|  |  |  |  |
| --- | --- | --- | --- |
| **resin**   |  |  |  | | --- | --- | --- | |  |  | [admin/configuration](http://192.168.86.128:8080/resin-doc/admin/index.xtp) |   [admin/configuration](http://192.168.86.128:8080/resin-doc/admin/index.xtp)  [**Quick Start**](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp) - Getting Started with Resin ------------------ [local](#_resin_installation_quick)  [**Eclipse**](http://192.168.86.128:8080/resin-doc/admin/resin-eclipse-support.xtp) - Resin Eclipse Support  [**Configuration**](http://192.168.86.128:8080/resin-doc/admin/config.xtp) - Configuration principles for the Resin Server  [**Overview**](http://192.168.86.128:8080/resin-doc/admin/config-overview.xtp) - Overview of Resin configuration  [**resin.properties**](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp) - Standard Resin properties for configuration  [**resin.xml**](http://192.168.86.128:8080/resin-doc/admin/config-resin-xml.xtp) - Custom resin.xml configuration  [**CanDI**](http://192.168.86.128:8080/resin-doc/admin/config-candi.xtp) - XML configuration for Dependency Injection services  [**EL expressions**](http://192.168.86.128:8080/resin-doc/admin/config-el.xtp) - EL expressions used in configuration  [**Resin Data Directory**](http://192.168.86.128:8080/resin-doc/admin/config-resin-data.xtp) -  [**Migrating**](http://192.168.86.128:8080/resin-doc/admin/migration.xtp) - Migrating to Resin 4.0  [**Administration**](http://192.168.86.128:8080/resin-doc/admin/resin-admin.xtp) - /resin-admin monitoring and administration  [**web console**](http://192.168.86.128:8080/resin-doc/admin/resin-admin-console.xtp) - The web-based Resin administration console  [**Clustering**](http://192.168.86.128:8080/resin-doc/admin/clustering.xtp) - Clustering and load balancing of multiple coordinated Resin servers  [**Overview**](http://192.168.86.128:8080/resin-doc/admin/clustering-overview.xtp) - Overview of Resin's clustering, load-balancing, elastic-computing and distributed caching features  [**Server**](http://192.168.86.128:8080/resin-doc/admin/cluster-server.xtp) - Server (JVM) configuration  [**Cloud**](http://192.168.86.128:8080/resin-doc/admin/cluster-cloud.xtp) - Dynamic Server (Cloud) configuration  [**Command Line**](http://192.168.86.128:8080/resin-doc/admin/resin-admin-command-line.xtp) - The Command Line Resin Administration  [**REST Interface**](http://192.168.86.128:8080/resin-doc/admin/resin-admin-rest.xtp) - Resin REST administration interface  [**Database**](http://192.168.86.128:8080/resin-doc/admin/database.xtp) - Configuration of the Resin database pool  [**Deployment**](http://192.168.86.128:8080/resin-doc/admin/deploy.xtp) - Deploying applications and .war files to a Resin cluster.  [**cloud**](http://192.168.86.128:8080/resin-doc/admin/deploy-cloud.xtp) - deploying .wars to the cluster/cloud  [**command-line**](http://192.168.86.128:8080/resin-doc/admin/deploy-command-line.xtp) - deploying .wars using the Resin command line  [**Filters**](http://192.168.86.128:8080/resin-doc/admin/filters.xtp) - Built-in Filters  [**Health**](http://192.168.86.128:8080/resin-doc/admin/health.xtp)  [**Health Checking**](http://192.168.86.128:8080/resin-doc/admin/health-checking.xtp) - Health checks, conditions, and remediation actions  [**Meters**](http://192.168.86.128:8080/resin-doc/admin/health-meters.xtp) - Statistic graphing with meters  [**Report**](http://192.168.86.128:8080/resin-doc/admin/health-report.xtp) - Health Reporting  [**Watchdog**](http://192.168.86.128:8080/resin-doc/admin/health-watchdog.xtp) - Resin's watchdog process manages Resin servers and checks status for reliability.  [**Installation**](http://192.168.86.128:8080/resin-doc/admin/install.xtp) - Detailed Options for Installation  [**Apache HTTPd**](http://192.168.86.128:8080/resin-doc/admin/starting-resin-apache.xtp) - Using Apache HTTPd with Resin  [**IIS 7 Plugin**](http://192.168.86.128:8080/resin-doc/admin/starting-resin-iis-7.xtp) - Using Microsoft IIS 7 with Resin  [**directory layout**](http://192.168.86.128:8080/resin-doc/admin/resin-directory.xtp) - Resin directory layout  [**Logging**](http://192.168.86.128:8080/resin-doc/admin/logging.xtp) - java.util logging configuration in Resin  [**Scheduled Tasks**](http://192.168.86.128:8080/resin-doc/admin/scheduled-tasks.xtp) - Scheduled tasks  [**Security**](http://192.168.86.128:8080/resin-doc/admin/security.xtp) - Security and authentication configuration  [**Overview**](http://192.168.86.128:8080/resin-doc/admin/security-overview.xtp) - Overview of security concepts  [**authenticators**](http://192.168.86.128:8080/resin-doc/admin/security-authenticators.xtp) - Authenticators for Resin server  [**authentication method**](http://192.168.86.128:8080/resin-doc/admin/security-authentication-method.xtp) - Authenticators for Resin server  [**authorization**](http://192.168.86.128:8080/resin-doc/admin/security-authorization.xtp) - Authenticators for Resin server  [**SSL**](http://192.168.86.128:8080/resin-doc/admin/security-ssl.xtp) - Integrating OpenSSL and JSSE  [**URL Rewrite**](http://192.168.86.128:8080/resin-doc/admin/http-rewrite.xtp) - Resin's URL rewriting capability, replacing mod\_rewrited sites  [**Web Server**](http://192.168.86.128:8080/resin-doc/admin/http.xtp)  [**HTTP server**](http://192.168.86.128:8080/resin-doc/admin/http-server.xtp) - Resin's built-in, high performance HTTP server  [**Virtual Hosts**](http://192.168.86.128:8080/resin-doc/admin/http-virtual-hosts.xtp) - HTTP Virtual Host configuration for multi-host sites  [**Web Applications**](http://192.168.86.128:8080/resin-doc/admin/http-web-apps.xtp) - HTTP servlet container for web applications  [**Proxy Cache**](http://192.168.86.128:8080/resin-doc/admin/http-proxy-cache.xtp) - Resin's built-in proxy cache  [**Classloaders**](http://192.168.86.128:8080/resin-doc/admin/advanced-classloaders.xtp) - Resin classloader architecture  [**Hessian**](http://192.168.86.128:8080/resin-doc/admin/hessian.xtp) - Hessian protocol  [**WebSocket**](http://192.168.86.128:8080/resin-doc/admin/websocket.xtp) - WebSocket support  [**JMX (MXBeans)**](http://192.168.86.128:8080/resin-doc/admin/advanced-jmx.xtp) - JMX management  [**Seam**](http://192.168.86.128:8080/resin-doc/admin/third-party-seam.xtp)  [development](http://192.168.86.128:8080/resin-doc/examples/index.xtp)  [Changes](http://192.168.86.128:8080/resin-doc/changes/index.xtp)  [**Release Notes**](http://192.168.86.128:8080/resin-doc/changes/release-notes.xtp)  [**Resin 4.0.51**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.51.xtp)  [**Resin 4.0.50**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.50.xtp)  [**Resin 4.0.49**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.49.xtp)  [**Resin 4.0.48**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.48.xtp)  [**Resin 4.0.47**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.47.xtp)  [**Resin 4.0.46**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.46.xtp)  [**Resin 4.0.45**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.45.xtp)  [**Resin 4.0.44**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.44.xtp)  [**Resin 4.0.43**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.43.xtp)  [**Resin 4.0.42**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.42.xtp)  [**Resin 4.0.41**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.41.xtp)  [**Resin 4.0.40**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.40.xtp)  [**Resin 4.0.39**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.39.xtp)  [**Resin 4.0.38**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.38.xtp)  [**Resin 4.0.37**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.37.xtp)  [**Resin 4.0.36**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.36.xtp)  [**Resin 4.0.35**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.35.xtp)  [**Resin 4.0.34**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.34.xtp)  [**Resin 4.0.33**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.33.xtp)  [**Resin 4.0.32**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.32.xtp)  [**Resin 4.0.31**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.31.xtp)  [**Resin 4.0.30**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.30.xtp)  [**Resin 4.0.29**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.29.xtp)  [**Resin 4.0.28**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.28.xtp)  [**Resin 4.0.27**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.27.xtp)  [**Resin 4.0.26**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.26.xtp)  [**Resin 4.0.25**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.25.xtp)  [**Resin 4.0.24**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.24.xtp)  [**Resin 4.0.23**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.23.xtp)  [**Resin 4.0.21**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.21.xtp)  [**Resin 4.0.20**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.20.xtp)  [**Resin 4.0.19**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.19.xtp)  [**Resin 4.0.18**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.18.xtp)  [**Resin 4.0.17**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.17.xtp)  [**Resin 4.0.16**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.16.xtp)  [**Resin 4.0.15**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.15.xtp)  [**Resin 4.0.14**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.14.xtp)  [**Resin 4.0.13**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.13.xtp)  [**Resin 4.0.12**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.12.xtp)  [**Resin 4.0.11**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.11.xtp)  [**Resin 4.0.10**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.10.xtp)  [**Resin 4.0.9**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.9.xtp)  [**Resin 4.0.8**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.8.xtp)  [**Resin 4.0.7**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.7.xtp)  [**Resin 4.0.6**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.6.xtp)  [**Resin 4.0.5**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.5.xtp)  [**Resin 4.0.4**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.4.xtp)  [**Resin 4.0.3**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.3.xtp)  [**Resin 4.0.2**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.2.xtp)  [**Resin 4.0.1**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.1.xtp)  [**Resin 4.0.0**](http://192.168.86.128:8080/resin-doc/changes/resin-4.0.0.xtp)  [**Change Logs**](http://192.168.86.128:8080/resin-doc/changes/change-logs.xtp)  [**Resin 4.0 Changes**](http://192.168.86.128:8080/resin-doc/changes/changes.xtp)  [**Resin 3.1 Changes**](http://caucho.com/resin-3.1/changes/changes.xtp)  [Reference](http://192.168.86.128:8080/resin-doc/reference.xtp)  Complete reference documentation |

|  |  |  |  |
| --- | --- | --- | --- |
| **resin 4.0 reference**   |  |  |  | | --- | --- | --- | | [Resin 4.0](http://192.168.86.128:8080/resin-doc/index.xtp) |  | [Quick Start](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp) |   [Quick Start](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp)  If you're using Resin for the first time, this section will show you how to install Resin, start it, and deploy your application.  [**Eclipse**](http://192.168.86.128:8080/resin-doc/admin/resin-eclipse-support.xtp) - Resin Eclipse Support  [Configuration](http://192.168.86.128:8080/resin-doc/admin/config.xtp)  Many sites can modify the *resin.properties* and use the standard *resin.xml* for configuration of their servers, JVMs, clusters, and applications. More complicated sites can modify the *resin.xml* itself. The xml configuration file *resin.xml* supports CDI custom services, EL expressions and variables variables, imports and control structures.  [**Overview**](http://192.168.86.128:8080/resin-doc/admin/config-overview.xtp) - Overview of Resin configuration  [**resin.properties**](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp) - Standard Resin properties for configuration  [**resin.xml**](http://192.168.86.128:8080/resin-doc/admin/config-resin-xml.xtp) - Custom resin.xml configuration  [**CanDI**](http://192.168.86.128:8080/resin-doc/admin/config-candi.xtp) - XML configuration for Dependency Injection services  [**EL expressions**](http://192.168.86.128:8080/resin-doc/admin/config-el.xtp) - EL expressions used in configuration  [**Resin Data Directory**](http://192.168.86.128:8080/resin-doc/admin/config-resin-data.xtp) -  [Migrating](http://192.168.86.128:8080/resin-doc/admin/migration.xtp)  Resin 4.0 introduced a number of new features and capabilities over earlier versions of Resin. Along with these changes, configuration and some operating semantics changed. Certain configuration was deprecated and should be removed or rewritten using newer constructs.  [Administration](http://192.168.86.128:8080/resin-doc/admin/resin-admin.xtp)  The /resin-admin web-app provides an administration overview of a Resin server. Resin-Pro users can obtain information across the entire cluster, profile a running Resin instance, and obtain thread dumps and heap dumps.  All Resin users should familiarize themselves with the thread dump, profile, and heap capabilities.  [**web console**](http://192.168.86.128:8080/resin-doc/admin/resin-admin-console.xtp) - The web-based Resin administration console  [Clustering](http://192.168.86.128:8080/resin-doc/admin/clustering.xtp)  [**Overview**](http://192.168.86.128:8080/resin-doc/admin/clustering-overview.xtp) - Overview of Resin's clustering, load-balancing, elastic-computing and distributed caching features  [**Server**](http://192.168.86.128:8080/resin-doc/admin/cluster-server.xtp) - Server (JVM) configuration  [**Cloud**](http://192.168.86.128:8080/resin-doc/admin/cluster-cloud.xtp) - Dynamic Server (Cloud) configuration  [Command Line](http://192.168.86.128:8080/resin-doc/admin/resin-admin-command-line.xtp)  Starting with version 4.0.17 Resin provides extended set of commands that allow administrators and programmers perform debugging and monitoring tasks on remote Resin server using command line.  All Resin users should familiarize themselves with the thread dump, profile, and heap capabilities.  [REST Interface](http://192.168.86.128:8080/resin-doc/admin/resin-admin-rest.xtp)  Starting with version 4.0.26 Resin provides REST interface. Simple and secure, it can be used for integration with services such as RightScale® and others.  [Database](http://192.168.86.128:8080/resin-doc/admin/database.xtp)  Resin provides a robust and tested connection pool that is used to obtain connections to databases.  [Deployment](http://192.168.86.128:8080/resin-doc/admin/deploy.xtp)  Resin's deployment capabilities include clustered deployment, remote deployment, versioning and rollback, command-line deployment, browser deployment, and standard webapps directory filesystem deployment.  [**cloud**](http://192.168.86.128:8080/resin-doc/admin/deploy-cloud.xtp) - deploying .wars to the cluster/cloud  [**command-line**](http://192.168.86.128:8080/resin-doc/admin/deploy-command-line.xtp) - deploying .wars using the Resin command line  [Filters](http://192.168.86.128:8080/resin-doc/admin/filters.xtp)  [Health](http://192.168.86.128:8080/resin-doc/admin/health.xtp)  Resin includes a powerful and configurable system for monitoring application server health and graphing performance metrics.  [**Health Checking**](http://192.168.86.128:8080/resin-doc/admin/health-checking.xtp) - Health checks, conditions, and remediation actions  [**Meters**](http://192.168.86.128:8080/resin-doc/admin/health-meters.xtp) - Statistic graphing with meters  [**Report**](http://192.168.86.128:8080/resin-doc/admin/health-report.xtp) - Health Reporting  [**Watchdog**](http://192.168.86.128:8080/resin-doc/admin/health-watchdog.xtp) - Resin's watchdog process manages Resin servers and checks status for reliability.  [Installation](http://192.168.86.128:8080/resin-doc/admin/install.xtp)  [**Apache HTTPd**](http://192.168.86.128:8080/resin-doc/admin/starting-resin-apache.xtp) - Using Apache HTTPd with Resin  [**IIS 7 Plugin**](http://192.168.86.128:8080/resin-doc/admin/starting-resin-iis-7.xtp) - Using Microsoft IIS 7 with Resin  [**directory layout**](http://192.168.86.128:8080/resin-doc/admin/resin-directory.xtp) - Resin directory layout  [Logging](http://192.168.86.128:8080/resin-doc/admin/logging.xtp)  Resin can perform access logging, specify where JDK logging interface messages go, and redirect the stderr and stdout for your applications.  [Scheduled Tasks](http://192.168.86.128:8080/resin-doc/admin/scheduled-tasks.xtp)  Resin's ScheduledTask capability lets you schedule events using a flexible cron-style trigger. The task can be any java.lang.Runnable bean, a method specified by EL, or a URL.  [Security](http://192.168.86.128:8080/resin-doc/admin/security.xtp)  [**Overview**](http://192.168.86.128:8080/resin-doc/admin/security-overview.xtp) - Overview of security concepts  [**authenticators**](http://192.168.86.128:8080/resin-doc/admin/security-authenticators.xtp) - Authenticators for Resin server  [**authentication method**](http://192.168.86.128:8080/resin-doc/admin/security-authentication-method.xtp) - Authenticators for Resin server  [**authorization**](http://192.168.86.128:8080/resin-doc/admin/security-authorization.xtp) - Authenticators for Resin server  [**SSL**](http://192.168.86.128:8080/resin-doc/admin/security-ssl.xtp) - Integrating OpenSSL and JSSE  [URL Rewrite](http://192.168.86.128:8080/resin-doc/admin/http-rewrite.xtp)  Resin comes with powerful and extensible rewrite capabilities that is as powerful as Apache's mod\_rewrite.  [Web Server](http://192.168.86.128:8080/resin-doc/admin/http.xtp)  Description and configuration of Resin's high-performance HTTP web server.  [**HTTP server**](http://192.168.86.128:8080/resin-doc/admin/http-server.xtp) - Resin's built-in, high performance HTTP server  [**Virtual Hosts**](http://192.168.86.128:8080/resin-doc/admin/http-virtual-hosts.xtp) - HTTP Virtual Host configuration for multi-host sites  [**Web Applications**](http://192.168.86.128:8080/resin-doc/admin/http-web-apps.xtp) - HTTP servlet container for web applications  [**Proxy Cache**](http://192.168.86.128:8080/resin-doc/admin/http-proxy-cache.xtp) - Resin's built-in proxy cache  [Classloaders](http://192.168.86.128:8080/resin-doc/admin/advanced-classloaders.xtp)  Resin organizes resources and classloaders into nested environments. Each environment context merges its own configuration with configuration inherited from the parent environment. Each environment context: server, host, web-app, etc. may be configured with any of the environment configuration tags.  [Hessian](http://192.168.86.128:8080/resin-doc/admin/hessian.xtp)  [WebSocket](http://192.168.86.128:8080/resin-doc/admin/websocket.xtp)  [JMX (MXBeans)](http://192.168.86.128:8080/resin-doc/admin/advanced-jmx.xtp)  JMX Consoles provide access to both the MBean's that Resin publishes for information about and control of the Resin server and Application specific MBeans.  [Seam](http://192.168.86.128:8080/resin-doc/admin/third-party-seam.xtp) |

|  |  |  |  |
| --- | --- | --- | --- |
| resin installation quick start  |  |  |  | | --- | --- | --- | | [admin/configuration](http://192.168.86.128:8080/resin-doc/admin/index.xtp) |  | [Eclipse](http://192.168.86.128:8080/resin-doc/admin/resin-eclipse-support.xtp) |   If you're using Resin for the first time, this section will show you how to install Resin, start it, and deploy your application.   1. [Preconditions](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#Preconditions) 2. [Directory Overview](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#DirectoryOverview) 3. [Installing Resin](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#quickstart)    1. [Debian Unix .deb and apt-get](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#DebianUnix.debandapt-get)    2. [RedHat and CentOS .rpm files](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#RedHatandCentOS.rpmfiles)    3. [Other Unix, Linux, Solaris, and Mac OS X](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#OtherUnixLinuxSolarisandMacOSX)    4. [Windows](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#Windows) 4. [Installing a license file for Resin Pro](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#InstallingalicensefileforResinPro) 5. [Resin as a Web Server](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#ResinasaWebServer) 6. [Browser-Based Administration (/resin-admin)](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#Browser-BasedAdministrationresin-admin)    1. [Installation with resin.properties and generate-password](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#Installationwithresin.propertiesandgenerate-password)    2. [Installation with /resin-admin](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#Installationwithresin-admin) 7. [Deploying Applications](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#DeployingApplications)    1. [Local network command-line deployment](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#Localnetworkcommand-linedeployment)    2. [Remote network command-line deployment](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#Remotenetworkcommand-linedeployment)    3. [webapps directory deployment](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#webappsdirectorydeployment) 8. [Running Resin as a Daemon](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#RunningResinasaDaemon) 9. [Command-Line PDF Reports](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#Command-LinePDFReports) 10. [Resin IDE Support](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#ResinIDESupport) 11. [More Installation Options](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#MoreInstallationOptions) 12. [Next Steps](http://192.168.86.128:8080/resin-doc/admin/starting-resin.xtp#NextSteps)  Preconditions  * Resin requires JDK 6.0 or later(JDK或以上版本). * An HTML 5 browser（支持H5的浏览器） for some /resin-admin features.   Oracle's JDK for Windows, Solaris, and Linux can be found at <http://www.oracle.com/technetwork/java/javase/downloads/index.html>. Oracle also has links to some other ports of the JDK. Resin will not be fully functional while using a JRE. Directory Overview When Resin is installed on a **Unix** system:   * **resinctl** is the [command-line interface](http://192.168.86.128:8080/resin-doc/admin/starting-resin-command-line.xtp). * [Configuration properties](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp) are in /etc/resin/resin.properties * [Deployed applications](http://192.168.86.128:8080/resin-doc/admin/deploy.xtp)（应用部署路径及网站根目录） are put in /var/resin/webapps * Logs are in /var/log/resin * Licenses belong in /etc/resin/licenses * OpenSSL keys belong in /etc/resin/keys   **Unix Directories**  /etc/resin/resin.properties # configuration properties  /etc/resin/resin.xml # configuration file 配置文件  /etc/resin/licenses/ # license location 证书路径  /etc/resin/keys/ # openssl keys  /var/resin/ # resin.root (content)  /var/resin/webapps/ # default deployment directory 默认部署路径  /var/log/resin/ # resin logs 日志路径  /usr/local/share/resin/ # resin.home (binaries and libraries)  When Resin is installed on a **Windows** system:   * **resin** is the [command-line interface](http://192.168.86.128:8080/resin-doc/admin/starting-resin-command-line.xtp). * [Configuration properties](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp) are in conf/resin.properties * [Deployed applications](http://192.168.86.128:8080/resin-doc/admin/deploy.xtp) are in webapps * Logs are in log * Licenses belong in conf/licenses * OpenSSL keys belong in conf/keys   **Windows/unzip Directories**  resin-4.0.x/ # installation directory: resin.home, resin.root  conf/resin.properties # configuration properties  conf/resin.xml # configuration file  conf/licenses/ # license location  conf/keys/ # openssl keys  webapps/ # default deployment directory  logs/ # resin logs Installing Resin Installation steps for the major operating systems are outlined below: Debian Unix .deb and apt-get Debian users can download a .deb packaged version of Resin or use apt-get to install Resin. The Debian package performs all of the installation steps above for you and creates all the recommended server and content directories. Simply download from the [Resin download page](http://caucho.com/download) and install using dpkg.  Alternatively, you can add Caucho's Debian repository to your system's repositories to use automated update tools like Synaptic and apt-get. To do this, you can add the debian repository as follows:  unix# add-apt-repository http://caucho.com/download/debian  After adding this line, update your local repository cache by running:  unix# apt-get update  Install Resin Professional with this command:  unix# apt-get install resin-pro  Or install Resin Open Source with this command:  unix# apt-get install resin RedHat and CentOS .rpm files RPM files are available at <http://caucho.com/download>. The RPM public key is at <http://caucho.com/download/rpm/RPM-GPG-KEY-caucho>.  **RPM/yum install**  unix# rpm --import http://caucho.com/download/rpm/RPM-GPG-KEY-caucho  unix# yum install http://caucho.com/download/rpm/4.0.30/x86\_64/resin-pro-4.0.30-1.x86\_64.rpm Other Unix, Linux, Solaris, and Mac OS X  1. Install JDK 6 or later and link /usr/java to the Java home or define the environment variable *JAVA\_HOME*. 2. tar -vzxf resin-4.0.x.tar.gz 3. cd resin-4.0.x 4. ./configure    * some details on the [./configure options](http://192.168.86.128:8080/resin-doc/admin/starting-resin-command-line.xtp). 5. make 6. sudo make install 7. Execute sudo resinctl start    * or run java -jar lib/resin.jar start 由于resin也是由java写的服务，所以也可以执行jar包 8. Browse to [http://localhost:8080](http://localhost:8080/)  Windows  1. Install JDK 6 or later. 2. Check that the environment variable *JAVA\_HOME* is set to the JDK location, e.g. c:\java\jdk1.6.0\_14 3. Unzip resin-4.0.x.zip 4. Define the environment variable *RESIN\_HOME* to the location of Resin, for example c:\resin-4.0.x 5. Execute resin.exe    * or run java -jar lib/resin.jar start 6. Browse to [http://localhost:8080](http://localhost:8080/)  Installing a license file for Resin Pro After installing Resin Professional, you need to install the license file.  If you have a license file for Resin, save it in */etc/resin/licenses*. You can also install the license from the command line with:  unix> resinctl license-add mylicense.license Resin as a Web Server Resin includes a high-performance HTTP server that outperforms NginX and Apache httpd（比nginx和apache的性能还高）. The easiest and fastest Resin configuration uses Resin as the web server as well as the application server（应用Resin作为web服务器和应用服务器）. We highly recommend you start with this configuration although you are free（免费） to use other web servers like Apache or IIS with Resin.  Resin Web Server has been tested to serve 100,000 requests per second（每秒1万pv）, making it comparable in performance to top C-based web servers. See [Wiki: Resin Web Server: Resin 4.0.x vs nginx 1.2 performance](http://wiki4.caucho.com/NginX_1.2.0_versus_Resin_4.0.29_performance_tests) for a comparison of Resin's performance vs nginx, a fast C-based HTTP server.  Keep in mind, Resin can be used for development or evaluation（激活） in addition to using it in production. If desired, you can easily apply for a development license to enable Resin Pro features（可以申请一个开发版本的证书来体验Resin专业版的特性）. You can also simply use Resin Open Source to start development.  browser <-> Resin httpd/servlets,beans <- html,jsp  The built-in HTTP server listens on port 8080 in the default configuration and can be changed to listen on the HTTP port 80 instead during deployment.  **Example: Starting Resin**  unix> resinctl start  c:\windows> resin start  For troubleshooting your installation, you can also run Resin in "console" mode, which will let you see the logging messages in your console.  **Example: Starting Resin in console mode**  unix> resinctl console 可以作为调试  For more details, see the [Resin Web Server](http://192.168.86.128:8080/resin-doc/admin/http-server.xtp) configuration page. Browser-Based Administration (/resin-admin) To enable the /resin-admin browser-based administration, you'll need to create an admin user and password（可以通过resin.properties配置文件添加或者通过浏览器添加）. You can either create the user in resin-admin itself, or if you are using the standard resin.xml and resin.properties you can generate the key from the command line resinctl Installation with resin.properties and generate-password You can generate the user and password properties from the command-line. Resin's password must be hashed for security.  **Example: generate-password**  unix> resinctl generate-password my-user my-password  admin\_user : my-user 此为上面命令指定的用户名  admin\_password : {SSHA}HTfP60Ceq0K0IAvN31wQtBxtql9D+8Wo 此为密码的加密后的值  Add the admin\_user and admin\_password lines to the end of the /etc/resin/resin.properties（需要将生成的admin\_user和密码添加到resin.properties文件并重启服务器才能通过这个用户登录管理页面） file. You may also use those values to change your admin-users.xml file.  You can update the resin.properties in one step on unix by using a pipe:  **Example: generate-password for resin.properties**  unix# resinctl generate-password my-user my-password >> /etc/resin/resin.properties #使用管道操作 Installation with /resin-admin  1. Create an admin user following the prompt at /resin-admin. 2. Rename admin-users.xml.generated to admin-users.xml. 3. Change the resin\_admin\_external to true in the resin.xml if you need access from a non-local IP address（如果允许）外网能够访问管理页面的话则可以将resin\_admin\_external设置为true). 4. Browse /resin-admin with an HTML 5 browser.   On Linux, *admin-users.xml.generated* is typically generated in /etc/resin/.  The steps are for security reasons. Copying the admin-users.xml verifies that you have access to the server. And the default resin\_admin\_external=false makes sure you're not exposing the /resin-admin to the internet.  For more information, see the [/resin-admin documentation](http://192.168.86.128:8080/resin-doc/admin/resin-admin-console.xtp). Deploying Applications Once you've made sure Resin is working, you can start to run applications and add content. Local network command-line deployment Command-line deployment(命令行部署) on a local network deploys a .war file（部署一个.war文件） to a running Resin server, using the .war file's name as the context-path（上下文路径）. It looks like the following example.  **Example: command-line deployment**  unix> resinctl deploy hello.war  unix> resinctl undeploy hello.war  The URL for the application would be: *http://localhost:8080/hello*.  To deploy to the root context-path（部署到根目录）, use a file name ROOT.war or use the --name attribute.  注：可能需要指定用户名或者密码（开启了admin\_remote\_enable），或者开启cluster\_system\_key  [root@master resin-4.0.53]# ./bin/resin.sh deploy /home/resiner/app/www-demo/hello.war  Authentication is required to access the remote service.  Include --user and --password parameters to connect, or enable 'cluster\_system\_key' in resin.properties.  BamError[type=auth,group=not-authorized,text=No user and password credentials were presented and cluster-system-key is not configured]  [root@master resin-4.0.53]# ./bin/resin.sh --user Daniel --password xxxxx deploy /home/resiner/app/www-demo/hello.war  Deployed production/webapp/default/hello from /home/resiner/app/www-demo/hello.war to http://127.0.0.1:8080/hmtp  **Example: root command-line deployment**  unix> resinctl deploy --name ROOT mywar.war  unix> resinctl undeploy --name ROOT    部署hello到根目录，这样就会直接访问192.168.86.128:8080的话就会直接输出hello的默认页面。实际上是将hello.war中的内容直接替换了ROOT的内容。  **由此也可以写个脚本来执行文件的复制操作**  **由此也知道resin支持热部署，只需要见war包复制到web路径即可**  [resiner@master resin-4.0.53]$ ./bin/resin.sh deploy --name ROOT /home/resiner/app/www-demo/hello.war  Authentication is required to access the remote service.  Include --user and --password parameters to connect, or enable 'cluster\_system\_key' in resin.properties.  BamError[type=auth,group=not-authorized,text=No user and password credentials were presented and cluster-system-key is not configured]  [resiner@master resin-4.0.53]$ ./bin/resin.sh deploy --user Daniel --password xxxxx --name ROOT /home/resiner/app/www-demo/hello.war  Deployed production/webapp/default/ROOT from /home/resiner/app/www-demo/hello.war to http://127.0.0.1:8080/hmtp  [resiner@master resin-4.0.53]$ ls webapps/ROOT  index.jsp META-INF WEB-INF  下线某个应用，实际上将该应用下的入口全部删除  [resiner@master resin-4.0.53]$ ./bin/resin.sh undeploy --user my-user --password my-password hello  Undeployed hello from <http://127.0.0.1:8080/hmtp>  [resiner@master resin-4.0.53]$ ls ./webapps/hello/  WEB-INF Remote network command-line deployment Deploying to a remote network requires some more configuration for security reasons:   1. enable remote administration (disabled by default) 2. configure an admin user and password   After the changes, resin.properties will look something like:  **Example: resin.properties for remote deployment**  Reisin.properties中可以指定多个用户  ...  admin\_user : my-user  admin\_password : {SSHA}HTfP60Ceq0K0IAvN31wQtBxtql0D+8Wo  admin\_remote\_enable : true  When you deploy, you will need to give the user and password:  **Example: remote deploy**  unix> resinctl deploy --user my-user --password my-password hello.war webapps directory deployment You can deploy .war files by copying them to the webapps directory like this（部署命令也是将war文件拷贝到网站根目录）: resin-4.0.x/webapps/hello.war. The URL for the application would be:<http://localhost:8080/hello>.  You can also deploy .wars in exploded（war模式—-将WEB工程以包的形式上传到服务器 ；war exploded模式—将WEB工程以当前文件夹的位置关系上传到服务器） form like: resin-4.0.x/webapps/hello/index.php. （也可以直接在应用的web路径中直接创建文件目录和文件）The URL for the application would be: <http://localhost:8080/hello>.  You can use a web.xml file to configure the 'hello' web application（类似tomcat）: resin-4.0.x/webapps/hello/WEB-INF/web.xml.  For more information on deployment, see the [deployment documentation](http://192.168.86.128:8080/resin-doc/admin/deploy.xtp). Running Resin as a Daemon Most production environments will run Resin as a background daemon. When running as a daemon, Resin detaches from the console and continues running until it is stopped.  The following are the basic steps to running Resin as a daemon(也可以使用参数console在前台启动:resin.sh console):   1. Start resin with resinctl start 2. Stop resin with resinctl stop 3. Restart resin with resinctl restart   The .rpm and .deb files install Resin in /etc/init.d/resin, which will start Resin when the system boots. Command-Line PDF Reports仅专业版 PDF reports about the Resin server can be generated with the command-line, the /resin-admin browser-based GUI, or as automatic health system tasks.  From the command-line you can generate a pdf-report with the following:  **Example: generating a PDF snapshot report**  unix> resinctl pdf-report  generated /var/resin/log/default-Watchdog-20111010T1426.pdf  You can also generate a report for the most recent restart event saved by the watchdog. The watchdog report will give information about why Resin was last stopped.  **Example: generating a PDF watchdog report**  unix> resinctl pdf-report -watchdog  generated /var/resin/log/default-Watchdog-20111010T1426.pdf Resin IDE Support Resin includes excellent support for Eclipse. In fact, using the Eclipse support may be the easiest way get started with Resin.  Resin plugin support is included in Indigo (Eclipse 3.7) and above (you can install the Resin plugin manually for earlier Eclipse versions). The Resin plugin allows you to create new server instances, configure servers, start servers, stop servers, restart instances, deploy/undeploy applications, debug on the server and so on. You can even automatically download and install the latest version of Resin using the plugin or apply for a development license for Resin Pro.  Further details on the Resin Eclipse plugin is available [here](http://192.168.86.128:8080/resin-doc/admin/resin-eclipse-support.xtp). More Installation Options More installation and configuration are available at [installation options](http://192.168.86.128:8080/resin-doc/admin/install.xtp). Next Steps  * [**resin.properties**](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp) configuration shows how to customize basic configuration. [Local](#_resin.properties:_standard_configur) * [**resin.xml**](http://192.168.86.128:8080/resin-doc/admin/config-resin-xml.xtp) configuration shows more advanced and specialized configuration. * [**command-line**](http://192.168.86.128:8080/resin-doc/admin/resin-admin-command-line.xtp) describes using the **resinctl** command line interface. * [**deployment**](http://192.168.86.128:8080/resin-doc/admin/deploy.xtp) deploying applications to a running server. * [**clustering**](http://192.168.86.128:8080/resin-doc/admin/clustering.xtp) describes clustering, cloud, and dynamic servers. * [**resin-admin**](http://192.168.86.128:8080/resin-doc/admin/resin-admin.xtp) describes the /resin-admin browser-based interface. * [**health**](http://192.168.86.128:8080/resin-doc/admin/health.xtp) describes the Resin health and monitoring system. * [**REST admin**](http://192.168.86.128:8080/resin-doc/admin/resin-admin-rest.xtp) describes the REST interface for remote third-party admin integration. * [**Resin HTTP Web Server**](http://192.168.86.128:8080/resin-doc/admin/http.xtp) describes fast, scalable HTTP web server. * [**HTTP URL Rewrite**](http://192.168.86.128:8080/resin-doc/admin/http-rewrite.xtp) describes Resin's URL rewriting (like mod\_rewrite). * [**Security**](http://192.168.86.128:8080/resin-doc/admin/security.xtp) describes authentication, authorization and SSL. |

# resin.properties: standard configuration

|  |  |  |
| --- | --- | --- |
| [Overview](http://192.168.86.128:8080/resin-doc/admin/config-overview.xtp) |  | [resin.xml](http://192.168.86.128:8080/resin-doc/admin/config-resin-xml.xtp) |

Most common Resin properties are configured by resin.properties. The resin.properties variables are used by the main resin.xml(resin.xml使用了resin.properties中定义的变量) configuration file.

1. [Overview](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#Overview)
   1. [Qualifying properties by cluster or server](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#Qualifyingpropertiesbyclusterorserver)
   2. [Extending resin.properties in custom resin.xml](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#Extendingresin.propertiesincustomresin.xml)
2. [Application Server Key Properties](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#ApplicationServerKeyProperties)
3. [Web Tier (Load Balancer) Key Properties](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#WebTierLoadBalancerKeyProperties)
4. [Memcached Tier Key Properties](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#MemcachedTierKeyProperties)
5. [property reference](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#propertyreference)
   1. [accept\_thread\_max](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#acceptthreadmax)
   2. [accept\_thread\_min](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#acceptthreadmin)
   3. [admin\_password](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#adminpassword)
   4. [admin\_user](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#adminuser)
   5. [app.http](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#app.http)
   6. [app.https](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#app.https)
   7. [cluster\_system\_key](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#clustersystemkey)
   8. [dependency\_check\_interval](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#dependencycheckinterval)
   9. [elastic\_cloud\_enable](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#elasticcloudenable)
   10. [elastic\_dns](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#elasticdns)
   11. [elastic\_server](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#elasticserver)
   12. [email](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#email)
   13. [http](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#http)
   14. [http\_address](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#httpaddress)
   15. [http\_ping\_urls](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#httppingurls)
   16. [jvm\_args](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#jvmargs)
   17. [openssl\_file](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#opensslfile)
   18. [openssl\_key](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#opensslkey)
   19. [openssl\_password](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#opensslpassword)
   20. [port\_thread\_max](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#portthreadmax)
   21. [properties\_import\_url](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#propertiesimporturl)
   22. [proxy\_cache\_enable](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#proxycacheenable)
   23. [proxy\_cache\_size](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#proxycachesize)
   24. [remote\_cli\_enable](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#remoteclienable)
   25. [rest\_admin\_enable](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#restadminenable)
   26. [rest\_admin\_ssl](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#restadminssl)
   27. [sendfile](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#sendfile)
   28. [session\_store](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#sessionstore)
   29. [setuid\_user](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#setuiduser)
   30. [setuid\_group](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#setuidgroup)
   31. [tcp\_cork](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#tcpcork)
   32. [web\_admin\_enable](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#webadminenable)
   33. [web\_admin\_external](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#webadminexternal)
   34. [web\_admin\_ssl](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#webadminssl)
   35. [webapp\_multiversion\_routing](http://192.168.86.128:8080/resin-doc/admin/config-resin-properties.xtp#webappmultiversionrouting)

# Overview

The resin.properties file is a condensed configuration（简要的配置） for the most common Resin configurations. Since it is a set of variables（变量的集合） used by the main resin.xml confguration file, the variables are defined by convention（约束，按照resin的规则定义的变量）. The sections following show some of the most common configurations, followed by a description of the variables.

The resin.properties is in the same directory as resin.xml. In unix systems as /etc/resin.properties. In systems deployed in a single directory, it will be ${resin.home}/conf/resin.properties.

**Example: key resin.properties**

...

app\_servers : 127.0.0.1:6800

app.http : 8080

setuid\_user : resin

setuid\_group : resin

jvm\_args : -Xmx2G -XX:MaxPermSize=256m

...

## Qualifying properties by cluster or server

集群和服务器支持的属性

Properties can be restricted(限制) to a server or a cluster by prefixing the variable by the cluster or server name（在变量之前加上前缀，前缀为集群或者服务的名字）. For example, a http port can be specific to the server named 'app-0' by using app-0.http. The port can be specific to the cluster 'app' with app.http.

**Example: http restrictions**

http : 8080 # default value if unspecified

app-0.http : 8081 # server 'app-0' uses 8081

app-1.http : 8082 # server 'app-1' uses 8082

app.http : 8089 # servers in 'app' cluster use 8089

web.http : 8090 # servers in 'web' cluster use 8090

## Extending resin.properties in custom resin.xml

Because the resin.properties variables are defined by the resin.xml, sites that use their own custom resin.xml can define their own variables in their resin.xml and set those variables in the resin.properties.（在resin.xml中可以使用resin.properties中定义的变量）

For custom users, the resin.properties is read using a <resin:properties> tag in the resin.xml（该标签引入resin.properties文件）. The variables are used with the standard EL values（使用EL表达式来应用变量）.

**Example: resin.xml import and use of resin.properties**

<resin xmlns="http://caucho.com/ns/resin"

xmlns:resin="urn:java:com.caucho.resin">

<resin:properties path="${\_\_DIR\_\_}/resin.properties" optional="true"/>

<dependency-check-interval>${my\_var?:'2s'}</dependency-check-interval>

...

</resin>

The previous example defines the dependency-check-interval using a resin.properties variable my\_var and a default value of 2s.

# Application Server Key Properties

A basic application server configuration needs a few key properties to configure the HTTP port, the servers in the cluster, the operating system user and the JVM arguments for memory and GC（一个基本的应用服务器配置需要一些关键的属性来配置HTTP端口、集群的服务、操作系统的用户以及用来内存管理和GC的jvm参数）.

Each server in the cluster can have the same resin.xml and resin.properties. On start, Resin will detect（检测） which server is the local one, select it and start it.

**Example: key resin.properties**

...

app\_servers : 127.0.0.1:6800 192.168.86.01:6800

app.http : 8080 #应用服务HTTP端口

setuid\_user : resin #应用实例的用户

setuid\_group : resin #应用实例的用户组

jvm\_args : -Xmx2G -XX:MaxPermSize=256m #jvm参数设置

...

The server names configured by app\_servers are generated automatically by position: app-0, app-1, ..., app-n.

|  |  |
| --- | --- |
| PROPERTY | DESCRIPTION |
| app\_servers | A list of IP:port addresses for each server in the app-tier cluster(应用集群). |
| app.http | The HTTP port for each app-tier server（集群中每个应用的HTTP端口）. |
| setuid\_user | The operating system user name for the Resin instance（运行resin实例的用户）. |
| setuid\_group | The operating system group name for the Resin instance（运行resin实例的用户组）. |
| jvm\_args | The Java arguments for the Resin instance（运行resin实例的jvm参数）. |

app.http configures the HTTP port for the 'app' cluster. The http defines the variable used in the resin.xml, and the app restricts（限制） it to servers in defined by app\_servers. This system allows for different http ports when starting multiple servers on the same machine, like a web-tier load balancer, an app-tier server, and an memcached server.

# Web Tier (Load Balancer) Key Properties

负载均衡关键属性

This web tier configuration has one web server that handles the HTTP, load balancing, and proxy caching, and one application server that runs the servlet applications（web tier配置web服务器，包括HTTP，负载均衡和代理缓存以及在servlet应用中运行的应用服务器）. The web server load balances and proxies HTTP requests to the backend application cluster（网络服务器的负载均衡和HTTP代理需要应用集群app的支持）. To scale up, add more servers to app\_servers.（为了扩展规模，增加更多的服务器到应用服务器集群）

**Example: web-tier resin.properties**

...

web\_servers : 127.0.0.1:6810 #web服务器集群

web.http : 80 #集群中每个服务器的

HTTP端口

proxy\_cache\_enable : true #启用代理缓存

proxy\_cache\_size : 256m #代理缓存的大小

app\_servers : 127.0.0.1:6800

app.http : 8080

setuid\_user : resin

setuid\_group : resin

jvm\_args : -Xmx2G -XX:MaxPermSize=256m

...

The server names configured by web\_servers are generated automatically by position: web-0, web-1, ..., web-n（根据配置的顺序自动给每个web服务器赋值名称）.

**Example: CLI starting the servers**

# resinctl start-all

The start-all command-line will start both servers（app-servers 和 web-servers） in the example because both servers are listening to a local port.

# Memcached Tier Key Properties

To configure Resin as a memcached server, add IP:port addresses to the memcached\_serversconfiguration.

**Example: memcached-tier resin.properties**

...

memcached\_servers : 127.0.0.1:6820

memcached\_port : 11211

memcached.http : 8080

The memcached.http is used for the /resin-admin managment and REST administration.

# property reference

# accept\_thread\_max

accept\_thread\_max limits the maximum threads waiting for a new connection on the port. Higher values are more resiliant to connection spikes because the idle threads are ready and waiting.

In general, the default value is reasonable and does not usually need changing.

**Example: accept\_thread\_max**

...

accept\_thread\_max : 32

accept\_thread\_min : 4

...

# accept\_thread\_min

accept\_thread\_min triggers a new thread to listen for connections when the number of waiting threads drops below the limit. It works with accept\_thread\_max to manage the threads waiting in the accept state. Higher values can improve connection spike resposiveness.

In general, the default value is reasonable and does not usually need changing.

**Example: accept\_thread\_min**

...

accept\_thread\_max : 32

accept\_thread\_min : 4

...

# admin\_password

admin\_password is the password used with admin\_user for /resin-admin administration, REST, and remote resinctl CLI administration.

admin\_password is generated with either the CLI resinctl generate-password or the /resin-admin login screen.

**Example: admin\_user in resin.properties**

...

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

web\_admin\_enable : true

web\_admin\_ssl : true

web\_admin\_external : false

remote\_cli\_enable : false

rest\_admin\_enable : true

rest\_admin\_ssl : true

...

**Example: resinctl generate-password**

unix> resinctl generate-password my-admin my-password

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

# admin\_user

admin\_user creates a login user name for /resin-admin administration, REST, and remote resinctlCLI administration. It is used with admin\_password and web\_admin\_enable.

admin\_password is generated with either the CLI resinctl generate-password or the /resin-admin login screen.

**Example: admin\_user in resin.properties**

...

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

web\_admin\_enable : true

web\_admin\_ssl : true

web\_admin\_external : false

remote\_cli\_enable : false

rest\_admin\_enable : true

rest\_admin\_ssl : true

...

**Example: resinctl generate-password**

unix> resinctl generate-password my-admin my-password

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

# app.http

The HTTP port for the 'app' cluster. The "app." is a prefix to the http property, restricting it to the named cluster, "app".

**Example: app.http**

...

app.http : 8080

...

# app.https

The HTTPS port for the 'app' cluster. The "app." is a prefix to the https property, restricting it to the named cluster, "app".

app.https is generally used with openssl\_file, openssl\_key and openssl\_password.

**Example: app.https**

...

app.https : 8443

...

# cluster\_system\_key

cluster\_system\_key configures a shared secret across Resin servers in a cluster. The shared secret is checked when servers in the cluster connect. The key can be any string.

**Example: cluster\_system\_key**

cluster\_system\_key : ms8cntp8743z

# dependency\_check\_interval

dependency\_check\_interval sets how often Resin will check for updated files. In development this will be a small value. In deployment it can be a larger value.

**Example: dependency\_check\_interval**

dependency\_check\_interval : 2m

# elastic\_cloud\_enable

elastic\_cloud\_enable lets dynamic servers join a cluster. If the "start" command either has an "--elastic-server" flag or if the elastic\_server property is set, Resin will connect to the cluster as a new dynamic server.

The cluster is either specified by "--cluster foo" or the home\_cluster property. The cluster triad hub servers are specified by the app\_servers property.

**Example: elastic\_cloud\_enable**

app\_servers : 182.168.1.10:6800

elastic\_server : true

elastic\_cloud\_enable : true

home\_cluster : app

To start a dynamic server, use resinctl with start and specify --elastic-server to force a dynamic server. The --cluster is optional if a home-cluster has been defined.

**Example: elastic\_cloud\_enable**

# resinctl start --elastic-server --cluster app

# elastic\_dns

elastic\_dns is used in cloud environments that assign IP addresses or DNS names after the server starts, for example in an EC2 environment. When elastic\_dns is enabled, Resin will start the server, wait and retry until the local IP addresses or DNS resolve to a matching server.

For example, a Resin cluster might have a fixed pre-allocated DNS name of "app0.mydomain" and app\_servers might include that address. At server boot time, the cloud assigns the dynamic IP 192.168.2.114. Later the cloud or the user will attach the IP to the DNS name "app0.mydomain". Resin will then recognize it and start with the correct value.

1. Reserve DNS address "app0.mydomain" with cloud provider.
2. Configure resin.properties with that fixed address and elastic\_dns.
3. Start Resin instance in the cloud (with address 192.168.2.114)
4. [Resin is in wait mode until the following step because it doesn't lookup "app0.mydomain" to 192.168.2.114 in the configuration files.]
5. Assign the DNS name "app0.mydomain" to 192.168.1.10 with the cloud provider.
6. Resin will detect the new assignment and start as that server.

**Example: elastic\_dns**

app\_servers : app0.mydomain:6800

elastic\_dns : true

You can specify the elastic\_dns in the resin.properties or equivalently specify it on the command line as --elastic-dns.

**Example: CLI elastic-dns**

# resinctl start-all --elastic-dns

# elastic\_server

elastic\_server starts the server as a dynamic server and joins a cluster. The dynamic server must have a hub server configured in the cluster to register itself. The cluster must also have theelastic\_cloud\_enable set to enable dynamic servers.

When the "start" command either has an "--elastic-server" flag or if the elastic\_server property is set, Resin will connect to the cluster as a new dynamic server.

The cluster is either specified by "--cluster foo" or the home\_cluster property. The cluster triad hub servers are specified by the app\_servers property.

* Either elastic\_server in the resin.properties or --elastic-server on the command line to start an elastic server.
* Either home\_server in the resin.properties or --cluster on the command line to specify the cluster.
* elastic\_cloud\_enable in the resin.properties to enable dynamic servers.
* At least one static server defined in app\_servers in the resin.properties as the hub. Three servers is preferred.

**Example: elastic\_server**

app\_servers : 182.168.1.10:6800

elastic\_server : true

elastic\_cloud\_enable : true

home\_cluster : app

To start a dynamic server, use resinctl with start and specify --elastic-server or elastic\_serverin the resin.properties to force a dynamic server. The --cluster is optional if a home-clusterhas been defined.

**Example: elastic\_server**

# resinctl start --elastic-server --cluster app

# email

email is an admin email address used to send automatic reports like weekly PDF reports.

**Example: email**

email : myuser@admin.example.org

# http

The HTTP port for all servers, usually restricted to the cluster or server as app.http or web.http. If the plain http property is used, it is the default http for all servers.

**Example: http**

...

http : 8080

app.http : 8081 # overrides for a server in cluster 'app'

...

# http\_address

http\_address selects a specific IP address and port for HTTP. It can be useful when each server needs to bind to a different address.

The http\_address can be qualified by the cluster or the individual server like other properties. For example, app-0.http\_address configures the address for server app-0 in the app cluster.

**Example: http\_address**

...

app-0.http\_address : 192.168.1.10

app-1.http\_address : 192.168.1.11

app.http : 8080

...

# http\_ping\_urls

http\_ping\_urls is a list of URLs to check for server reliability. Resin's health system will periodically check the URLs on the current server. If the server does not respond, the health system will force an automatic restart of Resin.

**Example: http\_ping\_urls**

http\_ping\_urls : http://www.myfoo.com/my-test http://www.mybar.com/my-test2

# jvm\_args

jvm\_args defines the JDK memory and GC parameters to start the Resin process.

**Example: jvm\_args**

jvm\_args : -Xmx2G -XX:MaxPermSize=256m

# openssl\_file

openssl\_file configures the OpenSSL \*.crt certificate file when using SSL. The file location is relative to the resin.xml file.

**Example: openssl\_file**

app.https : 8443

openssl\_file : keys/foo.crt

openssl\_key : keys/foo.key

openssl\_password : my-password

# openssl\_key

openssl\_key configures the OpenSSL \*.key key file when using SSL. The file location is relative to the resin.xml file.

**Example: openssl\_key**

app.https : 8443

openssl\_file : keys/foo.crt

openssl\_key : keys/foo.key

openssl\_password : my-password

# openssl\_password

openssl\_password configures the password for OpenSSL key file when using SSL.

**Example: openssl\_key**

app.https : 8443

openssl\_file : keys/foo.crt

openssl\_key : keys/foo.key

openssl\_password : my-password

# port\_thread\_max

port\_thread\_max restricts the active threads available for a single port. Connections beyond the limit will use the async/select manager, and queue for an available thread.

port\_thread\_max can be used to limit the maximum load on an overloaded system. If more connection arrive than port\_thread\_max and beyond the select manager, they will be delayed instead of overloading the system.

**Example: port\_thread\_max**

...

port\_thread\_max : 256

accept\_thread\_max : 32

accept\_thread\_min : 4

# properties\_import\_url

properties\_import\_url adds a new properties file to be processed after the resin.properties. Cloud servers using EC2-style /user-data can use the url to customize properties for each server.

**Example: properties\_import\_url for EC2**

...

properties\_import\_url : http://169.254.169.254/latest/user-data

# proxy\_cache\_enable

proxy\_cache\_enable enables Resin's HTTP proxy cache. The proxy cache improves performance by saving the server's generated pages. It can be used for formatted web pages or REST-style GET pages that change infrequently.

**Example: proxy\_cache\_enable**

proxy\_cache\_enable : true

proxy\_cache\_size : 256m

# proxy\_cache\_size

proxy\_cache\_size is the size of the in-memory proxy cache size. The actual maximum size is much larger, because the memory is used as a block cache for a disk based store. The memory block cache is also used for distributed caching.

**Example: proxy\_cache\_size**

proxy\_cache\_enable : true

proxy\_cache\_size : 256m

# remote\_cli\_enable

remote\_cli\_enable enables the CLI resinctl to be used for remote servers. By default the CLI is restricted to the local network.

**Example: remote\_cli\_enable**

remote\_cli\_enable : true

# rest\_admin\_enable

rest\_admin\_enable enables REST-based administration of a Resin server. REST can be used to integrate Resin with remote administration consoles or status scripts.

rest\_admin\_enable requires an admin\_user and admin\_password for security. It can optionally be restricted to SSL ports with resin\_admin\_ssl.

**Example: rest\_admin\_enable in resin.properties**

...

rest\_admin\_enable : true

rest\_admin\_ssl : true

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

...

# rest\_admin\_ssl

rest\_admin\_ssl requires an SSL connection for REST-based administration of a Resin server. REST can be used to integrate Resin with remote administration consoles or status scripts.

**Example: rest\_admin\_ssl in resin.properties**

...

rest\_admin\_enable : true

rest\_admin\_ssl : true

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

...

# sendfile

sendfile enables operating system sendfile() call, which will send an entire file without requiring Resin to read the file itself. sendfile is particularly useful when much of the traffic is large static files.

**Example: sendfile**

...

sendfile : true

tcp\_cork : true

# session\_store

session\_store enables clustered persistent sessions for failover.

**Example: session\_store**

...

session\_store : true

...

# setuid\_user

On unix, setuid\_user runs the Resin instance as the specified user for security.

**Example: setuid\_user**

...

setuid\_user : resin

setuid\_group : resin

...

# setuid\_group

On unix, setuid\_group runs the Resin instance as the specified group for security.

**Example: setuid\_group**

...

setuid\_user : resin

setuid\_group : resin

...

# tcp\_cork

tcp\_cork enables advanced TCP flow control on Linux systems for improved performance of large files. When it is enabled, sent data will be buffered in the operating system until the buffers fill, instead of being sent out with partial buffers. An application flush() will still force the data to be sent out.

**Example: tcp\_cork**

...

tcp\_cork : true

# web\_admin\_enable

web\_admin\_enable enables /resin-admin for a Resin server. /resin-admin can show the server status, report statistics graphs, deploy applications, and generate PDF reports.

web\_admin\_enable requires an admin\_user and admin\_password for security. It can optionally be restricted to SSL ports with web\_admin\_ssl.

**Example: web\_admin\_enable in resin.properties**

...

web\_admin\_enable : true

web\_admin\_ssl : true

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

...

# web\_admin\_external

web\_admin\_external enables /resin-admin access for servers outside the local network. If enabled, it should be always used with web\_admin\_ssl.

**Example: web\_admin\_external in resin.properties**

...

web\_admin\_enable : true

web\_admin\_external : true

web\_admin\_ssl : true

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

...

# web\_admin\_ssl

web\_admin\_ssl requires an SSL connection for /resin-admin administration of a Resin server. /resin-admin can show the server status, report statistics graphs, deploy applications, and generate PDF reports.

**Example: web\_admin\_ssl in resin.properties**

...

web\_admin\_enable : true

web\_admin\_ssl : true

admin\_user : my-admin

admin\_password : {SSHA}G3UOLv0dkJHTZxAwmhrIC2CRBZ4VJgTB

...

# webapp\_multiversion\_routing

webapp\_multiversion\_routing is a deployment versioning system where Resin selects the most recent deployed application. Web-apps named with numeric suffixes, e.g. foo-10.0.war and can be browsed as /foo. When a new version of the web-app is deployed, Resin continues to route active session requests to the previous web-app version while new sessions go to the new version, so users will not be aware of the application upgrade.

**Example: webapp\_multiversion\_routing**

...

webapp\_multiversion\_routing : true

...