PROGRESS Corticon

Pacific™ Application Server for Corticon®: TCMAN Reference Guide

PROGRESS

Notices

Copyright agreement

Table of Contents

Preface

For details, see the following topics:

- Progress Corticon documentation - Where and What
- Overview of Progress Corticon

Progress Corticon documentation - Where and What

Corticon provides its documentation in various online and installed components.

Access to Corticon tutorials and documentation

Corticon Online Tutorials	
Tutorial: Basic Rule Modeling in Corticon Studio	Online only. Uses samples packaged in the Corticon Studio.
Tutorial: Advanced Rule Modeling in Corticon Studio	Online only.
Corticon Online Documentation	
Progress Corticon User Assistance	Updated online help for the current release.
Corticon Server: Web Console Guide	Included in User Assistance. Not available in Studio help.
Progress Corticon Documentation site	Individual PDFs (including Web Console guide) and JavaDocs

Corticon Documentation on the Progress download site	
Documentation	Package of all guides in PDF format.
What's New Guide	PDF format.
Installation Guide	PDF format.
Corticon Studio Installers	Include Eclipse help for all guides except Web Console.

Components of the Corticon tutorials and documentation set

The components of the Progress Corticon documentation set are the following tutorials and guides:

Corticon Online Tutorials	
Tutorial: Basic Rule Modeling in Corticon Studio	An introduction to the Corticon Business Rules Modeling Studio. Learn how to capture rules from business specifications, model the rules, analyze them for logical errors, and test the execution of your rules -- all without any programming.
Tutorial: Advanced Rule Modeling in Corticon Studio	An introduction to complex and powerful functions in Corticon Business Rules Modeling Studio. Learn the concepts underlying some of Studio's more complex and powerful functions such as ruleflows, scope and defining aliases in rules, understanding collections, using String/DateTime/Collection operators, modeling formulas and equations in rules, and using filters.
Release and Installation Information	
What's New in Corticon	Describes the enhancements and changes to the product since its last point release.
Corticon Installation Guide	Step-by-step procedures for installing all Corticon products in this release.
Corticon Studio Documentation: Defining and Modeling Business Rules	
Corticon Studio: Rule Modeling Guide	Presents the concepts and purposes the Corticon Vocabulary, then shows how to work with it in Rulesheets by using scope, filters, conditions, collections, and calculations. Discusses chaining, looping, dependencies, filters and preconditions in rules. Presents the Enterprise Data Connector from a rules viewpoint, and then shows how database queries work. Provides information on versioning, natural language, reporting, and localizing. Provides troubleshooting of Rulesheets and Ruleflows. Includes *Test Yourself* exercises and answers.

Corticon Studio: Quick Reference Guide	Reference guide to the Corticon Studio user interface and its mechanics, including descriptions of all menu options, buttons, and actions.
Corticon Studio: Rule Language Guide	Reference information for all operators available in the Corticon Studio Vocabulary. Rulesheet and Ruletest examples are provided for many of the operators.
Corticon Studio: Extensions Guide	Detailed technical information about the Corticon extension framework for extended operators and service call-outs. Describes several types of operator extensions, and how to create a custom extension plug-in.
Corticon Enterprise Data Connector (EDC)	
Corticon Tutorial: Using Enterprise Data Connector (EDC)	Introduces Corticon's direct database access with a detailed walkthrough from development in Studio to deployment on Server. Uses Microsoft SQL Server to demonstrate database read-only and read-update functions.
Corticon Server Documentation: Deploying Rules as Decision Services	
Corticon Server: Integration and Deployment Guide	An in-depth, technical description of Corticon Server deployment methods, including preparation and deployment of Decision Services and Service Contracts through the Deployment Console tool. Describes JSON request syntax and REST calls. Discusses relational database concepts and implementation of the Enterprise Data Connector. Goes deep into the server to discuss state, persistence, and invocations by version or effective date. Includes troubleshooting servers though logs, server monitoring techniques, performance diagnostics, and recommendations for performance tuning.
Corticon Server: Deploying Web Services with Java	Details setting up an installed Corticon Server as a Web Services Server, and then deploying and exposing Decision Services as Web Services on the Pacific Application Server (PAS) and other Java-based servers. Includes samples of XML and JSON requests.
Corticon Server: Deploying Web Services with .NET	Details setting up an installed Corticon Server as a Web Services Server, and then deploying and exposing decisions as Web Services with .NET. Includes samples of XML and JSON requests.

Corticon Server: Web Console Guide	Presents the features and functions of remote connection to a Web Console installation to enable manage Java and .NET servers in groups, manage Decision Services as applications, and monitor performance metrics of managed servers.
Pacific™ Application Server for Corticon®: TCMAN Reference Guide	Provides reference information about TCMAN, the command-line utility for managing and administering the Pacific Application Server.

Overview of Progress Corticon

Progress® Corticon® is the Business Rules Management System with the patented "no-coding" rules engine that automates sophisticated decision processes.

Progress Corticon products

Progress Corticon distinguishes its development toolsets from its server deployment environments.

- **Corticon Studio** is the Windows-based development environment for creating and testing business rules:
  - When installed as a standalone application, Corticon Studio provides the complete Eclipse development environment for Corticon as the **Corticon Designer** perspective. You can use this fresh Eclipse installation as the basis for adding other Eclipse toolsets.
  - When installed into an existing Eclipse such as the **Progress Developer Studio** (PDS), our industry-standard Eclipse and Java development environment, the PDS enables development of Corticon applications in the **Corticon Designer** perspective that integrate with other products, such as Progress OpenEdge.

  Note: Corticon Studio installers are available for 64-bit and 32-bit platforms. Typically, you use the 64-bit installer on a 64-bit machine, where that installer is not valid on a 32-bit machine. The 64-bit Studio is recommended because it provides better performance when working on large projects. When adding Corticon to an existing Eclipse, the target Eclipse must be an installation of the same bit width. Refer to the *Corticon Installation Guide* to access, prepare, and install Corticon Studio.

- **Corticon Servers** implement web services for deploying business rules defined in Corticon Studios:
  - **Corticon Server for Java** is supported on various application servers, and client web browsers. After installation on a supported Windows platform, that server installation's deployment artifacts can be redeployed on various UNIX and Linux web service platforms as Corticon Decision Services.
  - **Corticon Server for .NET** facilitates deployment of Corticon Decision Services on Windows .NET Framework and Microsoft Internet Information Services (IIS).

Use with other Progress Software products

Corticon releases coordinate with other Progress Software releases:

- Progress OpenEdge is available as a database connection. You can read from and write to an OpenEdge database from Corticon Decision Services. When Progress Developer Studio for OpenEdge and Progress Corticon Studio are integrated into a single Eclipse instance, you can use the capabilities of integrated business rules in Progress OpenEdge. See the OpenEdge document OpenEdge Business Rules for more information. OpenEdge is a separately licensed Progress Software product.
- Progress DataDirect Cloud (DDC) enables simple, fast connections to cloud data regardless of source. DataDirect Cloud is a separately licensed Progress Software product.
- Progress RollBase enables Corticon rules to be called from Progress Rollbase. Rollbase is a separately licensed Progress Software product.

1

TCMAN Reference

TCMAN is a command-line utility for managing and administering the Pacific Application Server. TCMAN extends the basic Tomcat scripts for starting, stopping, and managing server instances.

This *TCMAN Reference* contains usage information for the `tcman` command as well as syntax information on all of the TCMAN actions.

For details, see the following topics:

The tcman command

Purpose

TCMAN is a command-line utility for managing and administering the Pacific Application Server. On UNIX systems, you run the `tcman.sh` script followed by appropriate TCMAN actions and options. On Windows systems, you run the `tcman.bat` batch file, which is identical syntactically and functionally with `tcman.sh`.

Note: For the sake of brevity, all the syntax statements and examples in this reference show the `tcman.sh` script.

Syntax

```
{ $CATALINA_HOME | $CATALINA_BASE } /bin/tcman.sh action [ general_options ]
[ action_options ]
```

Parameters

`$CATALINA_HOME|$CATALINA_BASE`

Specify whether to run TCMAN from the root directory of the installed Pacific Application Server (`$CATALINA_HOME`) or from the root directory of an instance (`$CATALINA_BASE`). The context of where you run TCMAN (whether from the `/bin` directory of the parent, or the `/bin` directory of an instance) affects which server the utility acts on.

Note: TCMAN automatically determines the value of `CATALINA_BASE` from the directory where you start it. When you run it from the `/bin` directory of an instance, the value of `CATALINA_BASE` is the root directory of the instance. If you run it from the `/bin` directory of the installed Pacific Application Server, the value of `CATALINA_BASE` is the root directory of the installed server (which is the same value as `CATALINA_HOME`).

`action`

Specify which TCMAN action to invoke.

`general_options`

Specify one or more of the TCMAN common options that can apply to most actions. Note that one or more of the general options may be required by a specific action. For example, the `list` action requires `-u` in order to pass a user name and password.

The output of `tcman.sh help` *`action`* includes a list of general options that are applicable to a particular action.

The following table is a list of the common options:

Table 1: TCMAN general options

Common options	Function
`-u` *`user_name`*`:`*`password`*	Pass a valid user name and a password for HTTP Basic access authentication.
`-v`	Display verbose output.
`-M` *`URL`*	Override the default manager that manages Web applications by specifying the URL of an alternative manager. *`URL`* is expressed in the following format: `{http|https}://`*`host`*`:`*`port/manager_application`*

Common options	Function
`-B`	Override default `CATALINA_BASE` environment settings.
`-n`	Debug the TCMAN action but do not execute changes.
`-I` *`instance_name`*	Run TCMAN from the `/bin` directory of the specified instance.

`action_options`

Specify an option that applies to the selected *`action`*. These options are explained in the topics that describe each action.

Example

Run the `help` action from the core server (`/psc/pashome`) to display a list of available TCMAN actions:

```
/psc/pashome/bin/tcman.sh help
usage:  tcman action [options...]
  manager actions:
      list        list deployed applications
      info        list server info
      deploy      deploy application
      undeploy    undeploy application
      reload      reload application
      status      show server status
      leaks       show server memory leaks
      enable      start web application running
      disable     stop running web application
      resources   list server global resources
      sessions    list a web application's sessions
  server actions:
      create      create a new server instance
      delete      delete server instance
      config      dump CATALINA_BASE configuration
      clean       clean/archive log files
      instances   list tracked server instances
      register    manually register an instance
      unregister  manually unregister an instance
      start       start this server
      stop        stop this server
      version     show the server version information
      test        test the server's configuration
  general actions:
      env         show tcman execution environment
      help        show this information
```

Manager actions

This section details the actions available for deploying, running, and monitoring web applications on a server instance.

List deployed applications (list)

Purpose

Display all the web applications that are deployed on an instance.

Note: This command may be used whether the instance is online or offline. However, the output differs. When used offline, TCMAN simply shows a list of deployed application directories in the instance's web applications directory. When used online, it provides additional run-time details about the deployed web applications.

To use this action, the Tomcat manager (`manager.war`) must be deployed on the instance if the instance is online. You can deploy `manager.war` from `$CATALINA_HOME/extras`.

Syntax

```
tcman.sh list [general_options] [-u user_id:password]
```

Parameters

general_options

Specify one or more of the options that can be used with any TCMAN action.

`-u` *user_id*`:`*password*

Specify a valid user name and password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.)

Note: This option is required if the server is online. It is not required if the server is offline.

Example

Show the Web applications deployed to `acme1` when the instance is online:

```
/psc/acme1/bin/tcman.sh list -u tomcat:tomcat
OK - Listed applications for virtual host localhost
/:running:0:ROOT
/manager:running:4:manager
/oemanager:running:0:oemanager
/oeadapters:running:0:oeabl
```

Show the Web applications deployed to `acme1` when the instance is offline:

```
 /psc/acme1/bin/tcman.sh list
OK - Listing directories for /psc/acme1/webapps
/manager:stopped:0:manager
/oeadapters:stopped:0:oeabl
/oemanager:stopped:0:oemanager
/:stopped:0:ROOT
```

Display OS and server information (info)

Purpose

Display server and OS information for a running instance.

To use this action, the Tomcat manager (`manager.war`) must be deployed on the instance and the instance must be running. You can deploy `manager.war` from `$CATALINA_HOME/extras`.

Use the `test` action to show configuration information about a server that is not running.

Syntax

```
tcman.sh info [general_options]  -u user_name:password
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help info` to see which general options are appropriate.

`-u` *user_name*`:`*password*

Pass a valid user name and a password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.)

Example

Display the OS and server information for the running instance named `acme1`:

```
$: /psc/pashome/tcman.sh info -I acme1 -u tomcat:tomcat
OK - Server info
Tomcat Version: Apache Tomcat/7.0.42
OS Name: Linux
OS Version: 2.6.18-164.el5
OS Architecture: amd64
JVM Version: 1.7.0_02-b13
JVM Vendor: Oracle Corporation
```

Deploy a Web application (deploy)

Purpose

Deploy a Web application (`.war` file) to a Pacific Application Server instance whether the server is running (online) or is not running (offline). TCMAN copies the web application to the server's web application directory. If the server is online, you must stop and restart it in order to complete the deployment.

Syntax

```
tcman.sh deploy [general_options] [-u user_id:password] [-a app _name]
war_file_path
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help deploy` to see which general options are appropriate.

`-u` *user_id*`:`*password*

Specify a valid user name and password for HTTP Basic access authentication.

Note: This option is required if the server in online. It is not required if the server is offline.

`-a` *app _name*

Specify a name for the web application. If you do not use this option, the application name will be the same as the `.war` file name.

war_file_path

Specify the location of the web application `.war` file that you want to deploy.

Example

Deploy and rename `oeabl.war` (a web application that implements OpenEdge adapters) to the `acme1` instance of the core `pashome` server:

```
/psc/acme1/bin/tcman.sh deploy -a oeadapters /psc/pashome/extras/oeabl.war
OK - deployed /psc/pashome/extras/oeabl.war to local directory
/psc/acme1/webapps
```

Note: The `$CATALINA_HOME/extras` directory (`/psc/pashome/extras` in the example above) also contains number of instance management applications, including `host-manager.war`, `manager.war`, and `oemanager.war`.

Undeploy a Web application (undeploy)

Purpose

Remove a Web application from running (online) or stopped (offline) instances. If the instance's autodeploy option is off, you must stop and restart a running server to complete removal. Note that the autodeploy option is set in the `.../conf/appserver.properties` file and is off by default.

Syntax

```
tcman.sh undeploy [general_options] [-u user_id:password] app _name
```

Parameters

general_options

Specify one or more of the options that can be used with any TCMAN action. Run `tcman.sh help undeploy` to see which general options are appropriate.

`-u` *user_id*`:`*password*

Specify a valid user name and password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.) This option is required if you are accessing an online instance.

app_name

Specify the name of the web application to remove.

Example

Remove the oemanager application from the `acme1` instance:

```
/psc/acme1/bin/tcman.sh undeploy -u tomcat:tomcat oemanager
              OK - Undeployed application at context path /oemanager
```

Reload a Web application (reload)

Purpose

Restart a deployed, running Web application so that the application can pick up changes to its classes or libraries.

To use this action, the Tomcat manager (`manager.war`) must be deployed on the instance and the instance must be running. You can deploy `manager.war` from `$CATALINA_HOME/extras`.

Note: The reload action does not reload the web application's `web.xml` file. To begin using changes to `web.xml`, you must stop and restart the web application.

Syntax

```
tcman.sh reload [general_options] -u user_id:password app_name
```

Parameters

general_options

Specify one or more of the options that can be used with any TCMAN action. Run `tcman.sh help reload` to see which general options are appropriate.

`-u` *user_id*`:`*password*

Specify a valid user name and password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.)

Note: This option is required if the server in online. It is not required if the server is offline.

app_name

Specify the name of the web application to restart.

Example

Reload the `oemanager` web application running on the `acme1` instance:

```
 /psc/acme1/bin tcman.sh reload  -u tomcat:tomcat oemanager
OK - Reloaded application at context path /oemanager
```

Display detailed server status (status)

Purpose

List information from the core server's memory, including web application statistics. Information includes memory pool usage, connector thread status, and connector status. Output is in XML format. (Note that redirecting the output to an XML viewer makes it more readable.)

To use this action, the Tomcat manager (`manager.war`) must be deployed on the instance and the instance must be running. You can deploy `manager.war` from `$CATALINA_HOME/extras`.

Syntax

```
tcman.sh status [general_options]  -u user_name:password [-f]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help status` to see which general options are appropriate.

`-u` *user_name*`:`*password*

Pass a valid user name and a password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.)

`-f`

Return full status information.

Example

Display core server's memory and web application statistics and use `xmllint` to format for readability:

```
$: tcman.sh status -u tomcat:tomcat | xmllint --format -
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet type="text/xsl" href="/manager/xform.xsl" ?>
<status>
  <jvm>
    <memory free="453196832" total="520028160" max="1051394048"/>
    <memorypool name="PS Eden Space" type="Heap memory" usageInit="50331648"
 usageCommitted="48758784" usageMax="55967744" usageUsed="1525560"/>
    <memorypool name="PS Old Gen" type="Heap memory" usageInit="469762048"
usageCommitted="469762048" usageMax="1006632960" usageUsed="63861584"/>
    <memorypool name="PS Survivor Space" type="Heap memory" usageInit="8388608"
 usageCommitted="1507328" usageMax="1507328" usageUsed="1444184"/>
    <memorypool name="Code Cache" type="Non-heap memory" usageInit="2555904"
 usageCommitted="3407872" usageMax="50331648" usageUsed="3303104"/>
   <memorypool name="PS Perm Gen" type="Non-heap memory" usageInit="67108864"
 usageCommitted="67108864" usageMax="134217728" usageUsed="47406400"/>
  </jvm>
  <connector name="&quot;http-bio-8601&quot;">
    <threadInfo maxThreads="150" currentThreadCount="0"
currentThreadsBusy="0"/>
   <requestInfo maxTime="0" processingTime="0" requestCount="0" errorCount="0"
 bytesReceived="0" bytesSent="0"/>
    <workers/>
  </connector>
  <connector name="&quot;http-bio-8501&quot;">
    <threadInfo maxThreads="300" currentThreadCount="10"
currentThreadsBusy="1"/>
    <requestInfo maxTime="2008" processingTime="2116" requestCount="10"
errorCount="0" bytesReceived="0" bytesSent="5838"/>
    <workers>
      <worker stage="S" requestProcessingTime="2" requestBytesSent="0"
requestBytesReceived="0" remoteAddr="127.0.0.1" virtualHost="localhost"
method="GET" currentUri="/manager/status" currentQueryString="XML=true"
protocol="HTTP/1.1"/>
    </workers>
  </connector>
</status>
```

Display memory leaks (leaks)

Purpose

List Web applications with potential memory leaks.

To use this action, the Tomcat manager (`manager.war`) must be deployed on the instance and the instance must be running. You can deploy `manager.war` from `$CATALINA_HOME/extras`.

Syntax

```
tcman.sh leaks [general_options]  -u user_name:password
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help leaks` to see which general options are appropriate.

`-u` *user_name*`:`*password*

Pass a valid user name and a password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.)

Example

Display memory leaks for web applications deployed on the `acme1` server instance:

```
/psc/acme1/bin/tcman.sh leaks -u tomcat:tomcat
OK - Found potential memory leaks in the following applications:
/warehouse
```

Start a Web application (enable)

Purpose

Start a web application that is deployed but not running.

To use this action, the Tomcat manager (`manager.war`) must be deployed on the instance and the instance must be running. You can deploy `manager.war` from `$CATALINA_HOME/extras`.

Syntax

```
tcman.sh enable [general_options] -u user_id:password app_name
```

Parameters

general_options

Specify one or more of the options that can be used with any TCMAN action. Run `tcman.sh help start` to see which general options are appropriate.

`-u` *user_id*`:`*password*

Specify a valid user name and password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.)

app_name

Specify the name of the web application to start.

Note: To start the ROOT web application, you can specify `/` or `ROOT`.

Example

Start the `oeabl` application deployed on the `acme1` instance:

```
tcman.sh enable -u tomcat:tomcat oeabl
OK - Started application at context path /oeabl
```

Stop a Web application (disable)

Purpose

Stop a running Web application.

To use this action, the Tomcat manager (`manager.war`) must be deployed on the instance and the instance must be running. You can deploy `manager.war` from `$CATALINA_HOME/extras`.

Syntax

```
tcman.sh disable [general_options] [-u user_id:password] app_name
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help disable` to see which general options are appropriate.

`-u` *user_id*`:`*password*

Specify a valid user name and password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.)

app_name

Specify the name of the web application to disable.

Note: To disable the ROOT web application, you can specify / or ROOT.

Example title

Disable the oeabl application running on the acme1 instance:

```
/psc/acme1/bin/tcman.sh disable -u tomcat:tomcat oeabl
OK - Stopped application at context path /oeabl
```

Display global server resources (resources)

Purpose

List the global resources used by the core server.

To use this action, the Tomcat manager (manager.war) must be deployed on the instance and the instance must be running. You can deploy manager.war from $CATALINA_HOME/extras.

Syntax

```
tcman.sh resources [general_options]  -u user_name:password
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run tcman.sh help resources to see which general options are appropriate.

-u *user_name*:*password*

Pass a valid user name and a password for HTTP Basic access authentication.

(The default is -u tomcat:tomcat.)

Example

Display global resources for the running instance, acme1:

```
$: /psc/acme1/bin/tcman.sh resources -u tomcat:tomcat
OK - Listed global resources of all types
ServiceRegistry/ServiceRegistryFactory:com.progress.appserv.services.naming.ServiceRegistry
UserDatabase:org.apache.catalina.users.MemoryUserDatabase
```

Display Web application HTTP sessions (sessions)

Purpose

Display how many sessions are active for the specified Web application, categorized by their duration.

To use this action, the Tomcat manager (`manager.war`) must be deployed on the instance and the instance must be running. You can deploy `manager.war` from `$CATALINA_HOME/extras`.

Syntax

```
tcman.sh sessions [general_options] -u user_id:password app_name
```

Parameters

general_options

Specify one or more of the options that can be used with any TCMAN action.

`-u` *user_id*`:`*password*

Specify a valid user name and password for HTTP Basic access authentication. (The default is `-u tomcat:tomcat`.)

app_name

Specify the name of the web application to analyze for session information.

Example

Show the active sessions for the `manager` application deployed on the `acme1` instance:

```
/psc/acme1/bin/tcman.sh sessions -u tomcat:tomcat manager
OK - Session information for application at context path /manager
Default maximum session inactive interval 30 minutes
<1 minutes: 1 sessions
8 - <9 minutes: 2 sessions
9 - <10 minutes: 1 sessions
```

Server actions

This section details the actions available for creating and monitoring server instances.

Create an instance (create)

Purpose

Create a new instance of the core Pacific Application Server server by running this action from `/bin` directory of the core server (`$CATALINA_HOME/bin/tcman.sh create`).

Syntax

```
tcman.sh create [general_options] [-f] [-p port_num] [-P port_num]
   [-s port_num] [-j port_num] [-W pathname] [-N instance_name]
   [-U user_id ] [-G group_id] base_path
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help create` to see which general options are appropriate.

`-f`

Copy all deployed web application archives (`.war` files) from `$CATALINA_HOME` to the new instance.

`-p` *port_num*

Specify the TCP port that listens for HTTP messages. The default is 8080.

`-P` *port_num*

Specify the TCP port that listens for HTTPS messages. The default is 8443.

`-s` *port_num*

Specify the TCP port to use to stop an instance. On Windows systems, you must specify a shutdown port. On UNIX, shutdown ports are optional.

`-j` *port_num*

Specify the TCP port that listens for AJP13 messages (an Apache protocol for handling requests from a web server to an application server). The default is 8009.

`-W` *pathname*

Specify the directory where web applications will be deployed. The default is `$CATALINA_BASE/webapps`.

`-N` *instance_name*

Specify a name for the instance. The default is the name of the directory where the instance is created.

All instances are automatically registered for tracking when they are created. If you intend to track an instance, the instance name cannot contain spaces or any of the following characters: "[. # |] $? + = { / , }"

-U user_id

Specify the user-id of the owner of all the files and directories of the instance. The default is the user-id of the current process. -G must be specified if you use this option.

-G group_id

Specify the group-id of the owner of all the files and directories of the instance. The default is the group-id of the current process. -U must be specified if you use this option.

base_path

Specify the pathname where you will create the instance.

Example

Create an instance of /psc/pashome in /psc/acme1:

```
$: /psc/pashome/bin/tcman.sh create -p 8501 -P 8601 -s 8701 /psc/acme1
Server instance acme1 created at /psc/acme1
```

Delete an instance (delete)

Purpose

Remove the directory tree and all of the files in an instance. Alias tracking is disabled for servers that are removed.

To execute this action, the instance cannot be running.

Note: You cannot recover any files or directories removed by the delete action. Backup anything you want to save before launching this action.

Also note that you cannot use delete to remove the installed, root server ($CATALINA_HOME).

Syntax

```
tcman.sh delete [general_options] [-y] [base_path | alias_name]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run tcman.sh help delete to see which general options are appropriate.

`-y`

Delete everything without prompting for confirmation.

`base_path`

Specify the pathname of the instance that you intend to delete.

`alias_name`

Refer to the instance that you intend to delete by its alias rather than its pathname.

Example

Delete the instance of `pashome` that was created in `/psc/acme3`:

```
$: /psc/pashome/bin/tcman.sh delete /psc/acme3
The following directory tree will be removed permanently:
 ( WARNING all deployed web applications will be DELETED!! )
/PAS/wrkdir/acme3
/PAS/wrkdir/acme3/conf
/PAS/wrkdir/acme3/temp
/PAS/wrkdir/acme3/common
/PAS/wrkdir/acme3/common/lib
/PAS/wrkdir/acme3/logs
/PAS/wrkdir/acme3/webapps
/PAS/wrkdir/acme3/webapps/ROOT
/PAS/wrkdir/acme3/webapps/ROOT/static
/PAS/wrkdir/acme3/webapps/ROOT/static/error
/PAS/wrkdir/acme3/webapps/ROOT/static/auth
/PAS/wrkdir/acme3/webapps/ROOT/META-INF
/PAS/wrkdir/acme3/webapps/ROOT/WEB-INF
/PAS/wrkdir/acme3/webapps/ROOT/WEB-INF/adapters
/PAS/wrkdir/acme3/webapps/ROOT/WEB-INF/adapters/rest/PingService
/PAS/wrkdir/acme3/webapps/ROOT/WEB-INF/adapters/soap
/PAS/wrkdir/acme3/webapps/ROOT/WEB-INF/classes
/PAS/wrkdir/acme3/webapps/ROOT/WEB-INF/classes/com
/PAS/wrkdir/acme3/webapps/ROOT/WEB-INF/classes/com/progress
/PAS/wrkdir/acme3/webapps/ROOT/WEB-INF/classes/com/progress/appserv
/PAS/wrkdir/acme3/work
/PAS/wrkdir/acme3/bin
Type 'yes' to continue
yes
Delete operation complete
server  removed at /PAS/wrkdir/acme3
```

Display and manage an instance's configuration (config)

Purpose

View, add, update, or delete the property values specified in `../conf/appserver.properties`.

When you run `tcman.sh config` with no parameters, it displays the core Tomcat server's configuration, and all the properties in both `.../conf/appserver.properties` and `.../conf/jvm.properties`. Note, however, that you can only view `jvm.properties`. You cannot modify its contents with the `config` action.

Syntax

```
tcman.sh config [general_options]
[prop_name | prop_name=value | +prop_name=value | ~prop_name]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help config` to see which general options are appropriate.

prop_name

Display the specified property and its value.

prop_name=value

Set the value of a property that exists in `.../conf/appserver.properties`.

+prop_name=value

Add a new property to `.../conf/appserver.properties` and set its value.

~prop_name

Remove the specified property from `.../conf/appserver.properties`.

Examples

Show the configuration and properties of `acme1`, an instance of the core server, `pashome`:

```
$: /psc/acme1/bin/tcman.sh config
Using CATALINA_BASE:   /psc/acme1
Using CATALINA_HOME:   /psc/pashome
Using CATALINA_TMPDIR: /psc/acme1/temp
Using JRE_HOME:        /tools/linuxx86_64/java64/jdk1.7.0_02/
Using CLASSPATH:
/psc/pashome/bin/bootstrap.jar:/psc/pashome/bin/tomcat-juli.jar
Using CATALINA_PID:    /psc/acme1/temp/catalina.pid
Server version: Apache Tomcat/7.0.42
Server built:   Jul 2 2013 08:57:41
Server number:  7.0.42.0
OS Name:        Linux
OS Version:     2.6.18-164.el5
Architecture:   amd64
JVM Version:    1.7.0_02-b13
...
```

Display the value of a single property:

```
$: /psc/acme1/bin/tcman.sh config psc.as.http.port
psc.as.http.port=8501
```

Update the value of a property that exists in the `appserver.properties` file and then check the value:

```
$: /psc/acme1/bin/tcman.sh config psc.as.http.port=6543
$: tcman.sh config psc.as.http.port
psc.as.http.port=6543
```

Add a new property/value pair to the `appserver.properties` file and then check the value:

```
$: /psc/acme1/bin/tcman.sh config +my.home.dir=/home/jarhead
$: tcman.sh config my.home.dir
my.home.dir=/home/jarhead
```

Remove a property/value pair from the `appserver.properties` file and check if deletion was successful:

```
$: /psc/acme1/bin/tcman.sh config ~my.home.dir
$: tcman.sh config my.home.dir
Property does not exist - my.home.dir
```

Caution: There are no restrictions to property removal. You can render the server unable to start if you remove a property required by `conf/server.xml`.

Notes

- All property names are case sensitive.
- You cannot enter multiple property names (*`prop_name`*) on the command line to view, update, or add properties to the `appserver.properties` file.
- You cannot use the config action to update existing values or add new values to the `jvm.properties` file

Display or modify the server features of an instance (feature)

Purpose

View, enable, or disable the server features contained in the `/conf/server.xml` file of an instance.

When you run `tcman.sh feature` with no parameters, it displays a list of the features (and their current status) that you can enable or disable. You can also display the status of a single server feature. After viewing the status of a feature, you can use `tcman.sh feature` to change its setting.

Syntax

```
tcman.sh feature [general_options] [feature_name[={on|off}]]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help feature` to see which general options are appropriate.

feature_name

Specify one of the features defined in an instance's `conf/server.xml` file. Running `tcman.sh feature` without *feature_name* displays a list of all the features.

`on`

Enables the named feature.

`off`

Disables the named feature.

Example

Display the list of server feature settings for `acme1`, enable AJP13 (Apache JServ Protocol. version 1.3), and verify that the feature is enabled:

```
$: /psc/acme1/bin/tcman.sh feature
SecurityListener=off
JMXLifecycle=off
PSCRegistry=on
HTTP=onHTTPS=on
AJP13=off
Cluster=off
UserDatabase=on
JAASRealm=off
LDAPRealm=off
PASInstrument=off
RemoteHostValve=on
RemoteAddrValve=onSingleSignOn=on
AccessLog=on
CrawlerSessionManager=on
StuckSessionValve=on

$: /psc/acme1/bin/tcman.sh feature AJP13=on


$: /psc/acme1/bin/tcman.sh feature AJP13
AJP13=on
```

Notes

- Server features for instances are set in `$CATALINA_BASE/conf/server.xml`. You can change feature status by manually editing this file. However, it is safer to use `tcman.sh feature` to avoid corrupting the file with erroneous entries.

- Run `tcman.sh feature` when the instance is offline.

Clean up or archive a server's log files (clean)

Purpose

Truncate, move, or delete the log files located in the `/logs` directory of the core server or instance. If the server is running, clean truncates log files to zero length. If the server is not running, clean deletes the log files from the file system.

You have the option to save log files to a subdirectory of `/logs`.

Syntax

```
tcman.sh clean [general_options] [-A]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help clean` to see which general options are appropriate.

`-A`

Archive log files to a subdirectory of `$CATALINA_BASE/logs`. The directory is automatically named with a month-day-year-second (*MM-DD-YYYY-ss*) time-stamp format. If the server is not running, the files in `$CATALINA_BASE/logs` are deleted.

Example

Archive the log files of `acme1`, an instance of the core server `pashome`, and save to a file:

```
/psc/pashome/tcman.sh clean -I acme1 -A
```

Display a server's instances (instances)

Purpose

Display all the instances created from the Pacific Application Server installed in `$CATALINA_HOME`.

Note:

The server does not keep track of its instances by default. Before you can track instances, you must create a `$CATALINA_HOME/conf/instances.unix` or a `$CATALINA_HOME/conf/instances.windows` text file. (Use the file name extension corresponding to the OS you are running.) The `instances` action displays the content of the file.

If the instances file exists, the TCMAN utility adds instance entries to the file when you execute a `create` action. TCMAN removes instance entries when you execute the delete action.

You can manually add or remove instance entries from the `instances` file by using the `register` or `unregister` actions.

Syntax

```
tcman.sh instances [general_options]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help instances` to see which general options are appropriate.

Example

Display the instances of the core server installed in `/psc/pashome`:

```
/psc/pashome/bin/tcman.sh instances
acme1 : /psc/acme1 : instance : ok
acme2 : /psc/acme2 : instance : ok
```

Register an instance for tracking (register)

Purpose

Register an instance for tracking purposes.

Note:

Instances are automatically registered for tracking when you execute a `create` action. You use the `register` action to restart tracking on instances after tracking was stopped.

A typical use for unregistering and then re-registering an instance is to make configuration changes when moving instances from one location (core server) to another. The `register` action enables tracking and also updates the value of `CATALINA_HOME` in all of the executable scripts in the instance's `/bin` directory to refer to the new core server.

Syntax

```
tcman.sh register alias_name instance_path
```

Parameters

alias_name

Specify a meaningful name for the instance. The alias name must be unique in the instances file.

instance_path

Specify the OS file system path to where the instance exists. This value will be expanded into a fully qualified OS directory path and will be verified to exist.

Example

Track `test1`, which is an alias for the instance `/psc/acme1`:

```
/psc/pashome/bin/tcman.sh register test1 /psc/acme1
```

Notes

When you register an instance for tracking or create a new instance with the `create` command, an entry is created in the core Pacific Application Server's `$CATALINA_HOME/conf/instances.[unix|windows]` file.

The `instances.[unix|windows]` file is a simple text file, which can be manually edited (with care) in the event that it becomes out of date. The format for entries is:

```
instance_name = base_path
```

An `instances.unix` file uses Unix OS file path syntax (forward slashes), and an `instances.windows` file uses Windows OS file path syntax (backslashes) to specify *base_path*.

Also note that in an instances file:

- Any line starting with a pound-sign (#) is a comment line.
- Blank lines are skipped.
- Instance names cannot contain spaces or any of the following characters: "`[ . # | ] $ ? + = { / , }`"

Stop tracking an instance (unregister)

Purpose

Stop tracking an instance by removing the instance's entry from the `$CATALINA_HOME/conf/instances.[unix|windows]` file.

Note:

You use the `register` action to restart tracking on instances after tracking was stopped with `unregister`.

A typical use for unregistering and then re-registering an instance, is to make configuration changes when moving instances from one location, or core server, to another. The `register` action not only enables tracking, it also updates the value of `CATALINA_HOME` in all of the executable scripts in the instance's `/bin` directory to refer to the new core server.

Syntax

```
tcman.sh unregister alias_name
```

Parameters

alias_name

Specify the alias name of the instance that you want to stop tracking. The alias name must exist in an `instances.[unix|windows]` file.

Example

Stop tracking `test1`, which is an instance of `/psc/pashome`:

```
/psc/pashome/bin/tcman.sh unregister test1
```

Start an instance (start)

Purpose

Start an instance of a Pacific Application Server.

Syntax

```
tcman.sh start [general_options] [-D | -J]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help start` to see which general options are appropriate.

`-D`

Start the server in Tomcat debug mode. `-D` overrides the `-J` option.

`-J`

Start the server in debug mode using the JDPA (Java Platform Debugger Architecture) APIs for debugging. `-J` cannot be used if the `-D` option is specified.

Before you run a server with the `-J` option, you must define a port for the JDPA debugger by setting the `JDPA_ADDRESS` environment variable to a unique TCP network port number.

Example

Start the server in `/psc/acme1`, which is an instance of the core server in `/psc/pashome`:

```
/psc/acme1/bin/tcman.sh start
Using CATALINA_BASE:   /psc/acme1
Using CATALINA_HOME:   /psc/pashome
Using CATALINA_TMPDIR: /psc/acme1/temp
Using JRE_HOME:        /tools/linuxx86_64/java64/jdk1.7.0_02/
Using CLASSPATH:
/psc/pashome/bin/bootstrap.jar:/psc/pashome/bin/tomcat-juli.jar
Using CATALINA_PID:    /psc/acme1/temp/catalina.pid
```

Stop an instance (stop)

Purpose

Stop a running instance.

Syntax

```
tcman.sh stop [general_options] [-F [-w seconds] ]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help stop` to see which general options are appropriate.

`-F`

Kill the sever process if it does not stop 5 seconds. Change the default 5 second interval by using the `-w` option.

`-w` *seconds*

Optionally specify the number of seconds to wait before killing a server process.

Example

Stop the server in `/psc/acme1`, which is an instance of the core server in `/psc/pashome`:

```
/psc/acme1/bin/tcman.sh stop
Using CATALINA_BASE:   /psc/acme1
Using CATALINA_HOME:   /psc/pashome
Using CATALINA_TMPDIR: /psc/acme1/temp
Using JRE_HOME:        /tools/linuxx86_64/java64/jdk1.7.0_02/
Using CLASSPATH:
/psc/pashome/bin/bootstrap.jar:/psc/pashome/bin/tomcat-juli.jar
Using CATALINA_PID:    /psc/acme1/temp/catalina.pid
```

Display server, OS, and runtime version information (version)

Purpose

Show the Apache Tomcat runtime version and OS information for an instance.

To execute this action, the instance cannot be running

Syntax

```
tcman.sh version [general_options]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help version` to see which general options are appropriate.

Example

Display the server and runtime information for `acme1`, an instance of the core server installed in `/psc/pashome`:

```
$: /psc/pashome/bin/tcman.sh version -I acme1
Using CATALINA_BASE:   /psc/acme1
Using CATALINA_HOME:   /psc/pashome
Using CATALINA_TMPDIR: /psc/acme1/temp
Using JRE_HOME:        /tools/linuxx86_64/java64/jdk1.7.0_02/
Using CLASSPATH:
/psc/pashome/bin/bootstrap.jar:/users/doc/agarbacz/psc/pashome/bin/tomcat-juli.jar
Using CATALINA_PID:    /psc/acme1/temp/catalina.pid
Server version: Apache Tomcat/7.0.42
Server built:   Jul 2 2013 08:57:41
Server number:  7.0.42.0
OS Name:        Linux
OS Version:     2.6.18-164.el5
Architecture:   amd64
JVM Version:    1.7.0_02-b13
JVM Vendor:     Oracle Corporation
```

Test a server configuration (test)

Purpose

Displays information on the configuration and environment of an instance. It also displays information about error conditions.

The `test` action starts a server (instance), loads all the configuration files, and then displays information. The instance is stopped, exiting gracefully even if there is an error condition.

To execute this action, the instance cannot be running

Syntax

```
tcman.sh test [general_options]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help test` to see which general options are appropriate.

Example

Run a test of the configuration of `acme1`, which is an instance of the core server installed at `/psc/pashome`:

```
$: /psc/pashome/bin/tcman.sh -I acme1 test
Using CATALINA_BASE:   /psc/acme1
Using CATALINA_HOME:   /psc/pashome
Using CATALINA_TMPDIR: /psc/acme1/temp
Using JRE_HOME:        /tools/linuxx86_64/java64/jdk1.7.0_02/
Using CLASSPATH:
/psc/pashome/bin/bootstrap.jar:/psc/pashome/bin/tomcat-juli.jar
Using CATALINA_PID:    /psc/acme1/temp/catalina.pid
. . .
```

Notes

The `test` action is particularly useful for testing to verify that a server will start and run properly after you make changes to configuration and properties files.

General actions

This section details the actions available for displaying help and server runtime environment information.

Display help (help)

Purpose

Display summary or detailed help for all TCMAN actions, property names, and server features.

Syntax

```
tcman.sh help [action | property | feature]
```

Parameters

action

Show the syntax and options of the specified action. If no action is specified, show a list of all actions and the general options.

property

Show the settings for specified property.

feature

Show if the specified feature is enabled or disabled.

Example

Display the usage help for the `create` action:

```
$: tcman.sh help create
usage: tcman create [options] -p <http-port> [instance-opts] <new-base-path>

    instance-opts:
       [-s <shutdown-port>]
       [-P <https-port>]
       [-j <ajp13-port>]
       [-W <web-apps-dir>]
       [-N <inst-alias-name>]
       [-U <file-owner> -G <file-group>]

   general options:
      -u uid:pwd  pass uid and pwd for HTTP BASIC authentication
      -v          print verbose output
      -M url      override the CATALINA_BASE manager's URL with
                  <{http|https}://<host>:<port>/<mgr-app>
      -B          override CATALINA_BASE environment setting
      -n          debug run action but do not execute changes
```

Display runtime environment information (env)

Purpose

Show details about a server's state.

Syntax

```
tcman.sh env [general_options] [keyword]
```

Parameters

general_options

Specify one or more of the general TCMAN options. Run `tcman.sh help env` to see which general options are appropriate.

keyword

Specify one or more keywords that represent the name of the state that you want to view. If no keyword is specified, then all of the state information is displayed.

Keywords include:

Keyword	Description
`running`	Indicate if a server is running (`1`) or not running (`0`).
`mgrurl`	Display the URL of the manager application.

Keyword	Description
type	Display the server type.
alias	Display the server's alias.
parent	Display the pathname of the parent of an instance.
tracking	Indicate if tracking is on (1) or off (0).
http	Display the server's http port number.
https	Display the server's https port number.
shut	Display the server's shutdown port number. A value of -1 indicates that there is no shutdown port.
pid	Display the server's process id. A hyphen (-) indicates that the server is not running.

Example

Display all of the state information for the instance created in /psc/acme1:

```
/psc/acme1/bin/tscman.sh env
catalina home:     /psc/pashome
catalina base:     /psc/acme1
java home:         /tools/linuxx86_64/java64/jdk1.7.0_02/
jre home:
manager http port: 8501
manager https port:8601
manager shut port: 8701
manager URL:       http://localhost:8501/manager
config type:       instance
config alias:      acme1
config parent:     /psc/pashome
server running:    0
instance tracking: 1
instance file:     /psc/pashome/conf/instances.unix
server process-id: -
```