



ServletExec[™]

ServletExec[™] 6.0

User Guide

for Microsoft Internet Information Server
Sun ONE Web Server
Sun Java System Web Server
and Apache HTTP Server

NEW ATLANTA COMMUNICATIONS, LLC

ServletExec™ 6.0 User Guide

November 26, 2007

Version 6.0



Copyright © 1997-2008 New Atlanta Communications, LLC

100 Prospect Place • Alpharetta, Georgia 30005-5445

Phone 678.256.3011 • Fax 678.256.3012

<http://www.newatlanta.com>

ServletExec is a trademark of New Atlanta Communications, LLC. Registration pending.
All other trademarks and registered trademarks herein are the property of their respective owners.

This product includes software developed by the Apache Software Foundation (<http://www.apache.org>).

No part of this document may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise without the prior written consent of New Atlanta Communications, LLC.

New Atlanta Communications, LLC makes no representations or warranties with respect to the contents of this document and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, New Atlanta Communications, LLC reserves the right to revise this document and to make changes from time to time in its content without being obligated to notify any person of such revisions or changes.

The Software described in this document is furnished under a Software License Agreement (“SLA”). The Software may be used or copied only in accordance with the terms of the SLA. It is against the law to copy the Software on tape, disk, or any other medium for any purpose other than that described in the SLA.

Contents

1. INTRODUCTION	1
1.1 SERVLETEXEC PRODUCTS.....	1
1.2 STARTING/RESTARTING SERVLETEXEC	2
1.2.1 SE ISAPI.....	2
1.2.1.1 Reinitializing ServletExec/ISAPI	2
1.2.2 AS Product Only	3
1.3 VERIFYING YOUR SERVLETEXEC INSTALLATION	4
1.4 ACCESSING THE SERVLETEXEC ADMIN UI	5
1.5 LICENSING SERVLETEXEC	5
1.6 SERVLETEXEC ADMIN UI SECURITY	6
1.6.1 Admin UI Authentication	6
1.6.1.1 Change the Authentication Method	7
Setting the Admin to Require SSL Connections	7
1.6.2 Setting the Security Roles and Users.....	8
1.6.3 Allowing Specific IPs to Access the Admin UI.....	8
1.7 VIEWING LOGS.....	9
1.7.1 Logging Options.....	10
1.8 TECHNICAL SUPPORT	12
2. JAVA VIRTUAL MACHINE.....	13
2.1 VIEWING SYSTEM PROPERTIES	13
2.2 VIEWING JAVA VM SETTINGS.....	14
2.3 SELECTING AND CONFIGURING A JAVA VM	15
2.4 DEFINING THE GLOBAL CLASSPATH.....	15
2.5 SETTING JAVA VM OPTIONS (SE/ISAPI ONLY)	17
2.6 USING OPTIONAL PACKAGES	17
3. WEB APPLICATIONS	19
3.1 WEB APPLICATION OVERVIEW	19
3.1.1 Directories.....	19
3.1.1.1 WEB-INF directory.....	20
3.1.2 Packaging	20
3.2 THE "DEFAULT-APP" WEB APPLICATION	20
3.3 USING SERVLETEXEC'S WEB APPLICATION ADMIN UI	21
3.4 RUNNING THE EXAMPLE WEB APPLICATION	21
3.5 DEPLOYING WEB APPLICATIONS	23
3.5.1 Creating a Web Application	24
3.5.2 Auto-Deploying a Web Application	25
3.5.3 Placing a Web Application onto the file system.....	26
3.5.4 Manually deploying a Web Application.....	26
3.5.4.1 ServletExec Extensions	28
Static File Caching	28
3.5.4.2 Static Page Serving	29
3.5.5 ServletExec Web Application Extensions.....	29
3.5.5.1 Session Tracking.....	30
3.5.5.2 External Libraries.....	30
3.5.6 Setting Up Roles, Users, and Role Mapping.....	31
3.5.6.1 Roles	31
3.5.6.2 Users	33
3.5.6.3 Role Mapping	35
3.5.7 Editing the Web Application's web.xml Settings.....	36
3.5.7.1 Application	36
3.5.7.2 Web Application Servlets.....	46
3.5.7.3 Filters.....	50
3.5.7.4 Security	52

3.5.8 Reloading a Web Application.....	56
3.5.9 Removing/Undeploying a Web Application	57
4. DATA SOURCES	58
4.1 MANAGING DATA SOURCES	58
4.2 ADDING A DATA SOURCE	59
4.2.1 Accessing a Data Source From a Servlet.....	60
4.3 EDITING A DATA SOURCE	61
4.4 REMOVING A DATA SOURCE	61
5. JAVASERVER PAGES (JSP)	62
5.1 JAVASERVER PAGES OVERVIEW	62
5.2 RUNNING THE JSP EXAMPLES	63
5.3 CONFIGURING JAVASERVER PAGES	63
5.3.1 Assigning a Servlet Alias	66
5.4 USING JAVASERVER PAGES	66
5.4.1 Invoking a JSP Page from a Servlet.....	67
5.4.2 JSP Compiler.....	67
5.5 MICROSOFT IIS EXTENSION MAPPING	67
6. JSP STANDARD TAG LIBRARY (JSTL).....	69
6.1 JSTL OVERVIEW.....	69
6.2 RUNNING THE JSTL EXAMPLES	69
6.3 CONFIGURING THE JSTL	70
6.4 USING THE JSTL.....	70
6.5 JDBC DRIVERS USED BY THE JSTL.....	70
7. JAVASERVER FACES (JSF).....	71
7.1 JSF OVERVIEW.....	71
7.2 RUNNING THE JSF EXAMPLES	71
8. SERVLETEXEC/AS BUILT-IN WEB SERVER	72
8.1 WEB SERVER SETTINGS.....	72
8.2 ACCESS LOGS WITH THE BUILT IN WEB SERVER	73
9. WEB SERVICES	76
10. MULTI-HOSTING / VIRTUAL SERVER ENVIRONMENTS	77
10.1 HARDWARE AND SOFTWARE VIRTUAL SERVERS	77
10.2 MICROSOFT IIS.....	78
10.3 SUN ONE / SJSWS.....	78
10.4 SERVLETExec/AS AND VIRTUAL SERVERS.....	79
10.5 THE VIRTUAL SERVER NAMED "DEFAULT"	79
10.6 ADDING/CONFIGURING A VIRTUAL SERVER.....	80
10.7 ADMINISTERING VIRTUAL SERVERS	80
10.8 COMPATIBILITY WITH OLDER BROWSERS.....	81
11. RESOURCE MONITORING.....	82
11.1 MONITORING REQUESTS	82
11.2 MONITORING SESSIONS.....	83
11.3 MONITORING THREADS.....	85
11.4 MONITORING MEMORY	86
12. DEVELOPING SERVLETS	88
12.1 SERVLETExec TECH SUPPORT FAQ	88
12.2 JAVA SERVLET API DOCUMENTATION	88
12.3 COMPILING SERVLETS	89
12.4 DEBUGGING SERVLETS	89
12.5 PROFILING SERVLETS.....	90
12.6 SERVLET THREADS	90

12.7 USER ACCOUNTS FOR MICROSOFT IIS	91
12.8 UNIX FILE DESCRIPTORS.....	91
12.9 SSL REQUEST ATTRIBUTES.....	92
12.9.1 Microsoft IIS.....	92
12.9.2 SunONE / SJSWS	92
12.9.3 Covalent Fast Start Server (CFSS)	92
12.9.4 Apache-SSL.....	93
12.9.5 Apache with mod_ssl.....	93
13. CUSTOMIZABLE SESSION TRACKING	94
13.1 INSTALLING CUSTOM SESSION MANAGERS	94
13.1.1 Installing a Generic Session Manager.....	94
13.1.2 Deploying the Example Session Manager.....	95
13.2 CREATING YOUR OWN SESSION MANAGER	95
13.3 USING THE JDBC SESSION MANAGER	96
13.3.1 How it Works.....	96
13.3.2 Tweaking the JDBC Session Manager's Properties.....	96



1. Introduction

New Atlanta ServletExec 6.0 is a high-performance, reliable, and cost-effective web application server that implements the Java™ Servlet API 2.5, JavaServer Pages™ (JSP) 2.1, and JSP Standard Tag Library (JSTL) 1.2 standards defined by Sun Microsystems, Inc. as component technologies of the Java 2 Platform, Enterprise Edition (J2EE™). Additional information about J2EE, servlets, and JSP technologies can be found on Sun's web site:

<http://java.sun.com/j2ee/>

ServletExec enables you to deploy servlets and JSPs on Microsoft® Internet Information Server (IIS), SunONE Web Server, Sun Java System Web Server (SJSWS), and Apache HTTP Server in a standard, robust, high-performance environment.

1.1 ServletExec Products

ServletExec supports two basic configurations (in-process and out-of-process) with two products. The key characteristics of each configuration are listed in Table 1 (below).

- **ServletExec 6.0 ISAPI**, an in-process web server plugin that adds high-performance servlet and JSP support to Microsoft IIS on Windows® 2008/Vista/2003/XP/2000.
- **ServletExec 6.0 Application Server [AS]**, an out-of-process web application server that includes web server adapters for deployment with Microsoft IIS, Apache, SunONE, and SJSWS. It also comes with a built-in, java-based web server which can be used (for development) in lieu of running behind a commercial grade web server. SE AS is available for Windows 2008/Vista/2003/XP/2000 and various UNIX platforms, including SPARC Solaris, Linux, HP-UX, and AIX.

We recommend using SE AS unless you're using Microsoft IIS **and** have specific reasons to use SE ISAPI.

ServletExec ISAPI	ServletExec AS
<ul style="list-style-type: none"> ▪ Runs within the web server process (loaded as a Windows DLL) 	<ul style="list-style-type: none"> ▪ Runs as a standalone process, separate from the web server
<ul style="list-style-type: none"> ▪ Best performance 	<ul style="list-style-type: none"> ▪ Greatest architectural flexibility
<ul style="list-style-type: none"> ▪ Easiest to install and configure 	<ul style="list-style-type: none"> ▪ Enhanced safety (process isolation)
<ul style="list-style-type: none"> ▪ Single Java VM per web server 	<ul style="list-style-type: none"> ▪ Multiple Java VMs per web server
<ul style="list-style-type: none"> ▪ ServletExec starts/stops automatically when the web server starts/stops 	<ul style="list-style-type: none"> ▪ ServletExec starts/stops independently of the web server

Table 1. ServletExec Configurations

1.2 Starting/Restarting ServletExec

1.2.1 SE ISAPI

If you've installed ServletExec/ISAPI, ServletExec automatically initializes and creates a Java VM instance when the IIS web server receives its first request.

1.2.1.1 Reinitializing ServletExec/ISAPI

Reinitializing ServletExec/ISAPI requires you to completely stop and restart the Microsoft IIS web server process.

Note

Stopping a website (or all websites) from the Microsoft Management Console (Internet Service Manager) does not stop the IIS process, but only stops that web site from responding to client requests.

To reinitialize IIS 7.0 on Windows 2008/Vista

You must "tell" IIS to unload the ServletExec DLL (which will cause the JVM in which SE is running to also be unloaded). There are three ways to do this in Windows 2008/Vista.

Using Server Manager...

1. Under Configuration, click **Services**.
2. Click **World Wide Web Publishing Service**, and then click **Stop** or **Restart**.

Using the command line...

- From the ServletExec ISAPI directory, execute the batch file STOP_IIS.BAT, to completely stop IIS

Using the Internet Information Services(IIS) Manager administrative tool...

- **Stop** or **Recycle** the single IIS Application Pool in which SE ISAPI is running.

To reinitialize IIS 6.0/5.1/5.0 on 2003/XP/2000

You must stop or recycle the **IIS Admin Service**, which unloads the ServletExec DLL. There are three ways to do this in Windows 2003/XP/2000.

Using Control Panel...

1. In Control Panel, click **Services**.
2. Click **IIS Admin Service**, and then click **Stop** or **Restart**.

Using the command line...

- From the ServletExec ISAPI directory, execute the batch file STOP_IIS.BAT, to completely stop IIS.

Using the Internet Services Manager administrative tool...

- Just click **Restart IIS**.

1.2.2 AS Product Only

If you've installed ServletExec/AS, the ServletExec process starts independently of the web server.

Note

There are subdirectories for each installed ServletExec/AS instance within the ServletExec/AS installation directory. The subdirectories are named *se-<instance-name>*. There is a separate StartServletExec.bat file (Windows) or StartServletExec shell script (UNIX) for each ServletExec/AS instance within the *se-<instance-name>* subdirectories.

To start ServletExec/AS on Windows 2008/Vista/2003/XP/2000**Using Control Panel...**

1. In Control Panel, click **Services**.
2. Click **ServletExec-<instance-name>**, and then click **Start**.

Using the Command Line...

- From the *se-<instance-name>* directory, execute the StartServletExec.bat file. This technique is sometimes helpful for troubleshooting SE startup/shutdown problems.

Note: This option will cause the SE AS instance to stop running when you logoff windows unless javaw.exe is used.

To start ServletExec/AS (UNIX)

- From the `ServletExecAS` directory, execute the `StartServletExec` shell script.

1.3 Verifying Your ServletExec Installation

The ServletExec installers and installation scripts can make all of the needed changes to your system to allow you to run servlets and JSP. If the ServletExec installer for your web server completed successfully, and you allowed it to modify the web server configuration files (if prompted), there is no need for additional manual configuration.

To verify your ServletExec installation

1. Restart your web server. If you installed ServletExec/AS, start the ServletExec/AS instance as described in Section 1.2. *See Section 1.2.1.1 for Microsoft IIS-specific instructions.*
2. After restarting your web server (and starting the ServletExec/AS instance, if necessary), verify your ServletExec installation using the included sample servlets. Invoke the sample servlets by entering the following URLs in your browser:

<http://localhost/servlet/DateServlet>

<http://localhost/servlet/TestServlet>

Note

If your browser is running on a different machine than your web server, replace `localhost` in the above URLs with the host name or IP address of your web server. The `/servlet/` prefix alias in the above URLs must be all lowercase and must not end with an “s”.

Important

Do not configure a `/servlet/` Virtual Directory for your web server--if you do, ServletExec will not work properly.

The above examples illustrate the simplest way to invoke servlets. Because these servlets did not require any configuration, we refer to them as “unconfigured servlets.”

If you’re unable to successfully execute these sample servlets, you should verify your installation using the ServletExec 6.0 Installation Guide. You can download the Installation Guide from:

http://www.newatlanta.com/products/servletexec/self_help/docs/index.jsp

Also, check the online Technical Support FAQ for additional help:

http://www.newatlanta.com/biz/c/products/servletexec/self_help/faq/home

1.4 Accessing the ServletExec Admin UI

You can configure nearly all ServletExec features from the browser-based administration user interface (referred to as the “ServletExec Admin Pages” or the “ServletExec Admin UI” throughout this guide).

To open or access the ServletExec Admin UI from a web browser

- Type this URL into the **Address** field and press **Enter**:

`http://localhost/servletexec/admin`

If you’re accessing or opening the ServletExec Admin UI for the first time, your browser display will appear similar to the one shown in Figure 1, which illustrates the Admin UI for ServletExec/ISAPI.

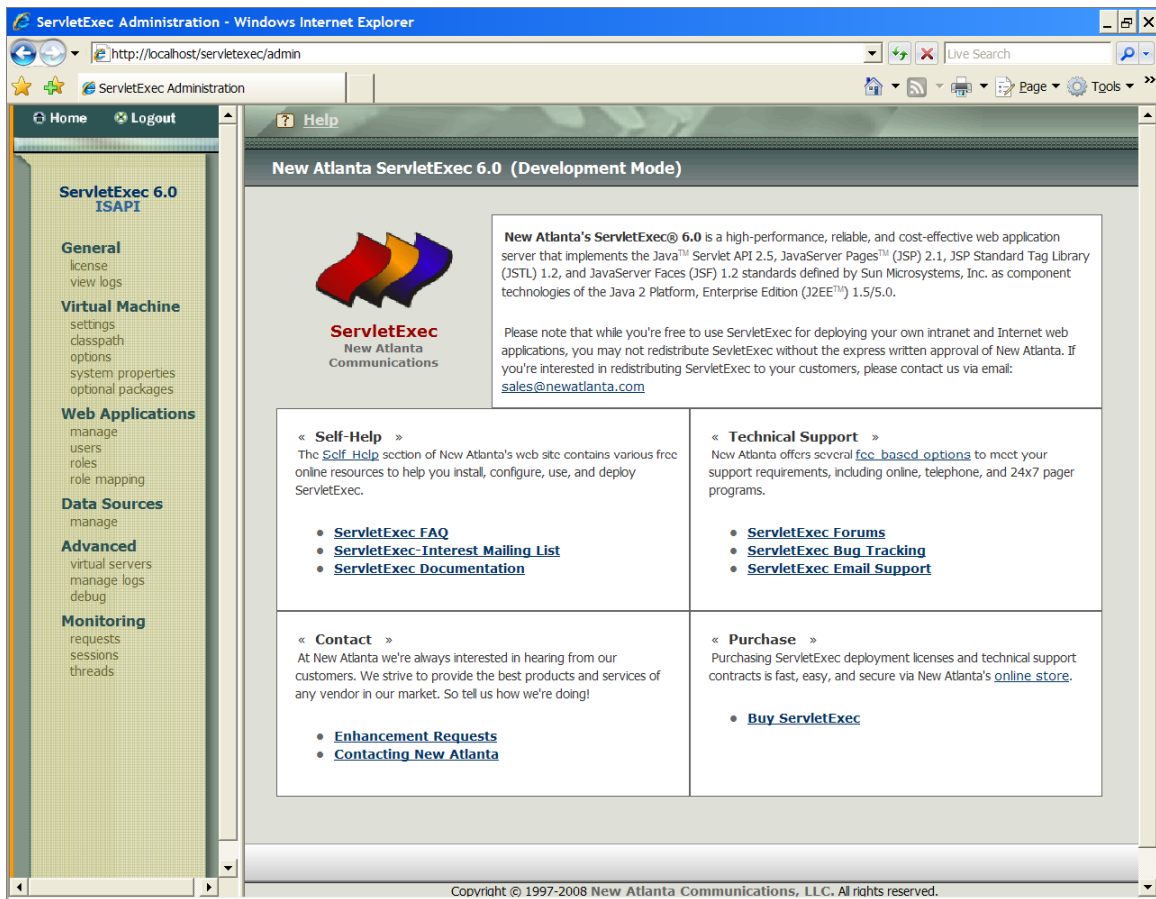


Figure 1. ServletExec/ISAPI Admin UI

1.5 Licensing ServletExec

In development mode, the License page appears by default when you launch ServletExec.

To bring up the ServletExec License and Security page

- Under **General** on the ServletExec Admin UI menu, click **license**.

Note

In development mode, ServletExec supports a maximum of three concurrent client requests. This allows you to evaluate ServletExec's features before purchasing a license.

The screenshot shows a web form for entering a license key. At the top, there is a text input field labeled 'License Key:'. Below it, the 'Mode:' is set to 'Development'. A red warning message states: 'In Development mode, you may use ServletExec for development only with the following restrictions:'. Two bullet points follow: 'ServletExec will process a maximum of 3 concurrent client requests.' and 'The "Allowed IPs" feature on the configure virtual server admin page is not available.'. At the bottom of the form is a button labeled 'Enter License Key'.

Figure 2. License Section of the License Page

To license ServletExec

1. Enter the license key and click **Enter License Key**.
2. Please review the ServletExec Software License Agreement (SLA) and abide by its terms and conditions. If you have any questions regarding the SLA, please contact us via email at sales@newatlanta.com

Important

You must purchase a separate license key for each physical server machine on which ServletExec is installed. ServletExec 6.0 is licensed on a per-CPU basis, so make sure you have purchased the appropriate license for your machine.

1.6 ServletExec Admin UI Security

The ServletExec admin runs as a web application and therefore can be protected using standard web application security. You can find more information about web applications and their settings in Chapter 3. It is also possible to restrict access to the SE Admin UI based on the IP address of the browser/client. For more details about that please see sections 1.6.3 or 10.7 of this document.

1.6.1 Admin UI Authentication

The two authentication-specific features of the admin that you can change are the method of authentication and the use of Secure Socket Layers (SSL) to access the admin. By default, FORM authentication is used and SSL is not required. If you want to change either of these settings, this is how.

1.6.1.1 Change the Authentication Method

By default, FORM authentication is used but a user can easily change this to BASIC or CLIENT-CERT on the "Configure Access Control" web application admin page. If this is switched to BASIC and ServletExec is running behind Microsoft IIS then the user must make sure that Basic Authentication is enabled in IIS (See the ServletExec 6.0 Installation Guide for instructions on enabling Basic Authentication for Microsoft IIS.).

Changing Authentication Method

If you decide to change the authentication method for the ServletExec administrator, you can do so as follows:

1. Log into the ServletExec administrator and click the Manage link from under Web Applications.
2. Click the web.xml link next to the ServletExec application name.
3. Select Access Control from underneath the security menu. You'll see a screenshot like Figure 3

Authentication Method:	<input type="radio"/> NONE <input type="radio"/> BASIC <input checked="" type="radio"/> FORM <input type="radio"/> CLIENT-CERT
Realm Name:	<input type="text"/> (valid for BASIC only)
Login Form:	<input type="text" value="/admin/login.jsp"/> (valid for FORM only)
Error Page:	<input type="text" value="/admin/login.jsp?failed=true"/> (valid for FORM only)
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

Figure 3. Web Application Security Access Control Page

4. Select the new authentication method and click the submit button.

If CLIENT-CERT authentication is selected then the email address that appears in the subject of the client certificate **that's already installed in your browser** is used to identify the client. In order for CLIENT-CERT authentication to work, the web server ServletExec is running behind must be configured to ask the client for a client certificate.

Setting the Admin to Require SSL Connections

By default, SSL is not required to access the ServletExec Admin UI. To require SSL, the transport guarantee setting should be set to CONFIDENTIAL in the web application security constraint that protects the admin pages. *You should only use this setting if your web server software (IIS, Apache, SunONE, or SJSWS) actually supports SSL, otherwise you will lock yourself out of the SE admin pages.*

Changing the Transport Guarantee

1. Log into the ServletExec administrator and click the Manage link from under Web Applications.
2. Click the web.xml link next to the ServletExec application name.
3. Select the constraints link from underneath security.
4. Click the EntireApplication link. You will see a page that contains Figure 4.

Data Transport Constraint	
Transport Guarantee	Description
<input checked="" type="radio"/> NONE <input type="radio"/> CONFIDENTIAL	<input type="text"/>

Figure 4. The Data Transport Constraint Box

5. Change the Transport Guarantee from NONE to CONFIDENTIAL. Click the submit button to save your changes.

1.6.2 Setting the Security Roles and Users

There is a special Container Role called *ServletExecAdminRole* that allows for access to the ServletExec Admin UI. Any Container Users that are a member of this role will have access to the admin UI. There is one default Container User with a username of *admin* and no password. You'll probably want to change that for security purposes. You can find more specific information on changing these roles in Section 3.5.6

1.6.3 Allowing Specific IPs to Access the Admin UI

You can restrict the IP addresses that have access to the Admin UI on the configure Virtual Server page.

Restricting IP Addresses

1. Log into the ServletExec admin and click Virtual Servers under the advanced menu.
2. Click on the default Virtual Server.

Configure Virtual Server	
Server Name:	<input type="text" value="default"/>
Allowed IPs:	<input type="text" value="*.*.*.* [*:*:*:*:*:*:*]"/>
Context Sharing:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

Figure 5. The Virtual Server Page

3. Enter a list of IP addresses that will be allowed to access the server. A comma is used to separate elements in the list. The specified IP addresses may include the “*” character to represent all IP addresses in a sub-range. For example, the following list of IP addresses would allow access to the ServletExec Admin UI from a client with the IP address “168.121.97.133” (or any IP address in the range from 204.84.126.1 to 204.84.126.255):

168.121.97.133,204.84.126.*

IPv4 addresses must be of the form x.x.x.x (where x is in the range 0-255).
IPv6 addresses must be of the form [y:y:y:y:y:y] (where y is a 4-digit hexadecimal number)

4. Click the submit button to save your changes.

Note

You can always access the ServletExec Admin UI from the local machine on which ServletExec is installed regardless of the Allowed IPs setting. If you wish to deny access from the local machine, remove the servletexec web application. Removing this web application prevents access to the ServletExec Admin UI from any machine; therefore, you will be required to edit the ServletExec configuration files by hand with a text editor after removing this web application.

1.7 Viewing Logs

From the View Logs page, you can check the ServletExec logs. You can view the logs for a web application’s servlets from the individual web application’s admin pages.

The screenshot shows a web interface titled "View Logs". It contains a form with the following elements:

- Log File:** A dropdown menu currently showing "access.log".
- Display Last:** A text input field containing the number "25", followed by the word "Entries".
- Buttons:** Two buttons at the bottom: "View Log File" and "Reset".

Figure 6. View Logs Page**To view the contents of ServletExec’s log files**

1. Under **General** on the ServletExec Admin UI menu, click **View Logs**.
2. In the **Log Files** list, select a log file.

See the table below for information on each log file.

Log File	Description
ServletExec.log	<ul style="list-style-type: none"> ▪ Resides in installation directory ▪ Contains informational and error messages from ServletExec ▪ Contains output from calls to <code>System.out</code> and <code>System.err</code>
ServletExecNative.log	<ul style="list-style-type: none"> ▪ Created by ISAPI only ▪ Resides in installation directory ▪ Contains informational and error messages generated by native code portions of ServletExec
access.log	<ul style="list-style-type: none"> ▪ Created by ServletExec/AS built-in web server only ▪ Resides in logs directory ▪ Contains information about requests received by the built-in web server

Table 2. ServletExec Log Files

1.7.1 Logging Options

The ServletExec Logging page gives you options for changing the number of backups to keep of the `ServletExec.log` and `Servlet.log` files, changing their rollover limit, and changing the location of log files.

The screenshot shows the 'ServletExec Logging' configuration page. It is organized into two main sections, each with a header and several input fields. The first section, 'ServletExec.log', has a directory field set to 'C:\SE_5.0\'', a 'Max ServletExec.log Backups' field set to '9', and a 'ServletExec.log Rollover Limit' field set to '100' K Bytes. The second section, 'Servlet.log', has a directory field set to 'C:\SE_5.0\Servlet Logs\'', a 'Max Servlet.log Backups' field set to '9', a 'Servlet.log Rollover Limit' field set to '100' K Bytes, and a 'Servlet.log Buffer Limit' field set to '40'. At the bottom of the form are two buttons: 'Submit' and 'Reset'.

Figure 7. ServletExec Logging Page

To use ServletExec's advanced logging options

- Under **Advanced** on the Admin UI menu, click **manage logs**.

The following logging options are configurable, as described in Table 3:

Logging Option	Description
ServletExec.log Directory	<ul style="list-style-type: none"> ▪ Specifies the directory to which ServletExec.log file and backups are written.
Max ServletExec.log Backups	<ul style="list-style-type: none"> ▪ Every time ServletExec restarts, a new ServletExec.log file gets created and the old files renamed ServletExec.log.<n>, where <n> is a number starting with 1 for the most recent backup.
ServletExec.log Rollover Limit	<ul style="list-style-type: none"> ▪ Specified in kilobytes. ▪ When the ServletExec.log file reaches specified size, a new ServletExec.log file gets created and the old file backed up as ServletExec.log.1 (all other backups have <n> incremented and the oldest backup file is deleted if necessary).
Servlet.log Directory	<ul style="list-style-type: none"> ▪ Specifies the directory to which Servlet.log and backups are written.
Max Servlet.log Backups	<ul style="list-style-type: none"> ▪ Every time ServletExec restarts, a new Servlet.log file gets created and the old files renamed ServletExec.log.<n>, where <n> is a number starting with 1 for the most recent backup.
Servlet.log Rollover Limit	<ul style="list-style-type: none"> ▪ Specified in kilobytes. ▪ When the Servlet.log file reaches specified size, a new ServletExec.log file gets created and the old file backed up as ServletExec.log.1 (all other backups have <n> incremented and the oldest backup file is deleted if necessary).
Servlet.log Buffer Limit	<ul style="list-style-type: none"> ▪ Specifies the number of Servlet.log entries to be buffered in memory before being written to the Servlet.log file.

Table 3. Advanced Logging Options Described

1.8 Technical Support

If you're having difficulty installing or configuring ServletExec, check the online Tech Support FAQ:

http://www.newatlanta.com/biz/c/products/servletexec/self_help/faq/home

You may also want to consider subscribing to the ServletExec-Interest mailing list, a user-supported discussion forum for ServletExec users:

http://www.newatlanta.com/products/servletexec/self_help/archive_search/index.jsp

Details regarding free and paid support options, including online-, telephone-, and pager-based support are available from the ServletExec web site:

<http://www.newatlanta.com/support/servletexec/index.jsp>

2

2. Java Virtual Machine

Setting Java Virtual Machine (VM) options for ServletExec

The Java VM offers several standard and non-standard options which may be configured to effect various aspects of how it runs. These include classpath & System Property settings as well as many others documented here:

<http://java.sun.com/j2se/1.5.0/docs/tooldocs/windows/java.html#options> (for Windows)

<http://java.sun.com/j2se/1.5.0/docs/tooldocs/solaris/java.html#options> (for Unix/Linux)

- With ServletExec/ISAPI, the Java Virtual Machine (VM) settings (including the global classpath) are configured using the Main ServletExec Admin UI.
- With ServletExec/AS, you can view the VM settings and classpath with the ServletExec Admin UI, but you can only modify them by manually editing the `StartServletExec.bat` file (Windows) or the `StartServletExec` script (UNIX).

See Section 3.7 in the ServletExec 6.0 Installation Guide for information on editing the `StartServletExec.bat` file (Windows).

See Section 4.7 in the ServletExec 6.0 Installation Guide for information on editing the `StartServletExec` script (UNIX).

ServletExec must be reinitialized (i.e. stopped and then restarted) before VM Option changes will take effect. See section 1.2 of this document for more details about reinitializing ServletExec.

2.1 Viewing System Properties

You can view the properties for the Java Virtual Machine by clicking **System Properties** under **Virtual Machine**. A sample display of properties is shown in Figure 8 (below).

Java Virtual Machine (VM) Properties	
Property Name	Property Value
awt.toolkit	sun.awt.windows.WToolkit
file.encoding	Cp1252
file.encoding.pkg	sun.io
file.separator	\
java.awt.graphicsenv	sun.awt.Win32GraphicsEnvironment
java.awt.printerjob	sun.awt.windows.WPrinterJob
java.class.path	C:\Program Files\New Atlanta\ServletExec AS\lib\servlet-api.jar;C:\Program Files\New Atlanta\ServletExec AS\lib\jsp-api.jar;C:\Program Files\New Atlanta\Atlanta\ServletExec AS\lib\ServletExecAdmin.jar;C:\Program Files\New Atlanta\ServletExec AS\lib\el-api.jar;C:\Program Files\New Atlanta\ServletExec AS\Atlanta\ServletExec AS\lib\jstl.jar;C:\Program Files\New Atlanta\ServletExec AS\lib\appserv-jstl.jar;C:\Program Files\New Atlanta\ServletExec AS\lib\activa AS\lib\mail.jar;C:\Documents and Settings\matt\My Documents\java\jars\JTurbo.jar;C:\Program Files\New Atlanta\ServletExec AS\se-bi\classes
java.class.version	50.0
java.endorsed.dirs	C:\java1.6\jre\lib\endorsed
java.ext.dirs	C:\java1.6\jre\lib\ext;C:\WINDOWS\Sun\Java\lib\ext
java.home	C:\java1.6\jre
java.io.tmpdir	C:\DOCUME~1\matt\LOCALS~1\Temp\
java.library.path	C:\java1.6\bin.;C:\WINDOWS\Sun\Java\bin;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;Files\Common Files\Adaptac Shared\System;C:\Program Files\Microsoft SQL Server\80\Tools\BINN;C:\Program Files\Metroworks\CodeWarrior\Other Metrow Files\Metroworks\CodeWarrior\Win32-x86 Support\Libraries\Runtime\Libs\MSL_All-DLLs;C:\Program Files\Metroworks\CodeWarrior\Win32-x86 Support\Librar Files\putty;C:\ant\bin;C:\Program Files\QuickTime\QTSystem\;C:\Sun\SDK_ee5_u3\bin
java.naming.factory.initial	com.newatlanta.servletexec.InitialContextFactory

Figure 8. Java Virtual Machine System Properties

2.2 Viewing Java VM Settings

This section explains how to view (or modify) the Java Virtual Machine settings.

Note for ServletExec/AS Users

As noted on the previous page, you must manually edit either the `StartServletExec.bat` file (Windows) or the `StartServletExec` script (UNIX) to modify Java VM settings.

See Section 3.7 in the ServletExec 6.0 Installation Guide for information on editing the `StartServletExec.bat` file (Windows).

See Section 4.7 in the ServletExec 6.0 Installation Guide for information on editing the `StartServletExec` script (UNIX).

To view (or modify) Java VM settings for ServletExec/ISAPI (Windows only)

- Under **Virtual Machine** on the Admin UI menu, click **Settings**. This brings up the Java Virtual Machine (VM) Settings page as shown in Figure 9.

The screenshot shows the 'Java Virtual Machine (VM) Settings' page. At the top, there is a 'Help' icon. The main content is organized into several sections:

- Current Java VM:** Java HotSpot(TM) Server VM 1.6.0_01 from Sun Microsystems Inc.
- Current VM Info:** mixed mode
- Operating System:** Windows XP 5.1 on x86
- Selected Java VM:** Radio buttons for 'Sun HotSpot Client' (unselected) and 'Sun HotSpot Server' (selected).
- Current Heap Size:** 4032 KB (4 MB)
- Unused/Free Heap:** 2890 KB (2 MB)
- Maximum Possible Heap:** 64832 KB (64 MB)
- Heap Summary:**
 - * The heap is currently **06.22%** of its maximum possible capacity.
 - * **71.69%** of the current heap size is available for use.
 - * The current heap will grow **1507.94%** if needed, before reaching its maximum possible size.
- Minimum Heap Size:** Input field with '4096' and 'K Bytes' label.
- Maximum Heap Size:** Input field with '65536' and 'K Bytes' label.
- Verbose GC:** Radio buttons for 'Enabled' (unselected) and 'Disabled' (selected).
- Verbose:** Radio buttons for 'Enabled' (unselected) and 'Disabled' (selected).
- Class GC:** Radio buttons for 'Enabled' (selected) and 'Disabled' (unselected).
- System Output:** Text input field containing 'com.newatlanta.servletexec.SESystemOutputStream'.
- System Error:** Text input field containing 'com.newatlanta.servletexec.SESystemOutputStream'.

At the bottom, there are three buttons: 'Submit', 'Reset', and 'Set Defaults'.

Figure 9. Java Virtual Machine (VM) Settings Page

2.3 Selecting and Configuring a Java VM

The information at the top of the Java Virtual Machine (VM) Settings page indicates which VM was selected during initialization, as Figure 9 shows. ServletExec supports 2 “flavors” of Sun’s Java VM. They are:

- Sun’s HotSpot™ Client Performance Engine
- Sun’s HotSpot™ Server Performance Engine

Notes

For changes to the selected VM to take effect, you must reinitialize (i.e. stop & then restart) ServletExec. To learn how to reinitialize ServletExec/ISAPI or SE/AS, read the beginning of this chapter.

2.4 Defining the Global Classpath

The Java VM classpath defines the directories and JAR files in which the VM will search when trying to load class (.class) files or resource (.properties) files. By default,

ServletExec sets the global classpath to include all JAR files required by ServletExec. With SE ISAPI, these required JARs do not show up on the VM Classpath page of the SE Admin UI. They can be viewed on the JVM System Properties page as shown in figure 8 above (the property named `java.class.path`).

With SE/ISAPI you can add more directories or JARs to the global classpath using the ServletExec Admin UI. This is actually required if you have a common set of class files (third-party libraries for example) that are shared by many web applications. With SE/AS you'd do this by manually editing the StartServletExec batch/script. Either way you must then restart ServletExec for the global classpath changes to take effect.

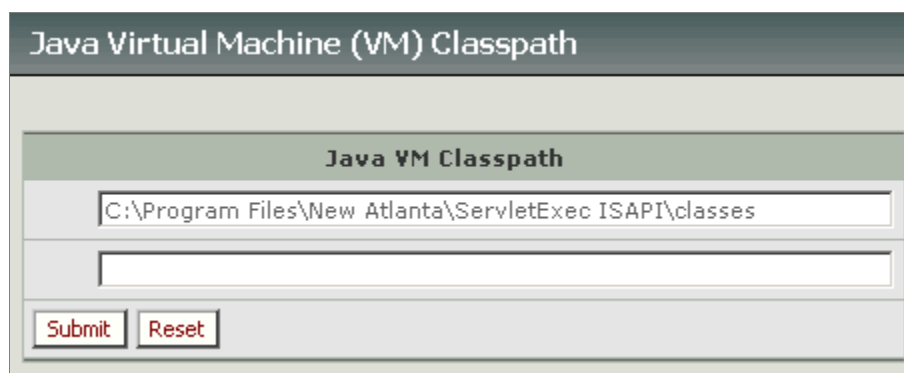


Figure 10. Java Virtual Machine (VM) Classpath Page

To view or modify ServletExec's Java VM classpath

- Under **Virtual Machine** on the Admin UI menu, click **classpath**. The Java Virtual Machine (VM) Classpath page will appear, as shown in Figure 10.
 - You must specify directories that are added to the classpath as full paths. When adding `.jar/.zip` files to the classpath, you must specify the full path to the file.
 - Windows: (a) Use the “\” character as the file separator when specifying directory paths. (b) Use the full UNC name and not a mapped drive letter to specify paths to network drives.¹ Use the following form.


```
\\<machine name>\<drive share name>\<path to file>
```
 - UNIX: Use the “/” character as the file separator when specifying directory paths.

Note

To learn how to reinitialize ServletExec/ISAPI or SE/AS, read the beginning of this chapter.

¹ This is because mapped network drive letters are associated with particular users. ServletExec ISAPI runs inside the IIS web server process, which runs as a Windows service and therefore is not aware of network drive letter mappings.

2.5 Setting Java VM Options (SE/ISAPI Only)

The Java Virtual Machine (VM) Options page (**SE/ISAPI only**) provides a convenient method for setting VM options. Treat its data entry field (see Figure 11) as you would the command line with the `java` command. For detailed information on the `java` command and available options, see the documentation included with the JDK (the start of this chapter gives links to that documentation).

To set Java VM options

1. Under **Virtual Machine** on the Main ServletExec Admin UI, click **options**.
2. Enter the options you want to configure for the VM and click **Submit**.

Figure 11. Java Virtual Machine (VM) Options Page (SE/ISAPI only)

2.6 Using Optional Packages

Optional Packages is Sun's new term for *Extensions* (or *Standard Extensions*). They are packages of Java classes and associated native code that extend the functionality of the core platform.

You install Optional Packages by adding the JAR file containing them to one of two locations, depending on how you want to make them available:

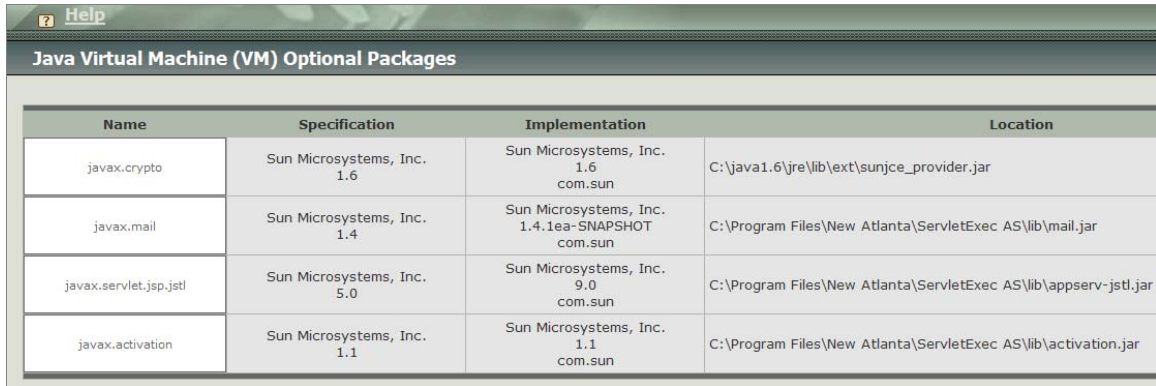
- Main ServletExec Classpath—This makes the Optional Package available to all ServletExec web applications.
- JDK's Ext directory—This makes the Optional Package available to all ServletExec web applications, as well as all Java applications that use that JDK.

ServletExec's Java Virtual Machine (VM) Optional Packages page lists all installed optional packages.

See Figure 12 for a sample page.

Note

Older Optional Packages that do not include the `Manifest.mf` file (which must also contain an *Extension-Name* entry) will not appear on this page, but should still function normally.



Name	Specification	Implementation	Location
javax.crypto	Sun Microsystems, Inc. 1.6	Sun Microsystems, Inc. 1.6 com.sun	C:\java1.6\jre\lib\ext\sunjce_provider.jar
javax.mail	Sun Microsystems, Inc. 1.4	Sun Microsystems, Inc. 1.4.1ea-SNAPSHOT com.sun	C:\Program Files\New Atlanta\ServletExec AS\lib\mail.jar
javax.servlet.jsp.jstl	Sun Microsystems, Inc. 5.0	Sun Microsystems, Inc. 9.0 com.sun	C:\Program Files\New Atlanta\ServletExec AS\lib\appserv-jstl.jar
javax.activation	Sun Microsystems, Inc. 1.1	Sun Microsystems, Inc. 1.1 com.sun	C:\Program Files\New Atlanta\ServletExec AS\lib\activation.jar

Figure 12. Java Virtual Machine (VM) Optional Packages Page

3

3. Web Applications

Deploying and configuring web applications with ServletExec

A web application, as defined by Java Servlet API 2.5, is a collection of servlets, HTML pages, Java class files, and other resources that can be deployed as a single entity, either as a structured hierarchy of directories or as an archive file (a Web Archive, or “.war” file).

A web application is rooted at a specific URL **context path** within a web server. For example, a catalog application could be located at the URL:

```
http://www.mycorp.com/catalog/
```

All requests that have the **/catalog** URL prefix right after the hostname will be routed to the ServletExec instance on which that web application has been deployed.

See the Java Servlet API 2.5 Specification for a detailed discussion of web applications:

<http://java.sun.com/products/servlet/>

3.1 Web Application Overview

A web application can consist of the following items:

- Servlets
- JavaServer Pages
- Utility Classes
- Static documents (HTML pages, GIF/JPEG image files, etc.)
- Client-side applets, beans, and classes
- Descriptive metadata that ties all of the above elements together

3.1.1 Directories

A web application exists as a structured hierarchy of directories. The root of this hierarchy serves as a document root for serving files that are part of this context. For example, for a web application configured with a URL context path of `/catalog/`

within a web server, the `index.html` file located at the base of the web application directory hierarchy will be served by `ServletExec` to satisfy a request for `/catalog/index.html`.

3.1.1.1 WEB-INF directory

A special directory exists within the application directory hierarchy named "WEB-INF". This directory contains all things related to the application that aren't in the document root of the application. In other words, the WEB-INF directory is not part of the public document tree of the application, and no file contained in the WEB-INF directory may be served directly to the client.

The contents of the WEB-INF directory are:

- `/WEB-INF/web.xml` deployment descriptor (as of Servlet 2.5, this file is optional)
- `/WEB-INF/classes/` directory for servlet and utility classes (`.class` files)
- `/WEB-INF/lib/` directory for Java archive (`.jar`) files containing servlets, beans, and other utility classes needed by the web application

3.1.2 Packaging

Web applications can be packaged and signed, using the standard Java Archive tools, into Web Archive format (`.war`) files. For example, an application for issue tracking could be distributed in an archive with the filename `issuetrack.war`.

3.2 The "default-app" web application

`ServletExec` automatically creates a web application named "default-app" the first time it initializes. This web application is configured with a context path of "/" and is configured to let the web server serve static content. This web application is different from other web applications in that all content (static and dynamic) is located under the web server's root and additional document directories. Because the default-app is configured to let the web server serve static content, request listeners will not receive request events for static content. Welcome files are also processed in a special manner by the default-app. If the default-app does not have any welcome files configured then the web server handles the processing of welcome files. If the default-app does have welcome files configured then it handles the processing of welcome files and the web server's welcome files (i.e. default documents) are ignored.

A useful resource for understanding the nuances of the default-app is SE FAQ #278: http://www.newatlanta.com/c/products/servletexec/self_help/faq/detail?faqId=278

3.3 Using ServletExec's Web Application Admin UI

The Manage Web Applications page serves as the “home” for ServletExec’s web application administration features. From this page (*see Figure 13*), you can:

- Add a web application
- Configure a web application’s ServletExec settings
- Edit a web application’s web.xml settings
- Reload a web application
- Remove a web application

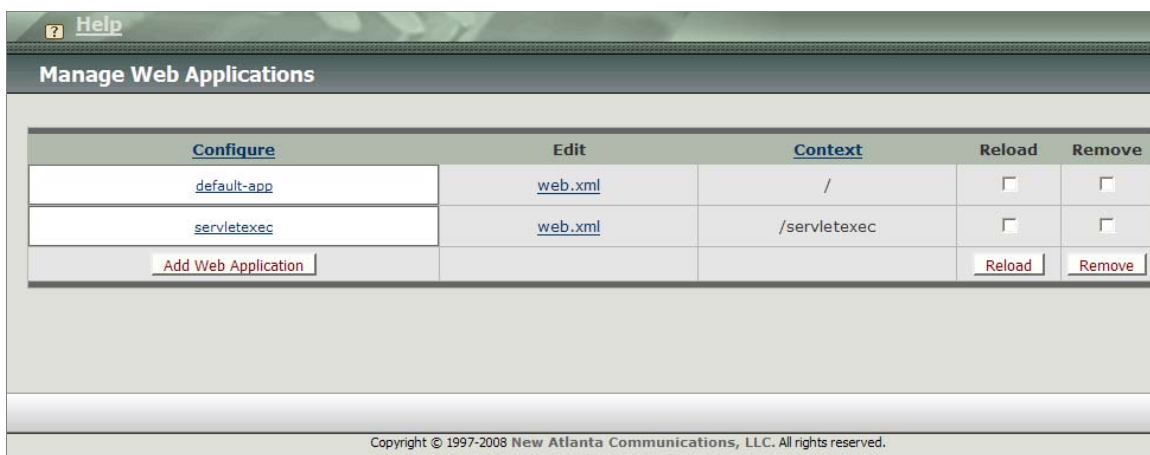


Figure 13. Manage Web Applications Page

3.4 Running the Example Web Application

The Example web application illustrates many of the features of web applications, including full source code for the examples.

Tip

Careful study of this example will answer many questions you may have regarding the use of web applications within ServletExec.

To run the Example web application

1. Under **Web Applications** on the Admin UI, click **manage** to bring up the Manage Web Applications page (*see Figure 13*).
2. Click **Add Web Application**. This brings up the Add a Web Application page (*see Figure 14*).

Configure a Web Application

Application Name:

URL Context Path:

Location:

ServletExec Extensions: Automatic Enabled Disabled

File Caching: Enabled Disabled

Static Content: ServletExec Web Server

Figure 14. Add a Web Application Page

- On the Add a Web Application page, you can specify any value for **Application Name** and **URL Context Path**. We recommend using *Example* and */example/*, respectively.

Note

Avoid using white space and non-alpha characters (such as characters for comma, semi-colon, quotation marks, etc.) in the Application Name.

- In **Location**, enter the full path to the `exampleWebApp` directory and click **Submit**. This returns you to the Manage Web Applications page.
 - For ServletExec/ISAPI, the default location of this directory is:

C:\Program Files\New Atlanta\ServletExec ISAPI\Examples\exampleWebApp

Manage Web Applications

Configure	Edit	Context	Reload	Remove
default-app	web.xml	/	<input type="checkbox"/>	<input type="checkbox"/>
Example	web.xml	/example	<input type="checkbox"/>	<input type="checkbox"/>
servletexec	web.xml	/servletexec	<input type="checkbox"/>	<input type="checkbox"/>

Figure 15. Manage Web Applications Page (with new application added)

- Under **Edit** on the Manage Web Applications page, click `web.xml` for your newly added web application. This opens a new browser window specific to your example

web application (see Figure 16). This new browser window also provides a new set of menu options that allow you to view or change the web.xml descriptor settings.

See Section 3.5.7 for more information on this topic.

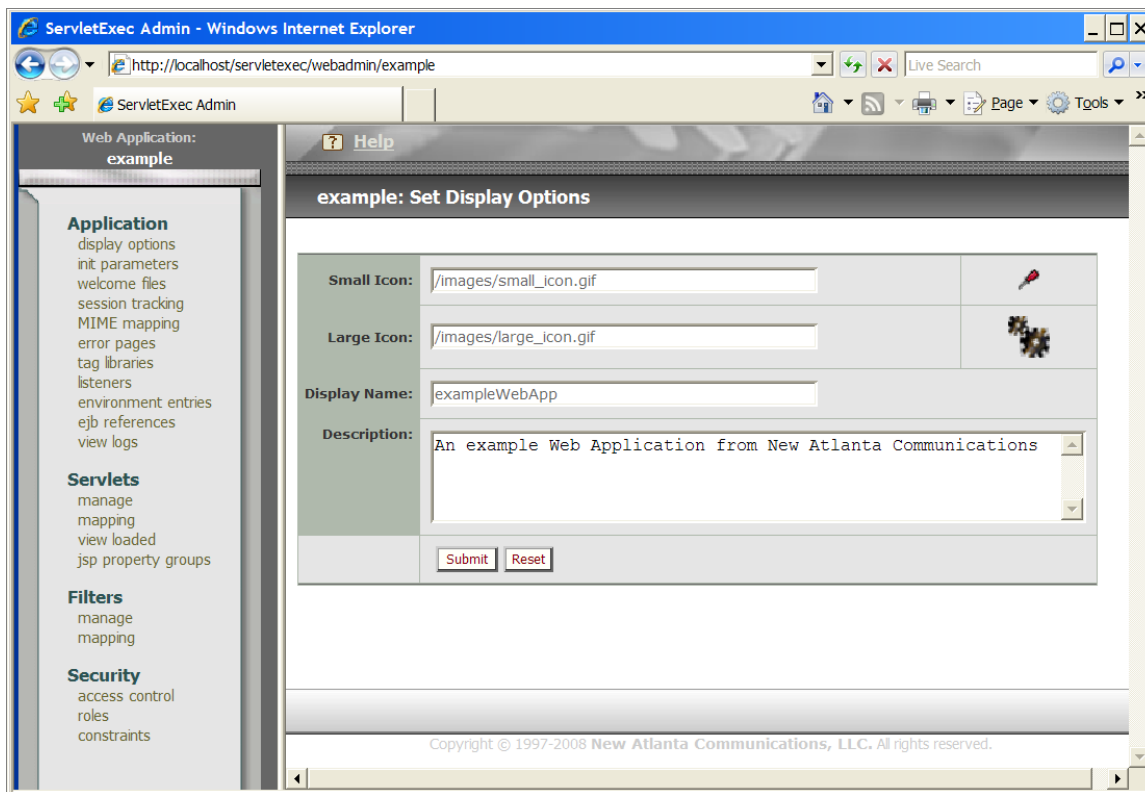


Figure 16. Web Application Admin UI Start Page

6. Enter the following URL to bring up the index page of the Example web application:
<http://localhost/example/>
Or just click on the “/example” link in the “Context” column (see Figure 15).
7. From the resulting index page, click each link to explore and learn from the Example web application.

3.5 Deploying Web Applications

The process of deploying web applications involves two or more of the following steps.

- Creating a web application
- Auto-deploying a web application
- Placing the web application on the ServletExec server’s file system
- Manually deploying the web application using the ServletExec Admin UI

- Configuring the web application for ServletExec
- Editing the web application's web.xml file.

3.5.1 Creating a Web Application

If you have yet to create the web application you wish to deploy, you can use the ServletExec Admin UI to create it. If you use this feature, you will not have to install or add the new web application, as this process accomplishes both those steps.

So, if you already have a web application, skip this step and go on to either Auto Deploying it (*see Section 3.5.2*) or Manually Deploying it (*see Section 3.5.4*)

To create a web application using the Admin UI

1. Under **Web Applications** on the Admin UI, click **manage** to bring up the Manage Web Applications page (*see Figure 19*).
2. Click **Add Web Application**. This brings up the Add a Web Application page (*see Figure 20*).
3. On the Add a Web Application page, enter an **Application Name** and a **URL Context Path**.

Notes

- The **Name** and **Context Path** must both be unique.
 - You can use any value in these fields but do avoid using white space and non-alpha characters (such as characters for comma, semi-colon, quotation marks, etc.).
-
4. In **Location**, enter the full path to the directory you wish to install the newly created web application into and click **Submit**. This returns you to the Manage Web Applications page. If the directory does not yet exist, you can have ServletExec create it for you in the following step, based on the information you enter into **Location**. This will bring up an error page alerting you to the missing directory (see Figure 17).
 5. If you're presented with the Manage Web Applications page, then you are done. If you're presented with the error page alerting you to the missing directory (or other missing resources), click **Deploy/Create Missing Resources**. This brings up the Manage Web Applications page displaying a successful result.

Application: MyCustomWebapp

! ERROR: Unable to Add the Web Application. It lacks the necessary Directory Structure and/or Files.

Problem Number	The Problem
1	The context root folder c:\myWebapps\MyCustomWebapp for the Web Application named MyCustomWebapp was not found.
2	The optional (but recommended) file c:\myWebapps\MyCustomWebapp\WEB-INF\web.xml was not found.

Create web.xml file

Figure 17. Missing Resources Error Page

Manage Web Applications

* Web application was added successfully.

Configure	Edit	Context	Reload	Remove
default-app	web.xml	/	<input type="checkbox"/>	<input type="checkbox"/>
MyCustomWebapp	web.xml	/onlineStore	<input type="checkbox"/>	<input type="checkbox"/>
servletexec	web.xml	/servletexec	<input type="checkbox"/>	<input type="checkbox"/>

Figure 18. Manage Web Applications Page After Successfully Creating a Web Application

3.5.2 Auto-Deploying a Web Application

Using ServletExec's auto-deployment feature reduces the process of deployment to two steps. You will need to re-initialize ServletExec afterwards, though.

If you use this method, you will not need to do any of the other sub-sections inside 3.5 Deploying Web Applications.

To auto-deploy a web application

1. Drop the web application's .war file (or top-level folder if your app is delivered in open/exploded directory structure) into a configured virtual server's folder residing in the webapps folder. You can use either the folder named default or any other folder for a

configured virtual server. For example: `C:\Program Files\New Atlanta\ServletExec AS\se-1\webapps\default`

2. Stop ServletExec (if it is running), then start ServletExec, to see the web application appear on the Manage Web Applications page (see 1.2 Starting ServletExec).

See the READ ME file inside the web apps folder for more information on auto-deployment.

3.5.3 Placing a Web Application onto the file system

Before deploying a web application onto ServletExec, you first have to place it on the physical server where ServletExec resides.

Placing a web application on the server machine

1. Copy the web application to any location on the file system of the server on which ServletExec is installed.

Important

Do not put the web application in a web server's document root. Doing so can create a security risk. A good practice is to place your application in a "neutral" location such as `C:\myWebapps\`. The only possible exception is if the context path of the webapp is a single forward slash (/) as is the case for the webapp named "default-app".

2. Check to make sure the ServletExec process (or the web server process for the ISAPI versions of ServletExec) has read permission enabled for the web application. Write permission will also be needed for the web.xml file. If web.xml is inside a .war file then that .war file must have read/write permission. See the note at the end of section 3.5.7 for more details.

3.5.4 Manually deploying a Web Application

Here you will begin using the ServletExec Admin UI in the deployment process. This section covers the installation of the web application into ServletExec.

To manually deploy a web application

1. Under **Web Applications** on the Admin UI, click **manage** to bring up the Manage Web Applications page (see *Figure 19*).
2. Click **Add Web Application**. This brings up the Add a Web Application page (see *Figure 20*).

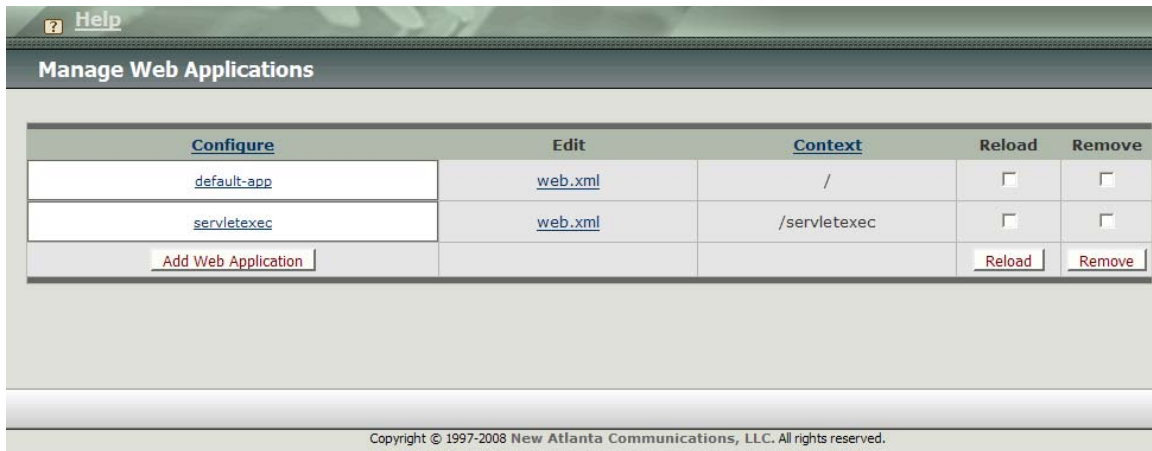


Figure 19. Manage Web Applications

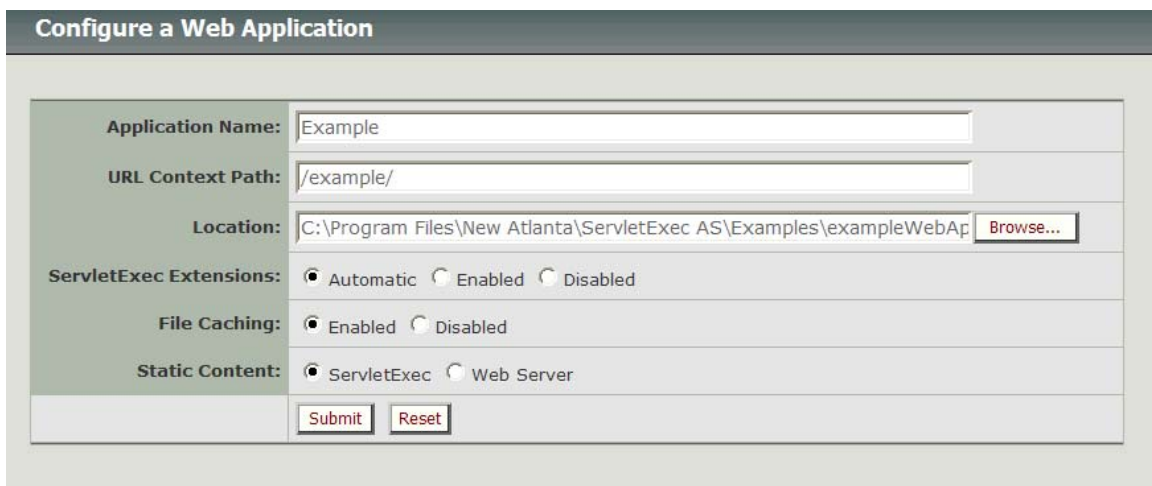


Figure 19. Add a Web Application Page

- On the Add a Web Application page, enter an **Application Name** and **URL Context Path**.

Note

You can use any value for the Name & Context Path but do not use white space or non-alpha characters (such as characters for comma, semi-colon, quotation marks, etc.)

- In **Location**, enter the full path to your webapp and click **Submit**. This returns you to the Manage Web Applications page. *The File Browsing feature* for defining the Location of a webapp is supported with Internet Explorer & Firefox browsers, but not with Netscape browser. The file browsing feature relies on JavaScript. This means that some browser brands/versions may need their JavaScript settings to be modified. For example we found that our installation of Firefox v2.0.0.3 was not rendering the "Ok" or "Cancel" buttons on the file browsing popup window. We were able to correct this by configuring our browser as follows: **Tools - Options - Content** click on the **Advanced...** button which pertains to JavaScript and check/enable the *Hide the status bar* setting. By default the only root that can be browsed is the root for the folder named by the user .home JVM System

Property. This may change depending upon the user account under which ServletExec is running. A JVM System Property named `com.newatlanta.servletexec.browseRoots` can be used to configure exactly which File System Roots/Partitions may be browsed. For example:

`-Dcom.newatlanta.servletexec.browseRoots=C,D,E`
(now only drives C, D, and E may be browsed)

`-Dcom.newatlanta.servletexec.browseRoots=*`
(now all visible drives may be browsed)

`-Dcom.newatlanta.servletexec.browseRoots=none`
(this turns off the file browsing feature)

3.5.4.1 ServletExec Extensions

Enabling **ServletExec Extensions** provides several additional session tracking options, and enables the configuration of external Java libraries. See Section 3.5.5 for details regarding these additional configuration options.

Important

Enabling ServletExec extensions temporarily makes the web application no longer portable to other servlet engine brands. To restore portability to other servlet engines, you will need to disable the **ServletExec Extensions** configuration option prior to porting the webapp.

Static File Caching

When adding a web application (or afterward), you have the option of having the web application's static files cached in memory. Enabling this feature can improve performance but obviously uses more memory. Figure 20 shows the panel with this feature that you see while adding a web application.

Note

You can also enable/disable **File Caching** after deploying a web application by clicking the name of the application under **Configure** on the Manage Web Applications page, which brings up the page shown in Figure 21.

Configure a Web Application

Application Name: Example

URL Context Path: /example/

Location: C:\Program Files\New Atlanta\ServletExec AS\Examples\exampleWebAp

ServletExec Extensions: Automatic Enabled Disabled

File Caching: Enabled Disabled

Static Content: ServletExec Web Server

Figure 20. Configure a Web Application

3.5.4.2 Static Page Serving

In addition, from the Add a Web Application page or the Configure Web Application for ServletExec page, you can choose to have the web server serve static pages, which can improve performance. To do this, make sure to select **Disabled**. This is actually disabling ServletExec from serving the static pages.

Note

This option is not available when ServletExec/AS is running behind the built-in web server. If you do use this option when SE AS is running behind a production-grade web server such as IIS or Apache, then the SE AS Instance must be running on the same machine as the web server software. That's the only way that the web server software can "see" the static resources in order to serve them. The only exception to that rule is the webapp named "default-app" whose static content can be served by the web server, even if the SE AS instance is running on a machine other than the webserver.

Important

Disabling Serve Static Pages also disables File Caching.

Important

Disabling Serve Static Pages causes the web server to serve static content. Therefore, request listeners will not receive request events for static pages.

3.5.5 ServletExec Web Application Extensions

Enabling **ServletExec Extensions** for a web application (see Section 3.5.4.1) provides several additional session tracking options, and enables the configuration of external Java libraries.

3.5.5.1 Session Tracking

A standard web application only supports setting the Session Timeout session tracking parameter, as illustrated in Figure 32. However, you can configure additional session tracking options by enabling **ServletExec Extensions** for your web application, as illustrated in Figure 22.

The screenshot shows the 'default-app: Configure Session Tracking' interface. It features a list of configuration options:

- Session Tracking:** Enabled Disabled
- URL Rewriting:** Enabled Disabled
- Protocol Switch Rewriting:** Enabled Disabled
- Cookies:** Enabled Disabled
- Persistence:** Enabled Disabled
- Maximum Residents:**
- Maximum Sessions:**
- Swap Directory:**
- Swap Interval:** Seconds
- Invalidation Interval:** Seconds
- Embedded SessionID Prefix:**
- Manager Class Name:**
- Session Timeout:** Minutes
- Cookie Name:**
- Cookie Comment:**
- Cookie Domain:**
- Cookie Maximum Age:**
- Cookie Path:**
- Cookie Secure:** True False

At the bottom right, there are **Submit** and **Reset** buttons.

Figure 21. Additional Session Tracking Options

3.5.5.2 External Libraries

For a standard web application, Java libraries (JAR files) are loaded from the web application's `WEB-INF\lib` directory. With **ServletExec Extensions** enabled, it is possible to configure your web application to load Java libraries (`.class` files and/or `.jar` files) that are external to the web application.

Figure 23 illustrates the Admin UI page for configuring external libraries. External libraries may be configured by specifying either the full path to a JAR file or a directory. If a JAR file is specified, then the JAR file is added to the web application's classpath. If a directory is specified, then ServletExec adds all `.class` files and then all JAR files found in the directory to the web application's classpath.

When loading Java classes, ServletExec searches for class files in the following order: (1) the web application's WEB-INF\classes; (2) JAR files in the web application's WEB-INF\lib directory; (3) external libraries; and, (4) the ServletExec Global classpath (i.e. the value of the System Property named `java.class.path`). See SE FAQ #355:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=355

Figure 22. External Libraries

3.5.6 Setting Up Roles, Users, and Role Mapping

You can manage ServletExec's local security domain using the Roles, Users and Role Mapping features. Each has its own page, which is available under **Web Applications** on the Admin UI.

The first thing you'll need to do is set up one or more Container Roles. Next, you'll associate Container Users with that Container Role. Finally, you'll need to map the Container Role to an existing Web Application Role.

See the Java Servlet 2.5 Specification, which is available at the following URL, for more information on web application security.

<http://java.sun.com/products/servlet/download.html>

3.5.6.1 Roles

From the Manage Roles page (*see Figure 24*), you can...

- Create a new role
- Create a new user
- Access the Modify Role page for any existing role
- Access the Modify User page for any associated user
- Access the admin pages for any associated web application
- Remove an existing role (if it does not have a web application association)

Manage Roles			
Roles	Users	Applications	Remove
Developers	Matt Gregg Paul	--none--	<input type="checkbox"/>
GreatScientists	Enrico Leonardo Albert	--none--	<input type="checkbox"/>
NotSoGreatScientists	Larry Curly Moe	--none--	<input type="checkbox"/>
<input type="button" value="New Role"/>	<input type="button" value="New User"/>		<input type="button" value="Remove"/>

Figure 23. Manage Roles Page

To create a new Container Role

1. From either the Manage Roles page or the Manage Users page, click **New Role**.
2. On the Create a New Role page, enter the **Role Name** and press TAB.
3. Enter a comma-separated list of users for this security role into **Associated Users** and click **Submit**.

Create a New Role	
Role Name:	<input type="text"/>
Users:	<input type="text"/>
	<input type="button" value="Submit"/> <input type="button" value="Reset"/>

Figure 24. Create a New Role Page

Important

Before a Container Role can be used, you must map it to an existing Web Application Role.

See the following sections for information on both steps.

See Section 3.5.6.2 for information on creating users.

To modify an existing Role, User, or Web Application

- On the Manage Roles page, click the name of the Role, User or Web Application in the appropriate column to bring up its configuration page.

To remove a Container Role

1. On the Manage Roles page, select the **Remove** checkbox for the appropriate Role and click **Submit**. If the **Remove** checkbox isn't available, go to Step 2 to disassociate the Role from its web application.
2. Under **Web Applications** on the Admin UI, click **role mapping**. This brings up the Manage Role Mapping page.
3. In the **Container Role** list for the appropriate **Web Application Name**, click **<Not Mapped>** and click **Submit**. ServletExec will not allow a container role with existing web application roles mapped to it, to be removed.
4. Return to the Manage Roles page and the **Remove** checkbox should be available.

3.5.6.2 Users

Again, before you can use a Container Role, it has to have Users associated with it.

From the Manage Users page (*see Figure 26*), you can...

- Create a new user
- Create a new role
- Access the Modify User page for any existing user
- Access the Modify Role page for any associated role
- Remove an existing user

Manage Users		
Users	Roles	Remove
Albert	GreatScientists	<input type="checkbox"/>
Curly	NotSoGreatScientists	<input type="checkbox"/>
Enrico	GreatScientists	<input type="checkbox"/>
Gregg	Developers	<input type="checkbox"/>
Larry	NotSoGreatScientists	<input type="checkbox"/>
Leonardo	GreatScientists	<input type="checkbox"/>
Matt	Developers	<input type="checkbox"/>
Moe	NotSoGreatScientists	<input type="checkbox"/>
Paul	Developers	<input type="checkbox"/>
<input type="button" value="New User"/>	<input type="button" value="New Role"/>	<input type="button" value="Remove"/>

Figure 25. Manage Users Page

To create a new user

1. From either the Manage Users page or the Manage Roles page, click **New User**.
2. On the Create a New User page (*see Figure 27*), enter the **User Name**.
3. Enter a comma-separated list of security roles for this user into **Associated Roles** and press.
4. Enter a Password (and Confirm Password) for the user (this is optional) and click **Submit**.

Create a New User	
User Name:	<input type="text"/>
Roles:	<input type="text"/>
Password:	<input type="password"/>
Confirm Password:	<input type="password"/>
	<input type="button" value="Submit"/> <input type="button" value="Reset"/>

Figure 26. Create a New User Page

See Section 3.5.6.1 for information on creating roles.

To modify an existing Role, User, or Web Application

- On the Manage Users page, click the name of the Role, User or Web Application in the appropriate column to bring up its configuration page.

To remove a user

- On the Manage Users page, select the **Remove** checkbox for the appropriate User and click **Submit**.

3.5.6.3 Role Mapping

As mentioned earlier in this section, before a Container Role can be used, you have to map it to an existing Web Application Role.

From the Manage Role Mapping page (*see Figure 28*), you can...

- Map a previously defined **Container Role** to an existing **Web Application Role**
- Modify an existing **Role Mapping**

To map a previously defined Container Role to a Web Application Role

- In the **Container Role** list in the Manage Role Mapping page, click the appropriate **Container Role** for the **Web Application Role** you are mapping, and click **Submit**.

See Section 3.5.7.4 if you have not yet defined any Web Application Roles.

To modify an existing Role Mapping

- In the **Container Role** list in the Manage Role Mapping page, click the appropriate new **Container Role** for the **Web Application Role** you wish to change, and click **Submit**.

Web Application Name	Web Application Role	Container Role
TripleExpresoWithCream	GreatScientists	GreatScientists

Figure 27. Manage Role Mapping Page

3.5.7 Editing the Web Application's `web.xml` Settings

When you click `web.xml` under **Edit** on the Manage Web Applications page, ServletExec opens a new browser window that provides configuration pages specific to that web application, as shown in Figure 29. You can also switch to these pages directly by using a URL with the following form:

```
http://localhost/servletexec/webadmin/<web app name>
```

Here are two examples:

```
http://localhost/servletexec/webadmin/catalog
```

```
http://localhost/servletexec/webadmin/example
```

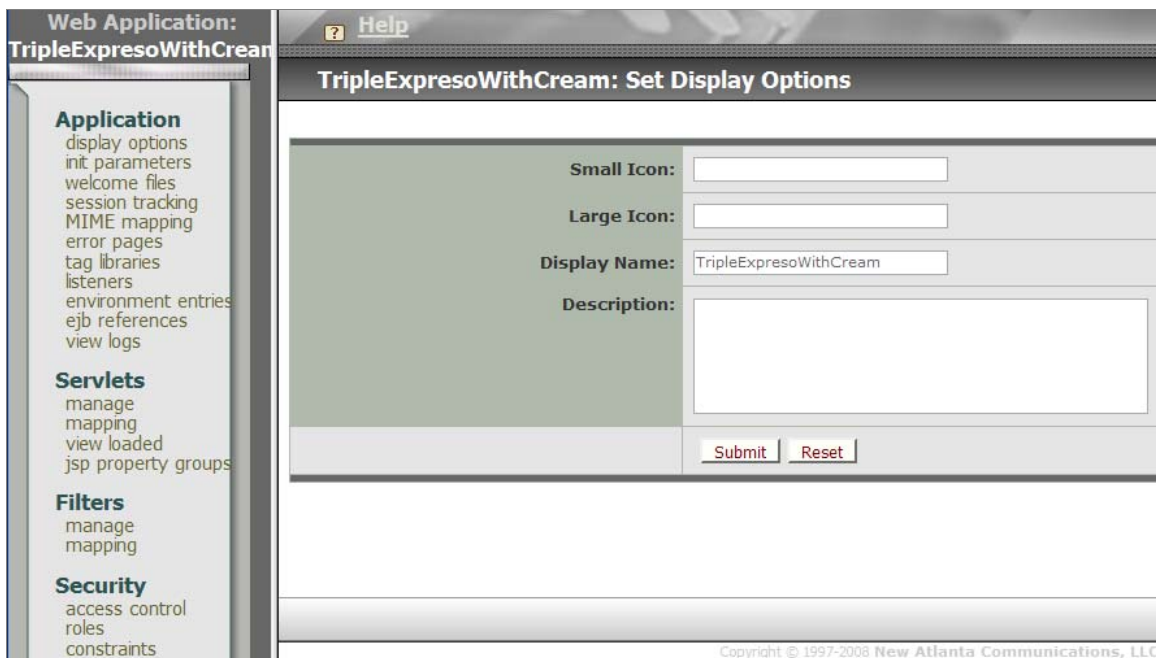


Figure 28. Web Application Admin UI Start Page

The application settings available from these pages correspond to the elements of the `web.xml` deployment descriptor for a web application.

Note

Any changes you make to a web application's settings using the web application admin pages automatically updates `web.xml` (even if that `web.xml` is inside a `.war` file). Conversely, any manual edits to the `web.xml` file will be reflected in a web application's admin pages (after that web application has been reloaded).

3.5.7.1 Application

The Application menu of a web application's admin pages provides various configuration options, including:

- Setting display options
- Managing initialization parameters
- Setting welcome files
- Configuring session tracking
- Setting MIME mapping (for responses to requests for static files with specific extensions)
- Managing error pages
- Managing tag libraries
- Managing listeners (Listeners are special classes that provide notification of Servlet Context and HTTP Session events)
- Managing Environment Entries
- Managing EJB references
- Viewing logs

If the **ServletExec Extensions** option is enabled for your webapp, then the **External Libraries** options is also available. See Section 3.5.4.1 for instructions for setting the **ServletExec Extensions** option; see Section 3.5.5.2 for information regarding the **External Libraries** option.

To set display options

- From the Set Display Options page (*see Figure 29*), enter any or none of the optional choices for defining how your web application appears in a browser.

To create a new initialization parameter

- From the Manage Initialization Parameters page (*see Figure 30*), enter data into **Name** and **Value**, and then click **Submit**.

To delete an initialization parameter

- From the Manage Initialization Parameters page (*see Figure 30*), clear all fields and click **Submit**.

Name	Value	Description
<input type="text"/>	<input type="text"/>	<input type="text"/>
init param 1 for application	param value 1 for application	description 1
init param 2 for application	param value 2 for application	description 2

Submit Reset

Figure 29. Manage Initialization Parameters Page

To add a welcome file

- From the Set Welcome Files page (*see Figure 31*), enter the filename into **Welcome Files** and click **Submit**.

To remove a welcome file

- From the Set Welcome Files page (*see Figure 31*), clear its field and click **Submit**.

Note

When ServletExec receives a request for a directory, it looks for welcome files in top-to-bottom order in that directory.

TripleExpresoWithCream: Set Welcome Files

Welcome Files

Submit Reset

Figure 30. Set Welcome Files Page

To configure session tracking

- From the Configure Session Tracking page (*see Figure 32*), choose the settings you wish for the web application and click **Submit**.

See *Help* in the lower portion of the page for specific information on each option.

A standard web application only supports setting the Session Timeout value, as illustrated in Figure 32. However, you can configure additional session tracking options by enabling **ServletExec Extensions** for your web application. See Section 3.5.4.1 for instructions for setting the **ServletExec Extensions** option; see Section 3.5.5.1 for information regarding the additional session tracking options.

The screenshot shows a web form titled "TripleExpresoWithCream: Configure Session Tracking". The form has a header bar with the title. Below the header, there is a section for "Session Timeout" with a text input field containing the number "30" and the text "Minutes" to its right. At the bottom of this section are two buttons: "Submit" and "Reset".

Figure 31. Configure Session Tracking Page

To add a MIME mapping

- From the Set MIME Mapping page (see Figure 33), enter an **Extension** and **MIME Type**, and click **Submit**.

To remove a MIME mapping

- From the Set MIME Mapping page (see Figure 33), clear all fields and click **Submit**.

The screenshot shows a web form titled "TripleExpresoWithCream: Set MIME Mapping". The form contains a table with two columns: "Extension" and "Mime type". The table has several rows of mappings. At the bottom of the form are two buttons: "Submit" and "Reset".

Extension	Mime type
gif	image/gif
htm	text/html
html	text/html
jpeg	image/jpeg
jpg	image/jpeg
txt	text/plain

Figure 32. Set MIME Mapping Page

Note

The MIME Mapping page settings determine which MIME type a client receives when a static page is served by ServletExec (not when a static page is served by the web server software).

To add an error page

- From the Manage Error Pages page (*see Figure 34*), enter an **HTTP Error Code** or **Java Exception** and corresponding **Location**, and then click **Submit**.

Notes

- Adding an error page specifies which page gets returned to the client when the specified error code or exception occurs in a servlet (not in a JSP). See the comments in the `throwException.jsp` (in the `exampleWebApp` which comes with SE) for more details.
- Error pages are matched in descending (top to bottom) order.

TripleExpresoWithCream: Manage Error Pages	
HTTP Error Code	Location
<input type="text"/>	<input type="text"/>
404	/jsp/errorHandler.jsp
Java Exception	Location
<input type="text"/>	<input type="text"/>
javax.servlet.ServletException	/jsp/errorHandler.jsp
<input type="button" value="Submit"/>	<input type="button" value="Reset"/>

Figure 33. Manage Error Pages Page

To remove an error page

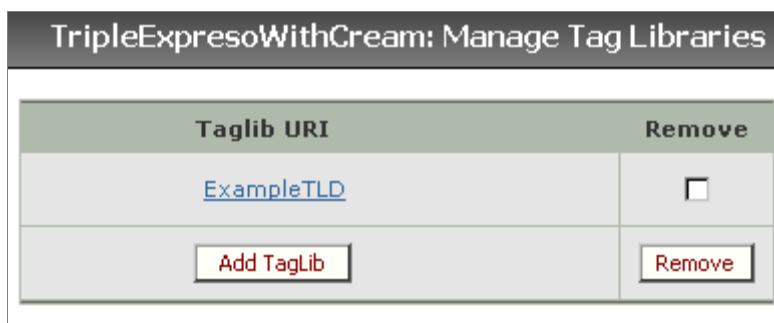
- From the Manage Error Pages page (*see Figure 34*), clear one or both fields and click **Submit**.

Tip

Using the following feature (Add a Tag Library) greatly simplifies JSP management by eliminating the need to make changes to JSPs after a *tag library descriptor* (TLD) is moved or renamed.

To add a tag library

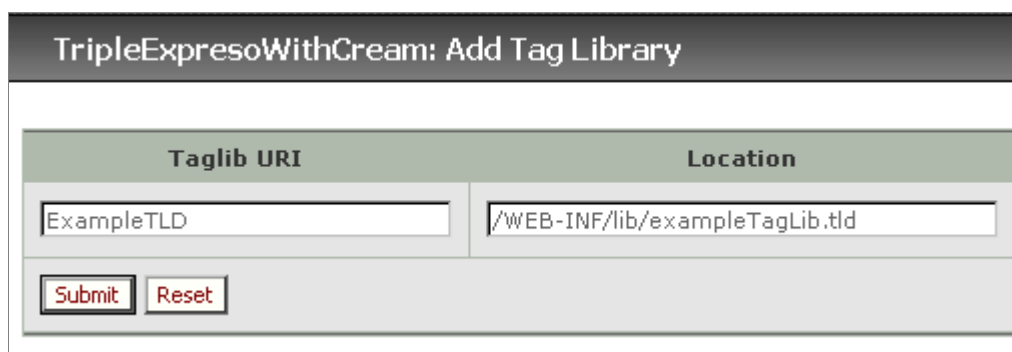
1. From the Manage Tag Libraries page (see Figure 35), click **Add Taglib**. This brings up the Add a Tag Library page.

**Figure 34. Manage Tag Libraries Page**

2. From the Add a Tag Library page (see Figure 36), enter a tag library name of your choosing into **Taglib URI** and its context-relative path into **Taglib Location**, and then click **Submit**.

Tip

- Adding a tag library to a web application's XML settings in ServletExec allows you to use the Taglib URI in a JSP Taglib directive to specify a TLD instead of having to specify the path to the TLD.

**Figure 35. Add a Tag Library Page****To edit a tag library**

1. From the Manage Tag Libraries page (see Figure 35), click the name under **Taglib URI** of the tag library you are editing. This brings up the Edit Tag Library page.
2. From the Edit Tag Library page (see Figure 37), make your changes and click **Submit** to save or **Reset** to restore the prior settings.

Figure 36. Edit Tag Library Page

To remove a tag library

- From the Manage Tag Libraries page (*see Figure 35*), select the **Remove** checkbox for the tag library you are removing, and click **Submit**.

Tip

Use the following feature (Listeners) to be notified whenever lifecycle events occur.

To add a listener

- From the Manage Listeners page (*see Figure 38*), enter the class name into **Listeners** and click **Submit**.

To remove a listener

- From the Manage Listeners page (*see Figure 38*), clear its field and click **Submit**.

Figure 37. Manage Listeners Page

To add an Environment Entry

1. From the Manage Environment Entries Page (*Figure 41*) click the **Add Environment Entry** button. This brings up the Add an Environment Entry page (*Figure 39*).

basicSessionManagementExample : Add an Environment Entry	
Name:	<input type="text" value="TestEnvironmentEntry"/>
Value:	<input type="text" value="Test"/>
Type:	<input type="text" value="java.lang.String"/>
Description:	<input type="text"/>
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

Figure 38. Add an Environment Entry

2. Enter the Name, Value, Type, and Description.
3. Click the **Submit** button to create the Environment Entry

To edit an Environment Entry

1. From the Manage Environment Entries Page (*Figure 41*) click on the Environment Entry name. This brings up the Edit an Environment Entry page (*Figure 40*).

basicSessionManagementExample : Edit Environment Entry	
Name:	<input type="text" value="TestEnvironmentEntry"/>
Value:	<input type="text" value="Test"/>
Type:	<input type="text" value="java.lang.String"/>
Description:	<input type="text"/>
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

Figure 39. The Edit Environment Entry Page

2. Change the Name, Value, Type, and Description values as necessary.
3. Click the **Submit** button to save your changes.

To remove an Environment Entry

1. From the Manage Environment Entries Page (*Figure 41*), check the remove checkbox next the environment entries you want to remove.
2. Click the **Remove** button to delete the Environment Entries.

Environment Entry	Remove
TestEnvironmentEntry	<input type="checkbox"/>

Figure 40. Manage Environment Entries

To add an EJB Reference

1. From the Manage EJB references page, click the **Add EJB Reference** button. This will bring up the add EJB Reference page (*Figure 42*).

Name:	<input type="text"/>
Type:	<input type="text"/>
Home:	<input type="text"/>
Remote:	<input type="text"/>
Link:	<input type="text"/>
Description:	<input type="text"/>
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

Figure 41. Add EJB Reference Page

2. Enter the name, type, home, remote, link, and description values.
3. Click **Submit** to save your EJB reference.

To edit an EJB Reference

1. From the Manage EJB references page, click the name of the EJB reference you want to edit. You should see something similar to Figure 43.

Figure 42. Edit an EJB Reference

2. Modify the name, type, home, remote, link, and description values as needed.
3. Click **Submit** to save your EJB reference.

To remove an EJB Reference

1. From the Manage EJB References Page, check the remove checkbox next to the EJB references you want to remove, as shown in Figure 44.

Figure 43. The Manage EJB References Page

2. Click the **Remove** button to delete the EJB references.

To view the web application's logs

- From the View Logs page (see Figure 45), select the log in the **Log Files** list, and then click **Submit**. The selected log display appears on the same page (see Figure 46).

Figure 44. View Logs Page

Displaying the last 25 entries of Servlet.log.	
[Wed Apr 16 13:38:17 EDT 2003]	ApplAdminServlet: init
[Wed Apr 16 13:39:08 EDT 2003]	ApplAdminServlet: destroy
[Wed Apr 16 13:39:08 EDT 2003]	JspServlet: destroy
[Wed Apr 16 13:39:08 EDT 2003]	Logging stopped
[Wed Apr 16 13:39:10 EDT 2003]	Logging started
[Wed Apr 16 13:39:22 EDT 2003]	JspServlet: init
[Wed Apr 16 13:39:54 EDT 2003]	JspServlet: destroy
[Wed Apr 16 13:39:55 EDT 2003]	Logging stopped
[Wed Apr 16 13:39:55 EDT 2003]	Logging started
[Thu Apr 24 11:03:26 EDT 2003]	Logging started
[Thu Apr 24 11:11:16 EDT 2003]	ApplAdminServlet: init
[Thu Apr 24 12:06:07 EDT 2003]	JspServlet: init

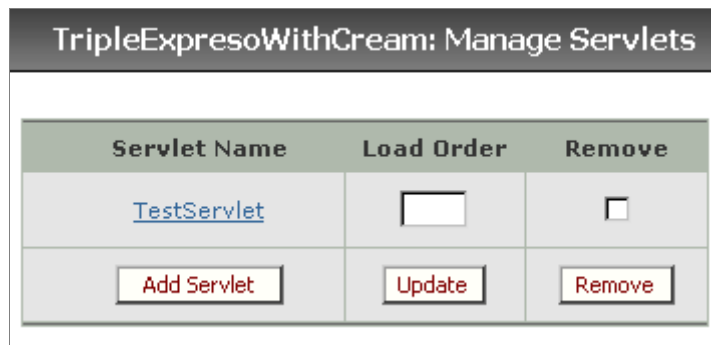
Figure 45. Sample Log**3.5.7.2 Web Application Servlets**

The Servlets menu of a web application's Admin UI provides three primary configuration choices. They are:

- Managing web application servlets (adding, editing, removing and setting load order)
- Mapping web application servlets to URL patterns
- Viewing all loaded web application servlets

To add a web application servlet

1. From the Manage Servlets page (see Figure 47), click **Add Servlet**. This brings up the Add a Servlet page.



Servlet Name	Load Order	Remove
TestServlet	<input type="text"/>	<input type="checkbox"/>
<input type="button" value="Add Servlet"/>	<input type="button" value="Update"/>	<input type="button" value="Remove"/>

Figure 46. Manage Web Application Servlets Page

2. From the Add a Servlet page (see Figure 48), complete all required (and optional, if desired) fields using the *Help* as a guide, and click **Submit**.

TripleExpresoWithCream: Add Servlet

Servlet Name:	<input type="text" value="TestServlet"/>	
Servlet Class:	<input type="text" value="TestServletWebApp"/>	
JSP File:	<input type="text"/>	
Display Name:	<input type="text" value="TripleExpresoWithCreams test Servlet"/>	
Description:	<input type="text" value="An example"/>	
Small Icon:	<input type="text"/>	<input type="text"/>
Large Icon:	<input type="text"/>	<input type="text"/>

Initialization Parameters		
Name	Value	Description
<input type="text" value="Init Arg 2"/>	<input type="text" value="Init Arg Value 2"/>	<input type="text" value="2nd init arg for TestServ"/>
<input type="text" value="Init Arg 3"/>	<input type="text" value="Init Arg Value 3"/>	<input type="text" value="1st init arg for TestServ"/>
<input type="text"/>	<input type="text"/>	<input type="text"/>

Security Role References		
Role Name	Role Link	Description
<input type="text"/>	<input type="text"/>	<input type="text"/>

Figure 47. Add a Web Application Servlet Page

To edit a web application servlet

1. From the Manage Servlets page (*see Figure 47*), click the name of the servlet you're editing and click **Submit**. This brings up the Edit Servlet page.
2. From the Edit Servlet page, make your changes and click **Submit** to save or **Reset** to restore the prior settings.

To change the load order for a web application servlet

- From the Manage Servlets page (see Figure 47), enter the appropriate value into **Load Order**, and click **Submit**.

Note

When a web application initializes, it loads only those servlets that have a specified load order. All other servlets load only when invoked.

To remove a web application servlet

- From the Manage Servlets page (see Figure 47), select the **Remove** checkbox for the appropriate servlet, and click **Submit**.

To map a web application servlet to a URL pattern

- From the Set Servlet Mapping page (see Figure 49), enter a **URL Pattern** and **Servlet Name**, and then click **Submit**.

To delete a servlet mapping

- From the Set Servlet Mapping page (see Figure 49), clear all fields and click **Submit**.

URL Pattern	Servlet Name
<input type="text"/>	<input type="text"/>
<input type="text" value="/exactAlias/"/>	<input type="text" value="ExactAliasServlet"/>
<input type="text" value="/prefixAlias/*"/>	<input type="text" value="PrefixAliasServlet"/>
<input type="text" value="*.Suffix"/>	<input type="text" value="SuffixAliasServlet"/>

Figure 48. Set Servlet Mapping Page

To view all loaded web application servlets

- From the web application's Admin UI, click **View Loaded** under **Servlets**.

servletexec: View Loaded Servlets	
Servlet Name	Servlet Information
AdminServlet	Provides an interface to administer ServletExec

Figure 49. View Loaded Web Application Servlets Page

3.5.7.3 Filters

Under the Filters menu of a web application's admin pages, you'll see two options. They are:

- Managing filters (adding, editing or removing)
- Mapping filters to URL patterns

To add a filter

1. From the Manage Filters page (*see Figure 51*), click **Add Filter**. This brings up the Add a Filter page.

TripleExpresoWithCream: Manage Filters	
Filter Name	Remove
TestFilter	<input type="checkbox"/>
<input type="button" value="Add Filter"/>	<input type="button" value="Remove"/>

Figure 50. Manage Filters

2. From the Add a Filter page (*see Figure 52*), complete all required (and optional, if desired) fields using the *Help* as a guide, and click **Submit**.

TripleExpresoWithCream: Add Filter

Filter Name:	<input style="width: 95%;" type="text" value="TestFilter"/>
Filter Class:	<input style="width: 95%;" type="text" value="TestWebAppFilter"/>
Display Name:	<input style="width: 95%;" type="text" value="TripleExpresoWithCreams test Filter"/>
Description:	<input style="width: 95%;" type="text" value="An example of a Web Application Filter"/>
Small Icon:	<input style="width: 95%;" type="text"/>
Large Icon:	<input style="width: 95%;" type="text"/>

Initialization Parameters		
Name	Value	Description
<input style="width: 95%;" type="text" value="Init Arg #2"/>	<input style="width: 95%;" type="text" value="Init Arg Value 2"/>	<input style="width: 95%;" type="text" value="2nd Init Arg for Te"/>
<input style="width: 95%;" type="text" value="Init Arg #1"/>	<input style="width: 95%;" type="text" value="Init Arg Value 1"/>	<input style="width: 95%;" type="text" value="1st Init Arg for Te"/>
<input style="width: 95%;" type="text"/>	<input style="width: 95%;" type="text"/>	<input style="width: 95%;" type="text"/>

Figure 51. Add a Filter Page

To edit a filter

1. From the Manage Filters page (*see Figure 51*), click the name of the filter you're editing. This brings up the Edit Filter page.
2. From the Edit Filters page, make your changes and click **Submit** to save or **Reset** to restore the prior settings.

To map a filter to a URL pattern

- From the Set Filter Mapping page (*see Figure 53*), enter a **URL Pattern** and **Filter Name**, a **Servlet Name** (must already appear on the Manage Servlets page) and **Filter Name**, and click **Submit**.

See Chapter 11 of the Java Servlet 2.4 Specification for detailed information on format.

TripleExpresoWithCream: Set Filter Mapping		
URL Pattern	Filter Name	Dispatchers
<input type="text"/>	<input type="text"/>	<input type="checkbox"/> INCLUDE <input type="checkbox"/> FORWARD <input type="checkbox"/> REQUEST <input type="checkbox"/> ERROR
Servlet Name	Filter Name	Dispatchers
<input type="text"/>	<input type="text"/>	<input type="checkbox"/> INCLUDE <input type="checkbox"/> FORWARD <input type="checkbox"/> REQUEST
<input type="button" value="Submit"/> <input type="button" value="Reset"/>		

Figure 52. Set Filter Mapping

To delete a filter mapping

- Clear all fields and click **Submit**.

3.5.7.4 Security

The Security menu of a web application's admin pages provides three primary settings pages. They are:

- Configuring access control (selecting an authentication method, defining the realm, specifying pages for the Form Based authentication method only)
- Defining web application roles
- Managing security constraints

Enabling secure access controls on any web application resource requires a minimum of:

- One security constraint
- One web application role
- One selected authentication method
- One defined user
- One user-to-role mapping

Note

Failure to define a user and role mapping prevents access to a set of resources.

See Section 3.5.6.2 for more information on setting up users.

See Section 3.5.6.3 for more information on role mapping.

See Chapter 12 of the 2.5 Java Servlet Specification for detailed information on web application security.

Although there is no required order for configuring web application security, you should define all web application roles that will be referenced by a security constraint before creating the security constraint. Otherwise, you'll have to stop during the process of creating the security constraint to define a role or roles.

To configure access control

1. From the Configure Access Control page (*see Figure 54*), choose one of the three **Authentication Methods** or click **NONE**. If you select the **CLIENT-CERT** method, click **Submit**. If you select one of the other methods, go on to Step 2.
2. Enter a **Realm Name** (only if you are choosing the **BASIC** method) and click **Submit** (this immediately activates the authentication method). If you are using another authentication method, go on to Step 3.
3. Enter the context-relative URL for the login form (**FORM** method only) into **Login Form**.
4. Enter the context-relative URL for the error page (**FORM** method only) into **Error Page** and click **Submit**. This immediately activates the authentication method.

TripleExpresoWithCream: Configure Access Control	
Authentication Method:	<input checked="" type="radio"/> NONE <input type="radio"/> BASIC <input type="radio"/> FORM <input type="radio"/> CLIENT-CERT
Realm Name:	<input type="text" value="Gregg's Testing"/> (valid for BASIC only)
Login Form:	<input type="text" value="login.html"/> (valid for FORM only)
Error Page:	<input type="text" value="login-error.html"/> (valid for FORM only)
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

Figure 53. Configure Access Control Page (Form-based setting)

5. A web application role is defined and known only within the context of a web application. Because a web application role is usually defined without the

knowledge of the user names being used in a real deployment environment, it is meaningless without being mapped to a container role (which “contains” container users).

To define a web application role

- From the Role Definitions page (*see Figure 55*), enter the name for the **Role** and an optional **Description**, and then click **Submit**.

Important

Web application security roles must be mapped to a container security role on the Manage Role Mapping page in the main ServletExec Admin UI.

Role Name	Description
GreatScientists	
NotSoGreatScientists	
Developers	

Submit Reset

Figure 54. Define Role Page

To delete a role

- Clear the contents in both **Role** and **Description**, and click **Submit**.

To add a security constraint

1. From the Manage Security Constraints page (*see Figure 56*), click **Add Constraint**. This brings up the Add Security Constraint page.

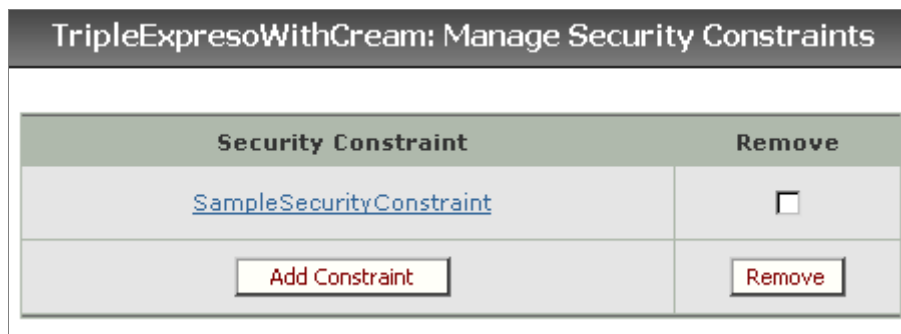


Figure 55. Manage Security Constraints Page

2. From the Add Security Constraint page (*see Figure 57*), enter a **Display Name**.
3. Specify the role or roles to grant access to the web resource collections in **Auth. Constraint**. If no roles are selected then no users will be able to access the resources specified in this security constraint.
4. Set the communication between client and server in **Data Constraint**.

Notes

- **NONE** indicates the web application does not require transport guarantees.
 - **CONFIDENTIAL** indicates that SSL is required.
-
5. In Web Resource Collections, specify the name of each web application resource the constraint will protect along with its associated URL pattern and HTTP access method, and click **Submit**. Not specifying an HTTP **method** allows all methods.

TripleExpresoWithCream: Add Security Constraint			
Display Name: <input type="text" value="SampleSecurityConstraint"/>			
Authorization Constraint			
Roles	Description		
<input type="checkbox"/> Unauthenticated access <input type="checkbox"/> All Application roles ("*") <input checked="" type="checkbox"/> Developers <input type="checkbox"/> GreatScientists <input type="checkbox"/> NotSoGreatScientists	<input type="text"/>		
Data Transport Constraint			
Transport Guarantee	Description		
<input checked="" type="radio"/> NONE <input type="radio"/> CONFIDENTIAL	<input type="text"/>		
Web Resource Collections			
Name	URL Patterns	Methods	Description
<input type="text" value="wrc-01"/>	<input type="text" value="/classified/*"/>	<input type="checkbox"/> GET <input type="checkbox"/> POST <input type="checkbox"/> HEAD <input type="checkbox"/> PUT <input type="checkbox"/> DELETE <input type="checkbox"/> OPTIONS <input type="checkbox"/> TRACE Custom HTTP Methods: <input type="text"/>	<input type="text"/>
<input type="button" value="Submit"/> <input type="button" value="Reset"/>			

Figure 56. Add Security Constraint Page

To edit a security constraint

1. From the Manage Security Constraints page (*see Figure 56*), click the name of the **Security Constraint** to edit.
2. From the Edit Security Constraints page, make your changes and click **Submit** to save.

3.5.8 Reloading a Web Application

Not using the SE Admin UI constitutes a *manual* change. If you *manually* change a web application in any way, you will need to Reload the web application into ServletExec.

To reload a web application

- From the Manage Web Applications page of the Admin UI, select the **Reload** check box for the web application you are reloading, and click **Reload**.

3.5.9 Removing/Undeploying a Web Application

If you ever need to delete a web application (permanently or temporarily), you will need to use the **Remove** feature. One example would be if you detected a security problem with a web application and wanted to remove it from operation until the problem was resolved.

Important

Even if you are permanently removing the web application from your machine, be sure to use the **Remove** feature within ServletExec before deleting the web application from your hard drive.

To remove a web application

- From the Manage Web Applications page of the Admin UI, select the **Remove** check box for the web application you are removing, and click **Remove**.
- If the removed application had been auto-deployed, you'll need to remove it from the auto-deploy folder. Otherwise SE will re-autodeploy it during the next SE startup. In other words, undeploying an auto-deployed webapp is a 2-step process.

4. Data Sources

Managing Data Sources with ServletExec

ServletExec allows you to add JDBC 2.0 data sources and manage them from within ServletExec. Using ServletExec's data sources management features also allows you to switch JDBC drivers or modify the settings for connecting to a database (user name and password, for example) without having to modify and recompile your servlets or JSPs.

4.1 Managing Data Sources

From the Manage Data Sources page (click **manage** under **Data Sources** on the Admin UI menu), you can view at-a-glance, all data sources that have been added to ServletExec, a brief description of each data source, and the availability of each data source. You can also perform the following operations:

- Add a data source
- Edit a previously added data source
- Remove a data source

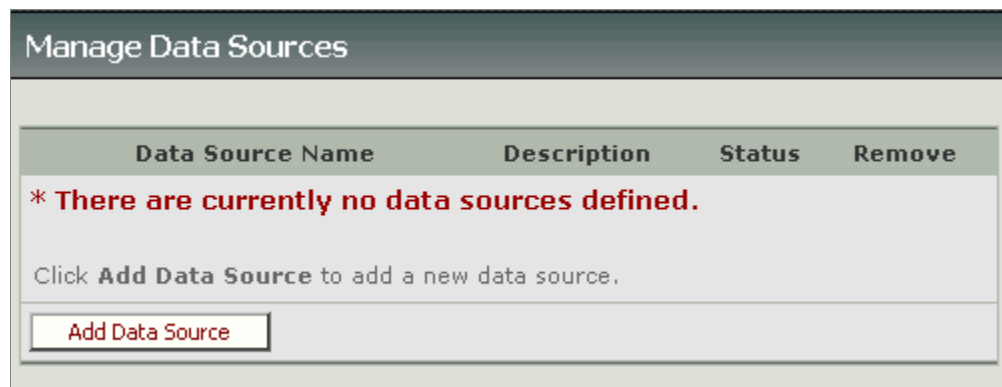


Figure 57. Initial Manage Data Sources Page

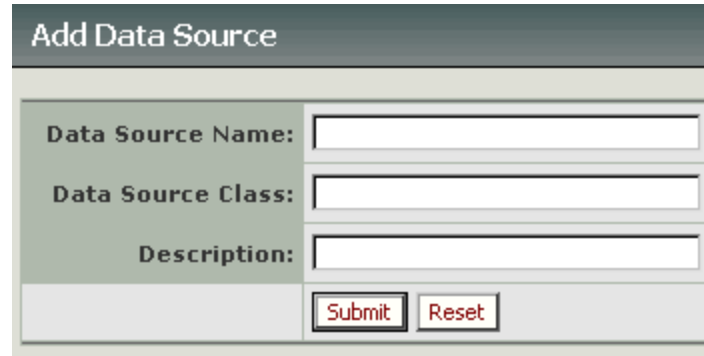
4.2 Adding a Data Source

To add a data source

1. From the Manage Data Sources page (*see Figure 58*), click **Add Data Source**. This brings up the Add Data Source page.
2. From the Add Data Source page (*see Figure 59*), enter the name, class and description for the data source and click **Submit**. This brings up the Edit Data Source page (*see Figure 60*), which displays all of the properties of the data source and their default values. If a property does not have a default value then <null> is displayed.

Please see step #4 of SE FAQ #350

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=350 for further details of what happens when you click the submit button shown in Figure 59.



The screenshot shows a web form titled "Add Data Source". The form has a dark header bar with the title. Below the header, there are three input fields: "Data Source Name:", "Data Source Class:", and "Description:". Each field is followed by a text input box. At the bottom of the form, there are two buttons: "Submit" and "Reset".

Figure 58. Add Data Source Page

Notes

You should refer to the documentation of your JDBC Drivers to determine the class to be used for the data source. It must be a class that implements `javax.sql.DataSource`.

Edit Data Source

*** Data source was added successfully.**

Data Source Name:	<input type="text" value="Employees"/>
Data Source Class:	<input type="text" value="com.newatlanta.jturbo.driver.DataSource"/>
Description:	<input type="text" value="A data source for the Employees database"/>

Data Source Properties

appName	<input type="text" value="<null>"/>
clientHost	<input type="text" value="<null>"/>
databaseName	<input type="text" value="<null>"/>
description	<input type="text" value="A data source for the Employees database"/>
encoding	<input type="text" value="<null>"/>
fetchSize	<input type="text" value="100"/>
language	<input type="text" value="<null>"/>
loginTimeout	<input type="text" value="0"/>
password	<input type="text" value="<null>"/>
portNumber	<input type="text" value="1433"/>
serverName	<input type="text" value="<null>"/>
sp	<input type="text" value="false"/>
sql70	<input type="text" value="true"/>
user	<input type="text" value="<null>"/>
<input type="button" value="Submit"/> <input type="button" value="Reset"/>	

Figure 59. Edit Data Source Page

4.2.1 Accessing a Data Source From a Servlet

When you need to access a data source from a servlet, there are two different ways to do so. You can use either of the following code formats:

To access a data source from a servlet

- Add code to the servlet's .java file using the following format:

```
Context ctx = new InitialContext();
DataSource ds = (DataSource)ctx.lookup( "jdbc/<data source name>" );
```

- Add code to the servlet's .java file using the following format:

```
Context ctx = new InitialContext();
Context envCtx = (Context)ctx.lookup( "java:comp/env" );
DataSource ds = (DataSource)envCtx.lookup( "jdbc/<data source name>" );
```

4.3 Editing a Data Source

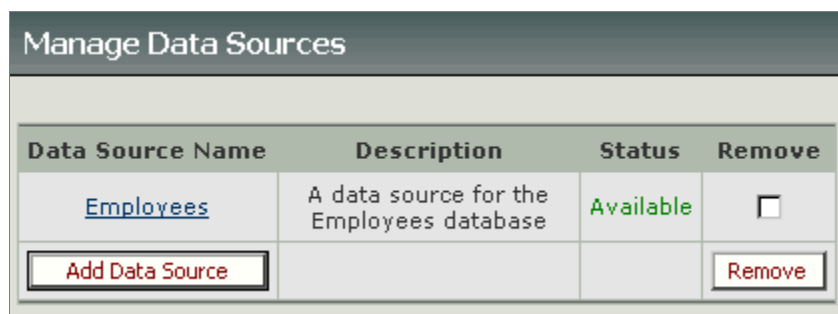
To edit a previously added data source

1. From the Manage Data Sources page (see Figure 61), click the **Data Source Name** to edit. This brings up the Edit Data Source page.
2. From the Edit Data Source page, make your changes and click **Submit** to save or **Reset** to restore the previous settings.

4.4 Removing a Data Source

To remove a data source

- From the Manage Data Sources page (see Figure 61), select the **Remove** checkbox beside the data source you're removing and click **Remove**.

**Figure 60. Removing a Data Source****Note**

Any time the status for a data source appears as Unavailable, check the `ServletExec.log` for the cause.

5. JavaServer Pages (JSP)

ServletExec implements the JavaServer Pages (JSP) 2.1 specification using the built-in `JspServlet`. The `JspServlet` comes preconfigured in ServletExec. This chapter describes how to modify the `JspServlet` configuration options and how to run the examples provided with ServletExec.

More information about JSP, including a copy of the JSP 2.1 specification, can be found on Sun's web site:

<http://java.sun.com/products/jsp>

5.1 JavaServer Pages Overview

The JavaServer Pages feature involves generating HTML pages from hybrid HTML/Java source files. These source files are typically named using the `.jsp` extension (not required, though). Java source code is embedded in `.jsp` pages using JSP tags that are described in the JSP 2.1 specification.

The `JspServlet` performs the following tasks at the first request for a `.jsp` file.

1. Translate the `.jsp` file into a Java source file (`.java`).
2. Compile the Java source file into a Java class file (`.class`).
3. Instantiate and run the compiled Java class file as a servlet.

For subsequent requests for the `.jsp` page, if the requested file was modified after its initial compilation, the `JspServlet` detects the modification and repeats the above tasks. If the requested file was not modified, the `JspServlet` uses the existing servlet to handle the page request.

By default, the `javac` compiler included with the JDK is used to compile the Java source file in the second step. You can configure the `JspServlet` to use an external (non-JDK) compiler such as IBM's Jikes. If the compiler encounters errors while compiling a `.jsp` page, the output error messages are sent to the client's browser.

For example, in order to use IBM's Jikes compiler on a Windows machine with ServletExec/ISAPI, you would configure the `JspServlet` to take the following init parameters:

```
compiler = D:\ibmJikes\jikes
```

```
compileCommand = -classpath 'C:\Program
Files\Java\jdk1.5.0_01\jre\lib\rt.jar;C:\Program Files\New
Atlanta\ServletExec ISAPI\lib\servlet-api.jar;C:\Program Files\New
Atlanta\ServletExec ISAPI\lib\jsp-api.jar;C:\Program Files\New
Atlanta\ServletExec ISAPI\lib\el-api.jar;C:\Program Files\New
Atlanta\ServletExec ISAPI\lib\jasper-el.jar;C:\Program Files\New
Atlanta\ServletExec ISAPI\lib\ServletExec60.jar;C:\Program Files\New
Atlanta\ServletExec ISAPI\Examples'
```

Note

You must also include all necessary classes in the classpath option that gets passed to the alternate Java Compiler through the `compileCommand` init argument of `JspServlet`.

5.2 Running the JSP Examples

If you have problems running this test, check the `ServletExec.log` file for error messages.

To run the `hangman.jsp` example provided with `ServletExec`

1. Place the file `hangman.jsp` in the web server's document root directory (`hangman.jsp` can be found in the `Examples` subdirectory of the `ServletExec` installation directory).
2. Add the bean `newatlanta.bean.HangmanBean` to the `ServletExec` classpath. For example, the full path to the `Hangmanbean.class` file for the default Microsoft IIS installation is:

```
C:\Program Files\New Atlanta\ServletExec ISAPI\Examples\newatlanta\bean\Hangmanbean.class
```

Therefore, the following path should be added to the `ServletExec` classpath:

```
C:\Program Files\New Atlanta\ServletExec ISAPI\Examples\
```

See Chapter 2 for a discussion of the `ServletExec` classpath.

Important

Be sure to stop and restart your web server after modifying the `ServletExec` classpath.

3. Invoke the `hangman.jsp` page from a browser using the following URL:

<http://localhost/hangman.jsp>

5.3 Configuring JavaServer Pages

`com.newatlanta.servletexec.JspServlet` (hereafter referred to as the `JspServlet`) is `ServletExec`'s JSP Engine. It is *implicitly* preconfigured in each webapp that is deployed onto `ServletExec`. And the `.jsp` extension is *implicitly* mapped to that `JspServlet` for each webapp. No manual configuration steps are needed before using JSP pages. In this way JSP pages work with SE "out-of-the-box".

Note

The `JspServlet` is preconfigured in `ServletExec` and does not need to be modified unless you change the value of at least 1 `init` parameter. `Init` params are discussed below.

See Section 3.5.7.2 *Web Application Servlets* for instructions on how to configure a *Servlet*..

You may assign optional initialization (`init`) parameters to the `JspServlet` by configuring it inside your webapp. Only then would you see that `Servlet` listed in your webapp's `web.xml` file. Default values are used, unless you explicitly modify/configure a value. An example is shown in SE FAQ #315:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=315

The `JspServlet` supports the following `init` arguments:

Init Parameter	Description
<code>checkDependencies</code>	If set to <code>true</code> then a JSP page or tag file will be re-compiled whenever any of its dependencies is modified. This should be set to <code>true</code> for development and <code>false</code> for production. The default value is <code>true</code> .
<code>compileAfterRestart</code>	If <code>true</code> , then after a web server restart all <code>.jsp</code> pages are recompiled even if they haven't been modified. This is necessary so that <code>.class</code> files used by a <code>.jsp</code> page that may have been modified will be recompiled (even if the <code>.jsp</code> page hasn't changed). Normally this situation occurs during development, so this parameter should be set to <code>true</code> during development and reset to <code>false</code> for production. The default value is <code>false</code> .
<code>compileCommand</code>	Specifies the compiler options to use when compiling the Java files (for example: <code>compilecommand=-classpath C:\myclasses</code>). The default value for this argument is <code>null</code> .
<code>compiler</code>	Specifies the path to the executable for an external (non-JDK) Java compiler to be used to compile the JSP pages (on Windows the path to the executable must not include the <code>.exe</code> extension). <code>Servletexec</code> has been tested with IBM's <code>jikes</code> compiler. If using this compiler, the <code>compileCommand</code> argument must be used to specify the classpath (see above). The default value for this argument is

Init Parameter	Description
	null, which causes the JDK javac compiler to be used.
convertRequestParams	Specifies how request parameters are handled by the setProperty tag. If true, then such request parameters are converted to the charset of the JSP page before being set in the Java Bean. The default value is false.
defaultPageClassName	Indicates the default base class for JSP pages. You can override the default base class by using the extends JSP directive. Default value is JspHttpJspPage. The specified class must meet all of the requirements for a JSP base class as specified in JavaServer Pages Specification 2.1.
errorPage	Specifies a URL to be invoked when a JSP error occurs (for example: /error.html or /servlet/ErrorServlet).
generateSMAP	If set to true then an SMAP will be generated whenever a JSP page is compiled. This should be set to true when JSP pages are being debugged in an IDE. The default value is false
import	A comma-separated list of package names to be imported by all JSP pages.
optimizeEmptyTags	If true then body manipulation methods are not called for empty custom tags. The default value is true.
packageLevel	Specifies the number of subdirectories from the root document directory path to be prepended to the class name as packages. This argument only needs to be used when virtual servers are used, but not configured using the ServletExec Admin UI. In this case, the value of 1 for this argument should be adequate to insure that the JSP pages generated for each virtual server have a unique class name. The default value is 0.
packagePrefix	The package to prepend to the class name. The default value for this argument is pagecompile.
pageCheckSeconds	The time in seconds the JspServlet should wait after the last check before checking to see if a JSP page has been modified and needs to be recompiled. Setting this parameter will

Init Parameter	Description
	improve performance because it will reduce the number of times the <code>JspServlet</code> accesses the file system. If set to 0, the <code>JspServlet</code> checks a JSP page for modifications every time the page is accessed. The default value is 0.
<code>urlRewriting</code>	If <code>true</code> , then all URLs in a JSP page are rewritten with the Session ID if URL rewriting is enabled for Session Tracking. The default value is <code>false</code> .
<code>verbose</code>	If <code>true</code> , prints a message to <code>System.out</code> when compiling a file. The default is <code>false</code> .
<code>workingDir</code>	The directory for storing the generated <code>.java</code> and <code>.class</code> files. The default value for this argument is the web application's configuration directory under the <code>ServletExec Data</code> directory.

Table 4. `JspServlet` Supported Init Arguments

5.3.1 Assigning a Servlet Alias

To use the `JspServlet`, the `JspServlet` name must be mapped to the servlet suffix alias `*.jsp`.

See Chapter 3 for instructions on configuring servlet names and aliases.

Note

The `JspServlet` name and `*.jsp` alias are *implicitly* preconfigured in `ServletExec` and do not need to be modified unless you want to map JSP pages using a different suffix alias.

5.4 Using JavaServer Pages

The `JspServlet` reads a `.jsp` file that contains embedded JSP directives, JSP declarations, JSP scriptlets, JSP expressions, JSP beans, & JSP tags, and then parses the file and creates a servlet that generates the HTML response page.

See the JSP 2.1 specification for a complete description of the JSP syntax:

<http://java.sun.com/products/jsp/>

Tip

Using the Add a Tag Library feature in `ServletExec` (*see page 41*) greatly simplifies JSP management by eliminating the need to make changes to JSPs after a TLD is moved.

5.4.1 Invoking a JSP Page from a Servlet

A servlet can invoke a JSP page using the `RequestDispatcher` interface. Here's an example:

```
RequestDispatcher dispatcher =
getServletContext().getRequestDispatcher( "/mypage.jsp" );

dispatcher.include( request, response );
// request and response are the parameters to the servlet's service(),
// doGet(), or doPost() method
```

5.4.2 JSP Compiler

JSP pages and tag files can be compiled from the command line using the JSP Compiler tool that can be downloaded from the New Atlanta web site. This tool allows a developer to easily verify if their JSP pages are syntactically correct. It can also be used to package the compiled JSPs in a web application instead of the JSP pages themselves. The standalone JSP Compiler can be downloaded from:

http://www.newatlanta.com/support/servletexec/other_resources.jsp

5.5 Microsoft IIS Extension Mapping

If using SE ISAPI (not SE AS) you can use the Extension Mapping feature of Microsoft IIS 7.0/6.0/5.1/5.0 to control access to JSP pages with NT File System (NTFS) security.

To enable Extension Mapping in IIS 7.0/6.0/5.1/5.0

1. Completely stop IIS as described in section 1.2.1.1 of this document
2. Edit the `servletexec.xml` file (which can be found in the `ServletExec Data` directory) by setting the value of the `<use-iis-ext-mapping>` element to `true`, and then save the file.
3. Add `.jsp` to the Microsoft IIS extension map as follows:
 - Right-click on **My Computer** and choose **Manage**.
 - Expand **Services & Applications** (lower left).
 - Expand **Internet Information Services**.
 - Right-click on **Websites** and choose **Properties**.
 - Click the **Home Directory** tab, and then click **Configuration....** This will open the Application Configuration dialog.
 - In the Application Configuration dialog, click **Add** to open the Add/Edit Application Extension Mapping dialog.
 - In the Add/Edit Application Extension Mapping dialog, set the **Executable** field to the path to the `ServletExec_ISAPI.dll` file and set the **Extension** field to `.jsp`. Make sure the **Check that file exists** checkbox is selected. See Figure 62 for an example.
4. Click **OK** to close all dialogs.

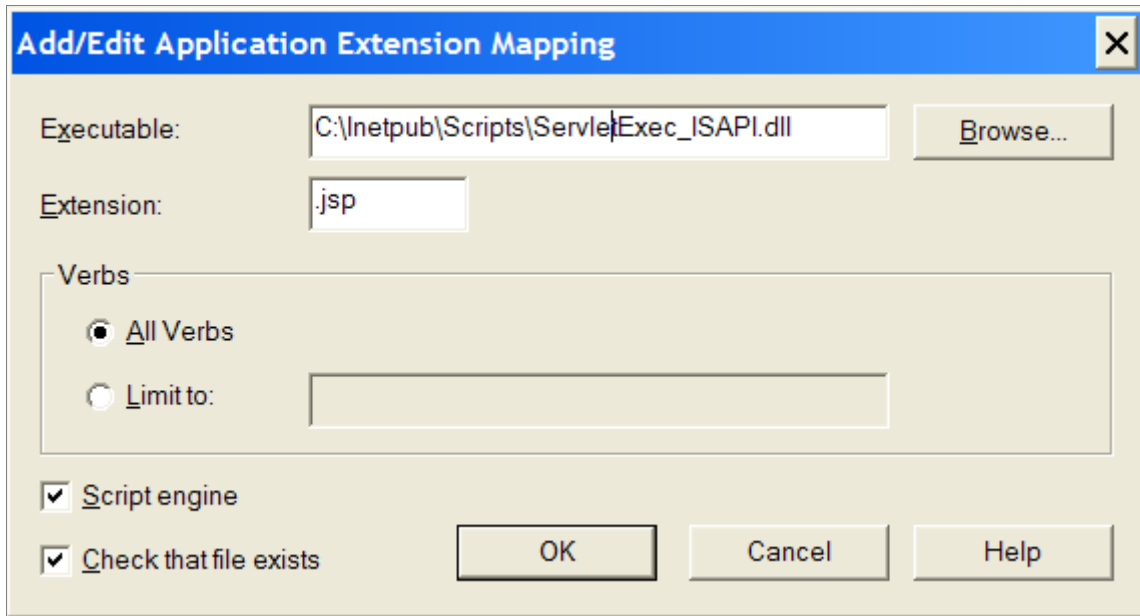


Figure 61. Add/Edit Application Extension Mapping (IIS 5.1)

Important

After setting the value of the `<use-iis-ext-mapping>` element to true in Step 2, you **must** configure **all** ServletExec suffix aliases using IIS Extension Mapping. If the suffix alias does not map to a physical file (unlike `.jsp`) then do not check the **Check that file exists** checkbox in the Add/Edit Application Extension.

Important

IIS Extension Mapping can't be used with SE AS since with SE AS, the SE adapter must do the filtering (not IIS). Otherwise the adapter won't know the group number (SE AS instance) to which the request is to be routed. Using this feature is generally not advisable since the Servlet Specification provides more robust and portable security measures (*see section 3.5.7.4 of this document for details*).

6. JSP Standard Tag Library (JSTL)

Using Standard Tags within JSP Pages

ServletExec 6.0 provides built-in support for the JSP Standard Tag Library (JSTL) 1.2 without requiring any additional installation or configuration. The JSTL defines a set of JSP custom tags that perform many common functions required by JSP authors. More information about the JSTL, including a copy of the JSTL 1.2 specification, can be found on Sun's web site:

<http://java.sun.com/products/jsp/jstl/>

JSTL support in ServletExec 6.0 is based on the JSTL implementation found in Glassfish v2-b50g. The 2 JARs which ship with SE 6.0, and which provide the JSTL implementation are `jstl.jar` & `appserv-jstl.jar`

6.1 JSTL Overview

The JSTL encapsulates, as simple tags, core functionality common to many JSP applications. JSTL has support for common, structural tasks such as iteration and conditionals, tags for manipulating XML documents, internationalization and locale-sensitive formatting tags, and SQL tags. It also provides an API for developers to simplify the configuration of JSTL tags and the development of custom tags that conform to JSTL conventions.

6.2 Running the JSTL Examples

The example web application included with ServletExec and described in Section 3.4 above, includes some simple JSP pages that use some JSTL tags. In addition, the Examples directory of the ServletExec installation directory contains a modified version of the `standard-examples.war` web application provided with the Jakarta JSTL 1.0 implementation; the modified file is named `jstl-standard-examples.war`. The Jakarta example web application has been modified only in that the JAR files that implement the JSTL have been removed; these JAR files are not needed in the web application WAR file because they are integrated into ServletExec 6.0, globally for all webapps. Deploy the `jstl-standard-examples.war` web application on ServletExec as described in Section 3.5 above.

6.3 Configuring the JSTL

No configuration changes are required to add JSTL support to ServletExec 6.0. All of the JAR files required to support the JSTL are included with ServletExec.

6.4 Using the JSTL

ServletExec is preconfigured to recognize the standard (i.e. “well-known”) URIs defined for the JSTL 1.0 and 1.1/1.2 tag libraries. No additional configuration of ServletExec or your web application is required. To use JSTL tags within your JSP pages, specify standard URIs within the JSP taglib directive; for example:

JSTL 1.1 / 1.2

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml" prefix="x" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
```

JSTL 1.0 (has both EL-based and RT-based tag libraries):

EL-based:

```
<%@ taglib uri="http://java.sun.com/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jstl/fmt" prefix="fmt" %>
<%@ taglib uri="http://java.sun.com/jstl/sql" prefix="sql" %>
<%@ taglib uri="http://java.sun.com/jstl/xml" prefix="x" %>
```

RT-based:

```
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jstl/fmt_rt" prefix="fmt" %>
<%@ taglib uri="http://java.sun.com/jstl/sql_rt" prefix="sql" %>
<%@ taglib uri="http://java.sun.com/jstl/xml_rt" prefix="x" %>
```

The Servlet API version specified in your web.xml file can impact how JSTL tags are processed. For more details please see SE FAQ #334:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=334

6.5 JDBC Drivers Used by the JSTL

Because the JSTL libraries are included in the ServletExec’s global classpath and not in the `webapp lib` or `classes` directories, any JDBC drivers required by the JSTL SQL tags must also be added to ServletExec’s global classpath and not in the `webapp lib` or `classes` directories. See Section 2.4 above for instructions on configuring the ServletExec classpath.

7. JavaServer Faces (JSF)

ServletExec 6.0 provides support for JavaServer Faces (JSF) 1.2 (JSR 252) by supporting JSP 2.1 Unified EL. The JSF specification defines an architecture and APIs, which simplify the creation and maintenance of Java Server application GUIs. JSF 1.2 relies on Unified EL support in the Servlet/JSP Engine in order to work properly. More information about JSF, including a copy of the JSF 1.2 specification, can be found on Sun's web site:

<http://java.sun.com/j2ee/javaserverfaces/>

7.1 JSF Overview

The purpose of JavaServer Faces (JSF) is to provide a standard way for building user interfaces to a JavaServer application. There are two parts to a JSF: a set of APIs for UI components and a JSP custom tag library for putting JSF interfaces into a JSP page. JSFs maintain the functional pieces of a user interface element without the presentation logic. That allows for greater reuse of components and allows for easy integration of client side events with server side event handlers.

7.2 Running the JSF Examples

Here is how we ran some JSF examples in SE 6.0. You should be able to do the same: We went to <https://javaserverfaces.dev.java.net/download.html> and got the latest stable release at that time (in binary form) which was version 1.2_04 (dated 3.5.2007). The file's name was “jsf-1.2_04-b07-FCS.zip”.

When expanded, it contained some TLDs, some samples, and a lib folder that contained `jsf-api.jar` & `jsf-impl.jar`

We modified `jsf-guessNumber.war` & `jsf-webtier-sample.war` by adding `jsf-api.jar` & `jsf-impl.jar` to their WEB-INF/lib folders.

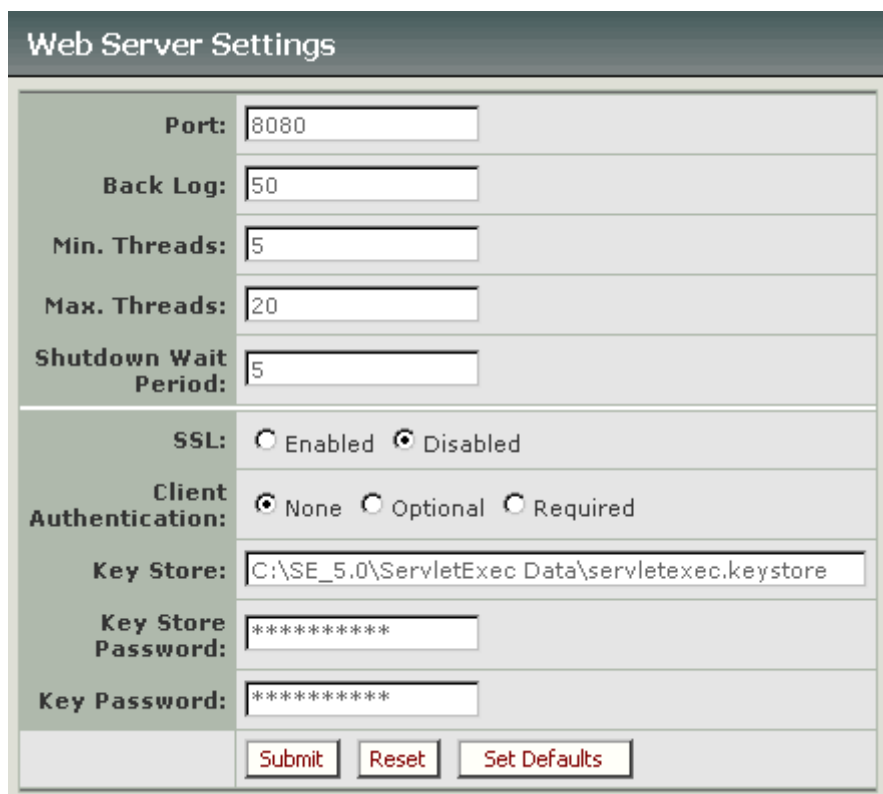
We deployed the apps and they worked fine for us. If you do this and have problems, please send email to support@servletexec.com describing those problems.

8. ServletExec/AS built-in web server

ServletExec/AS comes with a built-in web server. This web server is intended as a development tool and will help you if you have to install ServletExec/AS on a machine that does not already have a web server. You can also use it when you want to install ServletExec/AS on a machine, but do not want to interfere with existing settings on the existing web server. This chapter will tell you how to modify settings and view logs of the ServletExec/AS web server through the ServletExec admin.

8.1 Web Server Settings

There are many settings that can be modified or changed via the ServletExec admin. You can change the settings from the ServletExec admin by clicking on the settings link under the Web Server menu (Figure 63).



The screenshot shows the 'Web Server Settings' configuration page. It features a table of settings with text input fields and radio buttons. At the bottom, there are three buttons: 'Submit', 'Reset', and 'Set Defaults'.

Web Server Settings	
Port:	<input type="text" value="8080"/>
Back Log:	<input type="text" value="50"/>
Min. Threads:	<input type="text" value="5"/>
Max. Threads:	<input type="text" value="20"/>
Shutdown Wait Period:	<input type="text" value="5"/>
SSL:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled
Client Authentication:	<input checked="" type="radio"/> None <input type="radio"/> Optional <input type="radio"/> Required
Key Store:	<input type="text" value="C:\SE_5.0\ServletExec Data\servletexec.keystore"/>
Key Store Password:	<input type="password" value="*****"/>
Key Password:	<input type="password" value="*****"/>
<input type="button" value="Submit"/> <input type="button" value="Reset"/> <input type="button" value="Set Defaults"/>	

Figure 62. Web Server Settings

A description of the items that can be set from the Web Server settings page are shown in the following table:

Web Server Settings	Description
Port	<ul style="list-style-type: none"> ▪ Specifies the port used by the built in web server. The default value is 8080
Back Log	<ul style="list-style-type: none"> ▪ Specifies the number of incoming requests that will be queued before requests start getting refused by the Java Virtual Machine.
Min. Threads	<ul style="list-style-type: none"> ▪ Specifies the minimum number of threads in the thread pool
Max. Threads	<ul style="list-style-type: none"> ▪ Specifies the maximum number of threads in the thread pool.
Shutdown Wait Period	<ul style="list-style-type: none"> ▪ Specifies the number of minutes that ServletExec will wait for current requests to be complete before shutting down the server.
SSL	<ul style="list-style-type: none"> ▪ A radio button that enables or disables the use of SSL on the Web Server.
Client Authentication	<ul style="list-style-type: none"> ▪ Specifies the type of client authentication if SSL is enabled. The default is none, for no access. Optional and required are also valid options.
Key Store	<ul style="list-style-type: none"> ▪ Specifies the location of the key store file that contains the server's SSL certificate.
Key Store Password	<ul style="list-style-type: none"> ▪ Specifies the password for the key store file.
Key Password	<ul style="list-style-type: none"> ▪ Specifies the password for the server's certificate key in the key store file.

If you need to, you can click the set defaults button to reset the settings to their original state. After you change any settings you'll have to restart ServletExec/AS before the settings take affect.

8.2 Access Logs with the Built in Web Server

The ServletExec/AS Web Server collects log information, similar to what you'd find when using most other Web Servers. You can configure the log settings from the Access Log page (Figure 64).

Figure 63. Access Log Settings

Each of the settings is described in the following table:

Access Log Settings	Description
Format	<ul style="list-style-type: none"> Specifies the format that will be used for logging, either no logging, common format, or combined format. In the common format, each access will be logged with the client IP address, remote logical user, authenticated remote user, timestamp, first line of the request, the response status code, and the number of bytes in the body. The combined format includes everything in the common format plus the Referrer request header and the User-Agent request header.
Directory	<ul style="list-style-type: none"> Specifies the directory where logs will be stored.
Resolve Hosts	<ul style="list-style-type: none"> If enabled, the remote IPs are resolved into host names before being entered into the log. The default value is disabled.
Buffer Limit	<ul style="list-style-type: none"> Specifies the number of log entries that will be stored in memory before being written to the log file. The default value is 40.
Rollover Limit	<ul style="list-style-type: none"> Specifies the maximum size of a log. When a log reaches its maximum size it will be backed up, and a new log started. The default value is

Access Log Settings	Description
	100Kb.
Backups Limit	<ul style="list-style-type: none">▪ Specifies the maximum number of backup access logs that can exist. The default is 10.

If you need to, you can click the set defaults button to reset the settings to their original state. After you change any settings you'll have to restart ServletExec/AS before the settings take affect.

9. Web Services

ServletExec 6.0 supports web services developed using the Java API for XML Web Services (JAX-WS) 2.0 (JSR 224). JAX-WS replaces Sun's Java Web Services Developer Pack (JWS DP). In other words, JAX-WS replaces JAX-RPC. Additional details about that can be found here:

<https://jax-ws.dev.java.net/jax-ws-201-m1/docs/UsersGuide.html>

where it states:

Originally, JAX-WS 2.0 was named JAX-RPC 2.0 and the goal was to be backward compatible with JAX-RPC 1.1. Sometime after JAX-RPC 2.0 EAI release, JAX-RPC 2.0 specification was renamed to JAX-WS 2.0 and the goal of being backward compatible with JAX-RPC 1.1 was abandoned. JAX-WS 2.0 defines an all new set of packages and removes JAX-RPC 1.1 specific APIs.

This means that JAX-WS is a completely different technology than JAX-RPC and thus cannot run JAX-RPC applications. If you have an existing JAX-RPC application it must be converted to work with JAX-WS.

JAX-WS 2.0 is distributed inside Glassfish v2 (use b46 or higher).

A fully documented, working example of running a JAX-WS 2.0 based web service inside ServletExec 6.0 has been created and placed onto the New Atlanta website here:

http://www.newatlanta.com/products/servletexec/self_help/other_resources.jsp

under the heading Examples/Tools.

The example is contained in a file named `jaxws-example_60.zip`

10. Multi-hosting / Virtual Server Environments

ServletExec (SE) has several unique features for supporting virtual servers in multi-hosting environments. *Multi-hosting* is a web server software (i.e. IIS, Apache, etc...) feature whereby a single web server hosts multiple virtual servers with different domain/host names. For example, a single web server software installation may host both “www.abc.com” and “www.xyz.com”.

In multi-hosting environments, you have 2 choices regarding your ServletExec configuration. One option is to define your own custom ServletExec Virtual Server(s) via the SE Admin UI and configure each to process 1 specific hostname. The other option is to use multiple instances of SE AS, each running behind the same web server, and with the SE AS native adapter configured to route requests for 1 set of hostnames to 1 SE AS Instance, and requests for some other set of hostnames to a different SE AS instance.

Details about using SE AS in this manner are described by SE FAQ #259: http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=259 and also in section 10.4 of this document.

Details about defining your own custom SE Virtual Server(s) are given in the remainder of this chapter (particularly section 10.6).

Regardless of which option you choose, the main benefit is that you can deploy a different set of web applications for each hostname (or set of hostnames), and still be able to “reuse” the same context paths if you wish.

10.1 Hardware and Software Virtual Servers

Virtual server implementations come in two forms. One form, known as *Hardware Virtual Servers* (or *IP-based Virtual Servers*) assigns a different IP address to each virtual server. In the other form, known as *Software Virtual Servers*, the physical server has only a single IP address. The main effect on ServletExec of the form of virtual server implementation in use is the manner in which the document root directory is defined for each virtual server.

You will set most of ServletExec’s multi-hosting configuration options using the ServletExec Admin UI.

Sun and Apache web servers may require additional directives in the `obj.conf` or `httpd.conf` configuration files, respectively.

See the following sections for more information on multi-hosting configurations.

See the [ServletExec 6.0 Installation Guide](#) for additional discussion of these configuration files.

10.2 Microsoft IIS

Virtual Servers with IIS are often referred to as “Websites”. IIS only supports virtual servers on Windows 2000 Server, Windows 2003 Enterprise Edition, & Windows 2008/Vista. IIS does not support virtual servers on Windows 2000 Professional, Windows XP Professional, or Windows 2003 Standard Edition. Both hardware and software virtual servers are supported by IIS 7.0/6.0/5.1/5.0.

Notes

- The document root directory is specified for each virtual server during IIS configuration.
 - For each IIS virtual server, you must define a Virtual Directory [VD] that maps to the physical directory that contains the `ServletExec_ISAPI.dll` file. The VD must have “Scripts & Executables” permission enabled, and the name of that VD matters. For more details, please read SE FAQ #54
http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=54
-

10.3 Sun ONE / SJSWS

These web servers support both hardware and software virtual servers.

- Hardware virtual servers require a separate `NameTrans` directive to be added to the `obj.conf` file for each virtual server.
See the [ServletExec 6.0 Installation Guide](#) for discussion of the `obj.conf` file.
- In software virtual servers, the web server document root is the document root directory for all virtual servers. No additional configuration is required for `ServletExec`.

In addition to supporting multiple virtual servers, these web servers allow you to install multiple *instances* of the web server. Each server instance is associated with a unique IP address and/or port.

Notes

- You must install `ServletExec` separately for each web server instance.
 - Web Server instances (and `ServletExec`) are configured independently, as if each were a standalone server (because they are, one is a web server and the other is an application server).
-

10.4 ServletExec/AS and Virtual Servers

For ServletExec/AS, it's possible to configure the web server adapter(s) to forward requests to a specific ServletExec/AS instance based on the virtual server to which the request is addressed. For example, you can install two ServletExec/AS instances and configure the web server adapter(s) to forward requests for `www.abc.com` to one instance and `www.xyz.com` to the other.

Configuration data for the ServletExec/AS web server adapter is stored in the `webadapter.properties` file in the ServletExec config directory. Mapping of requests from virtual hosts to ServletExec/AS instances is controlled using the `servletoexec.<instance-name>.hosts` directives. For example, the following directives:

```
servletoexec.frodo.hosts=www.abc.com, www.123.com
...
...
...

servletoexec.bilbo.hosts=www.xyz.com
```

will cause requests for `www.abc.com` or `www.123.com` to be forwarded to the ServletExec/AS instance named "frodo", and requests for `www.xyz.com` to be forwarded to the ServletExec/AS instance named "bilbo". See the comments in the `webadapter.properties` file for more information, as well as SE FAQ #259: http://www.newatlanta.com/c/support/servletoexec/self_help/faq/detail?faqId=259

10.5 The Virtual Server named “default”

ServletExec automatically creates a Virtual Server [VS] named “default” the first time it initializes. ServletExec examines the hostname used in all incoming requests and attempts to match the requested hostname to the SE VS configured to handle that hostname. If no matching SE VS is found, then SE will route the request to the SE VS named “default”. Most people simply use the SE VS named “default” and have no need to create additional SE VS's.

It isn't necessary to configure all (or any) of your own custom SE virtual servers. ServletExec will use the configuration options for the SE VS named “default” when receiving requests for unconfigured servers. There are several possible configuration options, but perhaps the most prominent one is the fact that each SE VS defines its own set of web applications. These webapps may have the same name and context paths of webapps that are deployed on other SE VS's. For single-host environments, do not configure an SE virtual server. Instead, use (or modify) the configuration options for the default server. In other words, just use the SE Admin UI that's deployed in the SE VS named “default” to deploy your own custom webapps.

10.6 Adding/Configuring a Virtual Server

Figure 65 illustrates how to configure a virtual server. You must specify the host name (or IP address) you intend to use in your request URLs (i.e., “www.abc.com”) in the `Server Name` field. It must be a host name that actually resolves to your webserver software (i.e. you can’t simply make one up).

Figure 64. Add Virtual Server Page

The context sharing radio button indicates whether web applications can access contexts in other web applications by using the `ServletContext.getContext()` method. You can either enable or disable this feature. Section 1.6.3 shows how to configure the SE VS named “default”.

The `Allowed IPs` field is discussed below.

10.7 Administering Virtual Servers

Each SE VS will have its own copy of the SE Admin UI webapp deployed on it. But they will all use the same context path (`/servletexec`). So in order to access the right one, you must use the correct hostname in your browser when requesting `/servletexec/admin`. The hostname you use in the request URL will decide which SE Admin UI webapp you’ll be accessing and hence which set of deployed webapps (and other settings) you can modify.

Each SE VS will have its own copy of the default-app deployed on it. And each default-app will list the same Legacy Servlets folder as an External Library so that servlets that reside there (such as `TestServlet`, or `DateServlet` for example) will be visible to each of those default-apps. The hostname you use in the request URL will decide which default-app you’ll be accessing.

In addition, the `Allowed IPs` field allows you to specify a comma-separated list of IP addresses from which clients may bring up the ServletExec Admin UI for a particular virtual server. So you must be certain that you are using a browser that is running on a machine whose IP address will be granted access to the SE Admin UI. The specified IP addresses may include the “*” character to represent all IP addresses in a subrange. For example, the following list of IP addresses would allow access to the ServletExec Admin

UI from a client with the IP address 168.121.97.133, or any IP address in the range from 204.84.126.1 to 204.84.126.255:

```
168.121.97.133,204.84.126.*
```

IPv4 addresses must be of the form x.x.x.x (where x is in the range 0-255)
IPv6 addresses must be of the form [y:y:y:y:y:y:y] (where y is a 4-digit hexadecimal number).

ServletExec always allows access to the ServletExec Admin UI from the local machine on which ServletExec is running regardless of the setting of the `Allowed IPs` field.

To bring up the virtual server's admin pages, use the following URL:

```
http://<host-name>/servletexec/admin
```

where `<host-name>` is the host name of the SE virtual server (and also of your web server software).

To access the configuration options for the default virtual server, use a host name (or the IP address) in your request URL that has **not** been configured as an SE virtual server. This would be the `<host-name>` portion of the admin URL shown above. If multiple hardware virtual servers are defined, use the IP address of any virtual server.

10.8 Compatibility with Older Browsers

To support multiple virtual servers, ServletExec relies on receiving the server host name from the browser in the Host field of the HTTP request headers. Some older browsers do not set the Host field. When receiving requests from these browsers, ServletExec uses the configuration for the default server. If there is no default server configured, the browser will receive a "404 Not Found" result. If there is a default server configured, servlets that access files may have problems because they may not receive the proper document root directory for the virtual server.

The following browsers are known to support the HTTP request header Host field:

- Netscape Navigator® 2.0 and higher (Windows, UNIX)
- Microsoft Internet Explorer 3.0 and higher (Windows)

11. Resource Monitoring

Monitoring your application server resources within ServletExec is very easy. ServletExec's resource monitoring features allow you to view, in real-time, how ServletExec is handling its load. Monitoring these resources will give you the necessary information, should you need to adjust your system.

Specifically, you can monitor...

- Requests
- Sessions
- Threads

In addition to just viewing the displayed information, you can also modify how that information gets displayed, and perform other operations.

See the following sections for detailed information.

11.1 Monitoring Requests

The Monitor Requests page (*see Figure 66*) provides an at-a-glance display of all current and maximum requests. In addition to viewing the current and maximum requests, you can also:

- Clear the number of maximum requests for a web application (just select the **Reset Maximum Requests** checkbox and click **Reset**)
- Modify the page's refresh interval (just change the existing value and click **Submit**)
- Clear the number of maximum requests for all web applications (simply click **Reset All**)

Monitor Requests			
Web Application Name	Current Requests	Maximum Requests	Reset Maximum Requests
basicSessionManagementExample	0	2	<input type="checkbox"/>
TripleExpresoWithCream	0	1	<input type="checkbox"/>
servletexec	1	2	<input type="checkbox"/>
default-app	0	2	<input type="checkbox"/>
Total	1	7	
		<input type="button" value="Reset All"/>	<input type="button" value="Reset"/>
Refresh Interval			
<input type="text" value="10"/> Seconds			
<input type="button" value="Update Refresh Interval"/>			

Figure 65. Monitor Requests Page

11.2 Monitoring Sessions

The Monitor Sessions page (*see Figure 67*) shows all currently active sessions, the maximum active sessions, and the currently active sessions that have been swapped to disk. In addition, you can also modify the page settings as follows.

- Clear the number of maximum sessions for an individual web application (just select the **Reset Max Active** checkbox and click **Reset**)
- Modify the page's refresh interval (just change the existing value and click **Submit**)
- Clear the number of maximum sessions for all web applications (click **Reset All**)

Monitor Sessions				
Web Application Name	Current Active	Current Swapped	Max Active	Reset Max Active
basicSessionManagementExample	0	0	2	<input type="checkbox"/>
servletexec	0	0	2	<input type="checkbox"/>
default-app	0	0	2	<input type="checkbox"/>
Total	0	0	6	
			<input type="button" value="Reset All"/>	<input type="button" value="Reset"/>

Refresh Interval	
<input type="text" value="10"/>	Seconds
<input type="button" value="Update Refresh Interval"/>	

Figure 66. Monitor Sessions Page

If you want to find out more information on the sessions inside an application, click the application name to bring up the information in *Figure 68*. This page displays Session ID, the creation time, the time it was last accessed, the amount of seconds the thread has been inactive, the maximum number of seconds the thread will be allowed to be inactive, the number of attributes, and whether or not the session was swapped.

Session details for web-app: servletexec						
Session Id	Creation Time	Last Accessed Time	Inactive Time	Max Inactive Time	Num Attributes	Swapped
DzXMbuIZGPdjQWi-rZPqPw4TqEw	Thu Aug 28 13:52:35 EDT 2003	Thu Aug 28 13:52:57 EDT 2003	0	1800	2	false
<input type="button" value="Refresh"/>						

Figure 67. Session Details

Clicking on the SessionID will display the list of attributes associated with the session. The name of the attribute, the type of the attribute, the value of the attribute, and whether or not the value is serializable are displayed. This is shown in *Figure 69*.

Attribute details for session: DzXMbuIZGPdjQWi-rZPqPw4TqEw

Name	Type	Value	Serializable
com.newatlanta.servletexec.authUser	java.lang.StringBuffer	"admin"	true
com.newatlanta.servletexec.authConstraint	com.newatlanta.servletexec.AuthConstraint	com.newatlanta.servletexec.AuthConstraint@29ab3e	false

Figure 68. Session Attribute Details

11.3 Monitoring Threads

The Monitor Threads page (*see Figure 70*) provides a display of all running threads, along with the group, start time, and priority information for each thread.

Note

Start Time information is only available for threads that are processing a request.

In addition to viewing this thread information, you can also:

- Modify the page's refresh interval (just change the existing value and click **Submit**)
- Kill a specific request thread (select the **Kill** checkbox and click **Submit**)

Warning

Use this with caution. Killing request threads could create an unknown state for resources.


Monitor Threads				
Thread Name	Thread Group	Start Time	Priority	Kill
Reference Handler	system	Not Available	10	
Finalizer	system	Not Available	8	
Signal Dispatcher	system	Not Available	10	
CompileThread0	system	Not Available	10	
ServletExec: ServletExecServer	main	Not Available	5	
ServletExec: Session Invalidation Thread: host=default	main	Not Available	1	
ServletExec: Session Invalidation Thread: host=default, app=servletexec	main	Not Available	1	
ServletExec: Access Logger	main	Not Available	5	
ServletExec: request: time=1054756790718, uri=/servletexec/admin	main	Wed Jun 04 15:59:50 EDT 2003	5	<input type="checkbox"/>
ServletExec: Responder	main	Not Available	5	
ServletExec: Responder	main	Not Available	5	
ServletExec: Responder	main	Not Available	5	
ServletExec: Responder	main	Not Available	5	
DestroyJavaVM	main	Not Available	5	
ServletExec: Session Invalidation Thread: host=default, app=default-app	main	Not Available	1	
ServletExec: Session Invalidation Thread: host=default, app=basicSessionManagementExample	main	Not Available	1	
				<input type="button" value="Kill"/>

Refresh Interval
<input type="text" value="30"/> Seconds
<input type="button" value="Update Refresh Interval"/>

Figure 69. Monitor Threads Page

11.4 Monitoring Memory

You can use the VM Settings page of the main SE Admin UI (see Figure 71) to view the current state of the JVM's heap size.

 Help

Java Virtual Machine (VM) Settings

Current Java VM:	Java HotSpot(TM) Server VM 1.6.0_01 from Sun Microsystems Inc.
Current VM Info:	mixed mode
Operating System:	Windows XP 5.1 on x86
Selected Java VM:	<input type="radio"/> Sun HotSpot Client <input checked="" type="radio"/> Sun HotSpot Server
Current Heap Size:	4032 KB (4 MB)
Unused/Free Heap:	2890 KB (2 MB)
Maximum Possible Heap:	64832 KB (64 MB)
Heap Summary:	* The heap is currently 06.22% of its maximum possible capacity. * 71.69% of the current heap size is available for use. * The current heap will grow 1507.94% if needed, before reaching its maximum possible size.
Minimum Heap Size:	<input type="text" value="4096"/> K Bytes
Maximum Heap Size:	<input type="text" value="65536"/> K Bytes
Verbose GC:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled
Verbose:	<input type="radio"/> Enabled <input checked="" type="radio"/> Disabled
Class GC:	<input checked="" type="radio"/> Enabled <input type="radio"/> Disabled
System Output:	<input type="text" value="com.newatlanta.servletexec.SESystemOutputStream"/>
System Error:	<input type="text" value="com.newatlanta.servletexec.SESystemOutputStream"/>
<input type="button" value="Submit"/> <input type="button" value="Reset"/> <input type="button" value="Set Defaults"/>	

Figure 71. VM Settings Page

12. Developing Servlets

The best way to develop servlets that you plan to deploy on ServletExec is to use the built-in web server in ServletExec/AS. The built-in web server allows you to develop and debug your servlets in an environment that very closely simulates the ServletExec deployment environment.

12.1 ServletExec Tech Support FAQ

In addition to answering common questions about ServletExec installation and configuration problems, we are also regularly updating the online ServletExec Tech Support FAQ with debugging tips, bug reports, workarounds, and issues involving interactions with third-party products.

http://www.newatlanta.com/products/servletexec/self_help/index.jsp

12.2 Java Servlet API Documentation

Before beginning to develop servlets, you should review the Java Servlet API 2.5 specification and the Javadoc documentation, both of which can be downloaded from Sun's web site.

<http://java.sun.com/products/servlet/>

You should also review the JavaServer Pages (JSP) 2.1 specification, which can be downloaded from Sun's web site.

<http://java.sun.com/products/jsp/>

You may also want to review the Java Server Pages Standard Tag Library (JSTL) spec, which is located here:

<http://java.sun.com/products/jsp/jstl/>

The Java Server Faces (JFS) spec is located here:

<http://java.sun.com/j2ee/javaserverfaces/>

Other valuable online resources are listed on New Atlanta's web site:

http://www.newatlanta.com/products/servletexec/self_help/other_resources.jsp

12.3 Compiling Servlets

Before you can compile servlets, you will need to add the Java Servlet API classes to your compiler classpath. The Java Servlet 2.5 API classes are in the `servlet-api.jar` archive included with ServletExec, which can be found within the `lib` sub-directory of the ServletExec installation directory.

12.4 Debugging Servlets

The built-in web server in ServletExec/AS should be your first choice for debugging servlets. For more details about step-debugging your servlets with your favorite IDE, please see:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=2

Other debugging techniques include:

Instrument your code (add debug calls such as `System.out.println()` to your servlet code). The output from `System.out` and `System.err` is written to the `ServletExec.log` file. In addition, all exceptions and errors caught by ServletExec are written to this log file.

Output to `System.out` and `System.err` is not buffered, but is written immediately to the `ServletExec.log` file. Therefore, you should remove extraneous calls to `System.out` and `System.err` from your code before going into production, otherwise your servlet won't perform as well as it could. Or you could use the `servlet.log()` method for routine output in your servlets. Output to the `log()` method is buffered by ServletExec and written to the `Servlet.log` file by a background thread, thereby minimizing impact on performance of your servlet. Each webapp context has its own `servlet.log` file whose contents can be viewed on the Admin UI for each specific webapp.

When using SE ISAPI, an application named DBMON (`dbmon.exe`) is placed into the SE ISAPI installation folders. DBMON is diagnostic tool that is described further here:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=187

In short, DBMON creates a console window to which `System.out` and `System.err` are redirected, and to which exceptions and errors caught by ServletExec are logged. Simply launch DBMON any time ServletExec ISAPI is running. You can find DBMON within the ServletExec installation directory.

Note

DBMON does not display output when run remotely (via windows terminal services). See SE FAQ #187 (link above) for an alternative tool to use in that scenario.

12.5 Profiling Servlets

There are several commercial Java profiling tools that can be used to profile servlets running within ServletExec. They are:

▪ JProbe	http://www.quest.com/jprobe/index.asp
▪ Optimizeit	http://www.borland.com/optimizeit/index.html
▪ NuMega DevPartner	http://www.numega.com/products/suites/devp_java.shtml

Consult the documentation provided with your profiler to learn how to hook it into ServletExec. Information about using OptimizeIt with ServletExec is given in SE FAQ #59:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=59

12.6 Servlet Threads

Normally, a servlet's `doGet()`, `doPost()`, and `service()` methods may be invoked concurrently on multiple threads. Therefore, these methods must be thread-safe. Here are some general guidelines to follow:

- Parameters to these methods and local variables declared within these methods are thread-safe. No special effort is required on your part.
- Class variables (those not defined within a method) are not thread-safe. Access to class variables must be protected by using Java's `synchronized` keyword.
- DO NOT declare your `doGet()`, `doPost()`, or `service()` methods to be `synchronized`. Doing so will allow only a single thread to execute through your servlet, which will seriously degrade performance. Instead, used `synchronized` code blocks or other `synchronized` methods to control access to your class variables.

If you don't want to worry about threading issues, you can declare your servlet classes to implement the `SingleThreadModel` interface; doing so guarantees that no two threads will execute concurrently in the servlet. Because of potential performance degradation, use of the `SingleThreadModel` interface is not recommended.

For more detailed information regarding Java threads and use of the `synchronized` keyword, we highly recommend the following two references:

Concurrent Programming in Java	by Doug Lea	ISBN 0-201-69581-2
Java Threads	by Scott Oaks & Henry Wong	ISBN 1-56592-216-6

12.7 User Accounts for Microsoft IIS

Because ServletExec/ISAPI runs as part of the Microsoft IIS process, your servlets will run under different user accounts at different times. This will primarily affect their ability to read and write to the file system because access to the NT File System (NTFS) is based on the user account of the process. Here are the rules to determine which account your servlet is running under:

Rule	Description
1	During normal request processing in your servlet's <code>service()</code> , <code>doGet()</code> , or <code>doPost()</code> method, your servlet will be running under the user account of the authenticated user, if the user had to enter a username and password to access your servlet. Otherwise, your servlet will be running under the <code>IUSR_<server-name></code> account (the account under which IIS runs anonymous user requests).
2	If your servlet is configured to be loaded by ServletExec during initialization, your <code>init()</code> method will be executed under the <code>SYSTEM</code> account with IIS 5.1 and earlier and under the application pool identity specified for ServletExec with IIS 6.0. Otherwise, if your servlet is loaded when it receives its first request, Rule 1 applies.
3	Normally, your servlet's <code>destroy()</code> method will run under the <code>SYSTEM</code> account with IIS 5.1 and earlier and under the application pool identity specified for ServletExec with IIS 6.0 when it is invoked due to a shutdown of ServletExec. However, if your servlet is reloaded due to the web application being reloaded, Rule 1 applies.

Table 5. Microsoft IIS User Account Rules

If using SE ISAPI, you may find SE FAQ #60 to be very helpful:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=60

12.8 UNIX File Descriptors

If you're running ServletExec on a UNIX system that receives a very high traffic volume, you may begin to receive "too many files open" errors. This means the file descriptor limit of your system is being exceeded.

You can examine the current setting of the file descriptor limit using the `limit` command. Use the following `csh` shell command to increase the file descriptor limit (256 in this example):

```
limit descriptors 256
```

Use the following `sh` or `ksh` shell command to increase the file descriptor limit (256 in this example):

```
ulimit -n 256
```

Other shells will have similar commands to allow you to increase the file descriptor limit.

SE FAQ #4 discusses this:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=4

12.9 SSL Request Attributes

The Java Servlet API specification defines request attributes that must be exposed by the servlet container if a request is transmitted over a secure protocol, such as HTTPS. These attributes are listed in Table 6, below.

ServletExec supports the retrieval of SSL security information via these request attributes for the following web servers: Microsoft IIS, Apache-SSL, Apache with `mod_ssl`, SunONE, and SJSWS. The following sections provide details for each web server.

Attribute	Attribute Name
Cipher suite	<code>javax.servlet.request.cipher_suite</code>
Algorithm bit size	<code>javax.servlet.request.key_size</code>
Client certificate	<code>javax.servlet.request.X509Certificate</code>

Table 6. SSL Request Attributes

12.9.1 Microsoft IIS

For Microsoft IIS, ServletExec supports the retrieval of the key size and client certificate (if present); ServletExec does not support retrieval of the cipher suite for Microsoft IIS. Browsers will only send client certificates if IIS is configured to accept or require them. The client certificate information retrievable from Microsoft IIS is: Subject Distinguished Name (DN), Issuer DN, and Serial Number.

12.9.2 SunONE / SJSWS

For SunONE / SJSWS, ServletExec supports the retrieval of the cipher suite, key size, and client certificate (if present). Browsers will only send client certificates if the server is configured to require them. The client certificate information retrievable from these servers is: Subject Distinguished Name (DN) and Issuer DN.

12.9.3 Covalent Fast Start Server (CFSS)

For CFSS, ServletExec supports the retrieval of the cipher suite, key size, and client certificate (if present). Browsers will only send client certificates if CFSS is configured with `SSLVerifyClient` set to “require” or “optional,” and if the following directive is added to the CFSS `httpd.conf` file:

```
SSLOptions +ExportCertDate +CompatEnvVars +StdEnvVars
```

The client certificate information retrievable from CFSS is: Subject Distinguished Name (DN), Issuer DN, Serial Number, Version, Signature Algorithm Name, the notAfter date, and the notBefore date.

12.9.4 Apache-SSL

For Apache-SSL, ServletExec supports the retrieval of the cipher suite, key size, and client certificate (if present). Browsers will only send client certificates if Apache-SSL is configured with `SSLVerifyClient` set to 1, 2, or 3. Also, before installing ServletExec, the Apache-SSL `apxs` must be modified to add the path to the include directory for the Open SSL headers to `$CFG-CFLAGS`; for example:

```
-I/user/local/include
```

The client certificate information retrievable from Apache-SSL is: Subject Distinguished Name (DN), Issuer DN, Serial Number, Version, the notAfter date, and the notBefore date.

12.9.5 Apache with mod_ssl

For Apache with `mod_ssl`, ServletExec supports the retrieval of the cipher suite, key size, and client certificate (if present). Browsers will only send client certificates if `mod_ssl` is configured with `SSLVerifyClient` set to “require” or “optional,” if the following directive is added to the Apache `httpd.conf` file:

```
SSLOptions +ExportCertDate +CompatEnvVars +StdEnvVars
```

and if the following directive in `httpd.conf` is uncommented-out:

```
SSLCACertificateFile
```

The client certificate information retrievable from CFSS is: Subject Distinguished Name (DN), Issuer DN, Serial Number, Version, Signature Algorithm Name, the notAfter date, and the notBefore date.

13. Customizable Session Tracking

Customizing Session Tracking

ServletExec has always provided a Session Manager. The Session Manager is responsible for creating & maintaining Session Objects. Sessions are tracked across multiple requests using a SessionID (either by rewriting emitted URLs to contain the SessionID, or by storing the SessionID in a Cookie). Applications may store Session data inside the Session object using standard Servlet API method calls. The default Session Manager stores sessions either in memory or on the file system of the server. ServletExec also offers the ability to create your own custom session manager. Each web application can be configured with a different type of Session Manager if you wish. Using your own Session Manager allows you to generate your own SessionIDs and store session data wherever you like, such as in a database. This chapter will show you how to:

- Install a session manager
- Extend the default session manager
- Use the provided `JdbcSessionManager` to store sessions in a database (this facilitates running SE in a load-balanced or failover environment).

13.1 Installing Custom Session Managers

This section will show you how to install a session manager. Once installed, web applications can be set up to use that session manager.

13.1.1 Installing a Generic Session Manager

You can follow these instructions to install your own session manager:

1. Place the custom session manager in ServletExec's Global classpath or in the web application lib or classes folder.
2. On the configure web application admin page for the web application, enable ServletExec extensions for your particular webapp.
3. On the session tracking admin page for that web application, set the Session Manager Class Name to the fully qualified name of the custom session manager.

Information later in this chapter will help to explain how you can create your own session manager.

13.1.2 Deploying the Example Session Manager

If you do not have a session manager, New Atlanta provides an example of a custom session manager at their web site. You can use the `basicSessionManagementExample.war` as a guide to create your own session manager, as will be discussed later in this chapter. You can download it from:

http://www.newatlanta.com/products/servletexec/self_help/other_resources.jsp

1. Use the SE Admin UI to deploy `basicSessionManagementExample.war` (see Figure 14 in section 3.4 of this document). Be sure to enable ServletExec Extensions on that screen.
2. On the session tracking admin page for `basicSessionManagementExample`, set the Session Manager Class Name to `BasicSessionManager`.

New Atlanta's `basicSessionManagementExample` was created as an example to help you create your own session managers and should not be used in a production environment.

Point your browser to `http://<servername>/basicSessionManagementExample` to view the JavaDocs and an example which shows the manager in action.

13.2 Creating Your Own Session Manager

ServletExec allows you to create your own session manager. There are three classes that you must use to create your session manager.

These are the three classes:

- **SessionManager:** Custom session managers are built by extending this class and implementing its abstract methods.
- **Session:** This class is used by `SessionManager` to create session objects. Session objects are always wrapped in a `SessionWrapper` which handles Servlet-specific requirements including throwing `IllegalStateException` for invalidated sessions and calling the appropriate event-listening methods to notify the event-listeners of an attribute change.
- **SessionManagerConfiguration:** Represents the in-memory configuration of the web application's session manager. This class contains a lot of methods for retrieving data about the session manager. Using it is not required, although the `getConfiguration()` method of the `SessionManager` class does need to return this object.

If you haven't already done so, you should download the `basicSessionManagementExample.war` file from the New Atlanta site. (www.newatlanta.com) and study it. It contains source code and documentation which can be of use to you when creating your own Session Manager.

13.3 Using the JDBC Session Manager

ServletExec contains a JDBC session manager, which allows for storage of session data in a database. This is useful in clustered/load-balanced environments where session data needs to be shared by multiple instances of ServletExec. This session manager also provides for session data being stored between restarts of your webapp. This session manager was built using the same techniques that anyone would use to create a custom session manager in ServletExec. A quick “tutorial” is given here:

http://www.newatlanta.com/c/support/servletexec/self_help/faq/detail?faqId=350

13.3.1 How it Works

When you deploy an application with ServletExec Extensions enabled, you can specify that the `JdbcSessionManager` be the session manager for that webapp. This is done on the Session Tracking page of the Admin UI for that particular webapp. The fully-qualified name for that session manager is:

```
com.newatlanta.servletexec.session.JdbcSessionManager
```

To use the JDBC Session Manager you must first configure a JDBC `DataSource` on the ServletExec admin pages (*see section 4.2 Adding a Data Source*). You must specify the name of the `datasource` in `jdbcSessionManager.properties` file. The default name for the `datasource` is `JTurbo_SQLServer_DataSource`. An error will occur upon start up if this `datasource` does not exist, and `JdbcSessionManager` is being used.

When the JDBC Session Manager is set as the session manager of an application, ServletExec will automatically deploy a request listener into that webapp, named `JdbcSessionManagerListener`. Do not delete this request listener, for it is used to handle the opening and closing of the JDBC Connections. This listener is automatically removed if the session manager is changed.

The JDBC session manager will insert a dummy row into the database for the purposes of ensuring that only one session invalidation thread will be invalidating sessions at a time. The `SessionID` of the dummy row is hard coded to be `zzzzzzzzzz-dummy`. The dummy `SessionID` length is different than a standard `SessionID`, and therefore the dummy ID will not be mistaken for a real `SessionID` generated by the Session Manager. The standard locking is per row, not per table, so that requests can come from different users without having to wait for other requests to finish.

13.3.2 Tweaking the JDBC Session Manager’s Properties

If you need to configure the JDBC Session Manager, you can do so using the `jdbcSessionManager.properties` file. This file can be copied out of the `ServletExec60.jar` file and placed in the `WEB-INF/classes/` folder of the webapp using this directory structure: `/com/newatlanta/servletexec/session/`.

The property file includes this information:

- The name of the data source to contain the data. The default name is `JTurbo_SQLServer_DataSource`, and can be changed as desired.

- The amount of time before a stale session is invalidated and the number of sessions that the Session Manager will try to invalidate at once.
- Queries to create the session table, retrieve session data, create session data, and modify session data. Queries were tested with Microsoft SQL Server 2000 and Oracle 9i release 2 (9.2.0.1.0)

The file is fully documented and can easily be modified to suit your needs. If you need to modify the SQL Queries used by the JDBC Session Manager, or you wish to add more queries (to support another database brand or version perhaps), you can do so by editing the `jdbcSessionManager.properties` file. If you intend to modify that file, we recommend that you first extract a copy of it from the `ServletExec60.jar` file and place the copy into in the classes folder of your webapp, preserving the directory structure so that you have:

```
WEB-INF/classes/com/newatlanta/servletexec/session/jdbcSessionManager.properties
```

Once there it can be modified as needed.