the app:



Name: Sample App

Version: 1.0.0.0

Force installation:

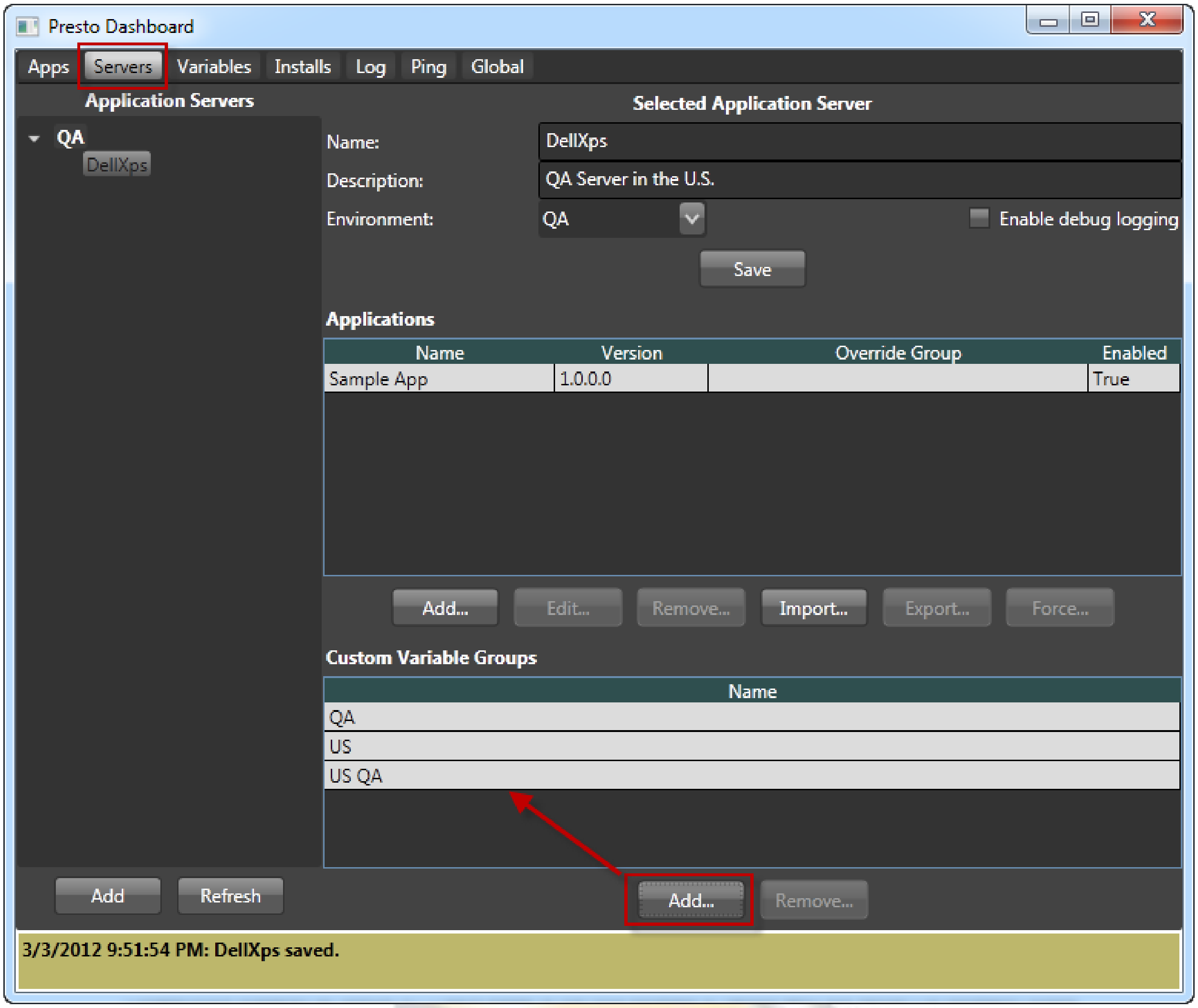
Tasks

Order	Description	Type	Stop
1	Bring the clustered resource offline	DosCommand	1
2	Set PeerResolver in config file	XmlModify	1
3	Stop Service	DosCommand	1
4	Delete Service	DosCommand	1
5	Delete Installation Files	DosCommand	1
6	Create Installation Folder	DosCommand	1
7	Copy Production Release Files	CopyFile	1
8	Set Site in config file	XmlModify	1
9	Set discovery proxy SiteCode in config file	XmlModify	1

Name
Sample App

3/3/2012 9:50:29 PM: Sample App saved.

And here's the server:



Name: DellXps

Description: QA Server in the U.S.

Environment: QA

□ Enable debug logging

Applications

Name	Version	Override Group	Enabled
Sample App	1.0.0.0		True

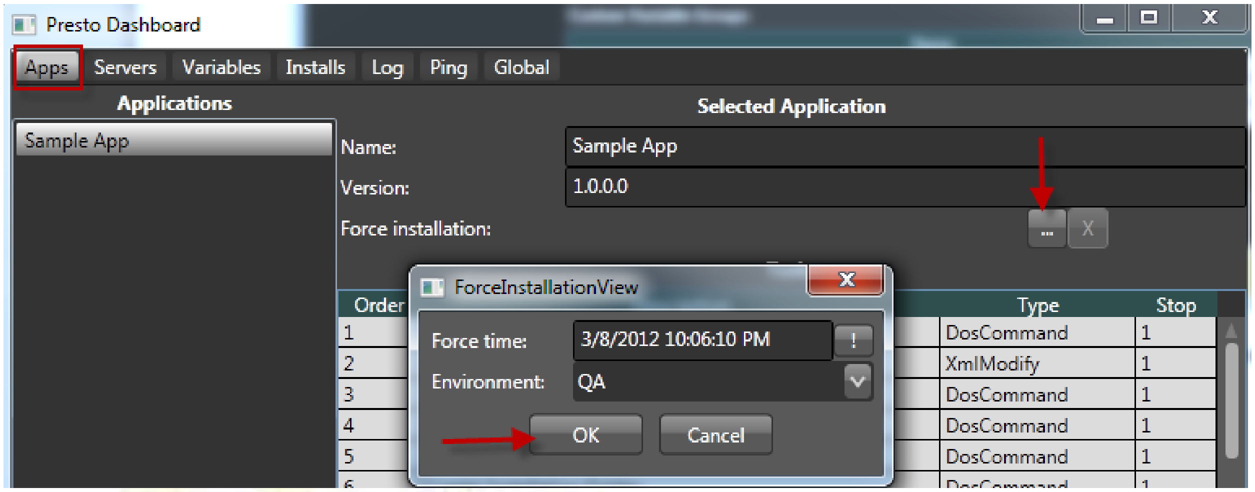
Custom Variable Groups

Name
QA
US
US QA

3/3/2012 9:51:54 PM: DellXps saved.

Now that everything is in place, all we have to do is initiate the installation. There are two ways to do so:

This will cause the application to be installed on all servers in the chosen environment:



Sample App

Name: Sample App

Version: 1.0.0.0

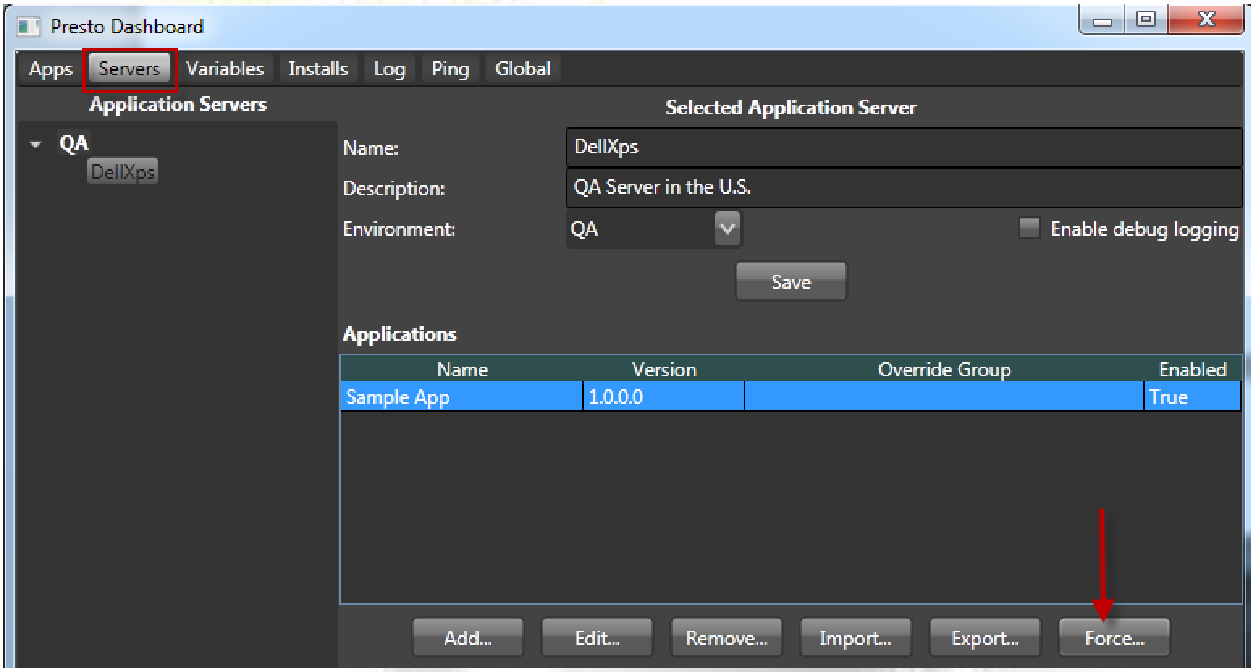
Force installation:

Force time: 3/8/2012 10:06:10 PM

Environment: QA

Order	Type	Stop
1	DosCommand	1
2	XmlModify	1
3	DosCommand	1
4	DosCommand	1
5	DosCommand	1
6	DosCommand	1

And this will cause the installation to happen only on this server:



QA
DellXps

Name: DellXps

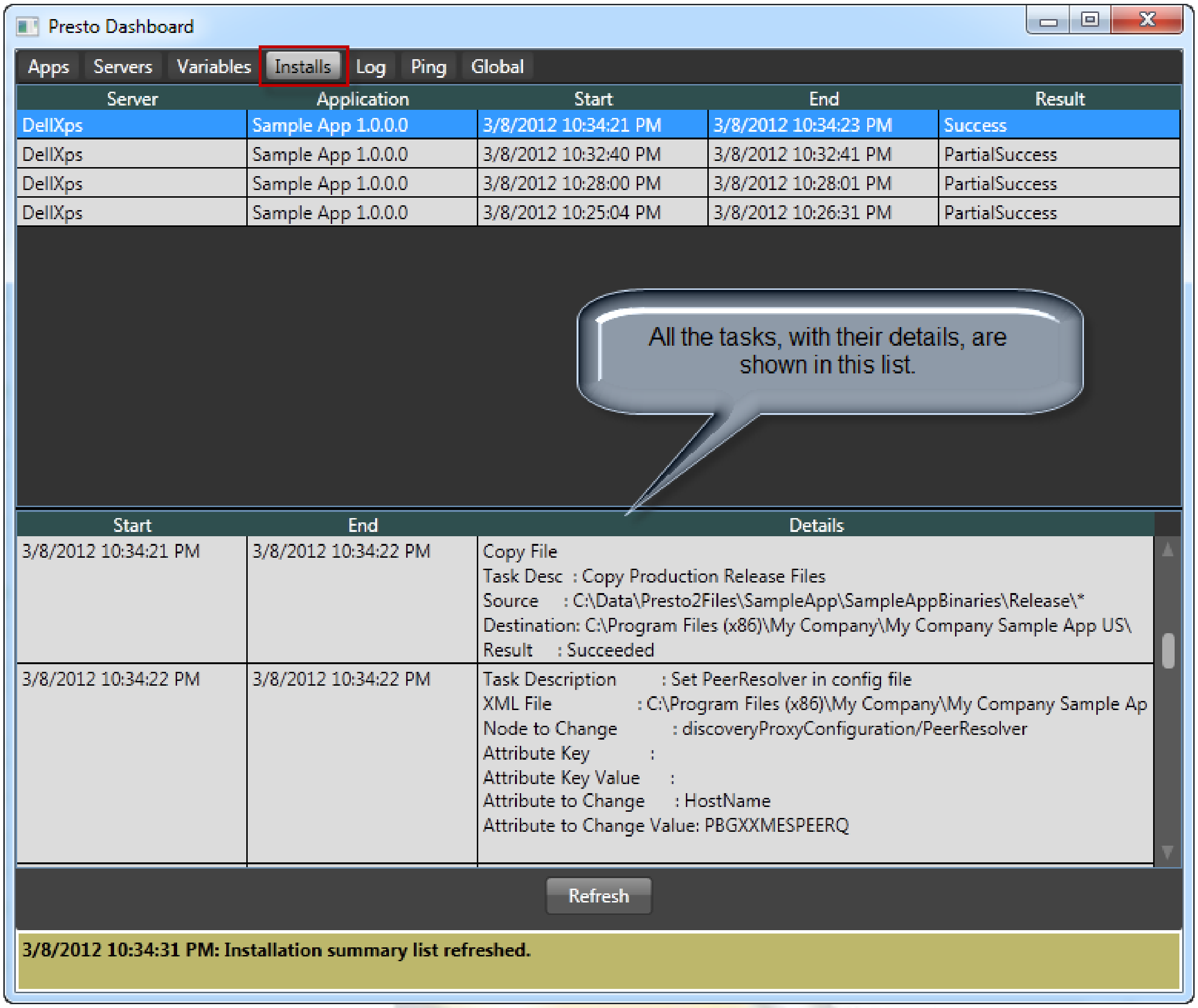
Description: QA Server in the U.S.

Environment: QA

□ Enable debug logging

Name	Version	Override Group	Enabled
Sample App	1.0.0.0		True

When the installation is complete, we can view the results on the Installs tab:



Server	Application	Start	End	Result
DellXps	Sample App 1.0.0.0	3/8/2012 10:34:21 PM	3/8/2012 10:34:23 PM	Success
DellXps	Sample App 1.0.0.0	3/8/2012 10:32:40 PM	3/8/2012 10:32:41 PM	PartialSuccess
DellXps	Sample App 1.0.0.0	3/8/2012 10:28:00 PM	3/8/2012 10:28:01 PM	PartialSuccess
DellXps	Sample App 1.0.0.0	3/8/2012 10:25:04 PM	3/8/2012 10:26:31 PM	PartialSuccess

Start	End	Details
3/8/2012 10:34:21 PM	3/8/2012 10:34:22 PM	Copy File Task Desc : Copy Production Release Files Source : C:\Data\Presto2Files\SampleApp\SampleAppBinaries\Release\* Destination: C:\Program Files (x86)\My Company\My Company Sample App US\ Result : Succeeded
3/8/2012 10:34:22 PM	3/8/2012 10:34:22 PM	Task Description : Set PeerResolver in config file XML File : C:\Program Files (x86)\My Company\My Company Sample Ap Node to Change : discoveryProxyConfiguration/PeerResolver Attribute Key : Attribute Key Value : Attribute to Change : HostName Attribute to Change Value: PBGXXMESPEERQ

3/8/2012 10:34:31 PM: Installation summary list refreshed.

Installation

RavenDB

Presto currently uses build 573 of RavenDB, found here:
http://builds.hibernatingrhinos.com/builds/RavenDB

Download RavenDB-Build-573 and save it on the server’s (what will be your DB server) hard disk.

Unzip…

Self-updating Service Host

The Self-updating service host needs to be installed on every server that will install applications that were set up in the Presto Dashboard. The job of this service is to detect new versions of the Presto Task Runner, shut down the current Presto Task Runner, copy the new files, and finally start a new instance of the Presto Task Runner. This allows for updates to the Presto Task Runner by just copying the new files to one location, and let all of the servers update themselves.

Presto Task Runner

The Presto Task Runner (PTR) files need to be stored in one location, so the Self-updating Service Host can load it from this location.

Presto Dashboard

Dashboard…

Setting up Applications

Adding an Application

Task: Modify XML

Task: Copy File

Task: DOS Command

Importing Tasks

Exporting Tasks

Reordering Tasks

Assigning Custom Variables

System Variables
The following system variables are available to use as variables within tasks:
$(sys:applicationName) – The name of the application being installed
$(sys:applicationVersion) – The version string of the application being installed
$(sys:serverName) – The name of the server where the application is being installed

Typical uses for these variables would be:

Setting up Servers

Adding a Server

Associating Applications with Servers

Importing Applications

Exporting Applications

Assigning Custom Variables

Debug Logging

Custom Variables

Adding a Custom Variable Group

Importing Custom Variable Groups

Exporting Custom Variable Group

Adding Custom Variables

Installs Tab

Log Tab

Global Tab

Freezing Installations

Deployments

Initiating Deployments

Monitoring Deployments