Sun Java™ System

Application Server Enterprise Edition 8.1 管理指南

2005Q1

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

文件號碼 819-1554

目錄

前言 **15**

本書的讀者 15
閱讀本書之前 16
本書的結構 16
本書使用的慣例 16
印刷排版慣例 16
符號 17
預設路徑和檔案名稱 18
Shell 提示符號 19
相關文件 19
此文件集中包含的文件 20
其他伺服器文件 21
存取 Sun 線上資源 21
與 Sun 技術支援部門聯絡 22
相關的協力廠商網站參考 22
Sun 歡迎您提出意見和建議 22

第 1 章 入門 **23**
關於 Sun Java System Application Server 23
什麼是 Application Server？ 23
Application Server 架構 24
存取外部系統 25
管理主控台 26
asadmin 公用程式 27
Application Server Management Extension (AMX) 27
Application Server 配置 28
配置 Application Server 28

配置網域 28
建立網域 29
刪除網域 29
列示網域 29
啓動網域 29
若要在 Windows 中啓動預設網域，請執行以下步驟： 30
重新啓動伺服器或網域 30
停止網域 30
若要使用管理主控台停止網域，請執行以下步驟： 30
若要在 Windows 中停止預設網域，請執行以下步驟： 30
重新建立網域管理伺服器 31
DAS 遷移的步驟 31
應用程式伺服器實例 33
關於應用程式伺服器實例 33
定義 Application Server 實例 34
關於獨立實例 36
檢視一般伺服器資訊 36
管理應用程式 36
管理資源 37
Administration Server 進階設定 38
設定應用程式配置 38
設定自動部署 38
設定其他特性 39
設定網域屬性 39
實例特定的配置特性 40
新增或刪除實例特性 40
建立實例 41
啓動實例 41
恢復作業事件 42
停止實例 42
關閉伺服器實例 42
配置變更 43
變更 Application Server 配置 43
Application Server 中的連接埠 44
檢視連接埠號 44
變更管理伺服器連接埠 44
變更 HTTP 連接埠 45
變更 IIOP 連接埠 45
使用管理服務配置 JMX 連接器 46
編輯 JMX 連接器配置 46
變更 J2SE 軟體 47
使用線上說明 47
更多資訊 48

第 2 章 配置叢集 49
關於叢集 49
什麼是叢集？ 49
叢集類型 50
叢集、實例、負載平衡器和階段作業 50
用於叢集的管理主控台作業 51
建立叢集 51
配置叢集 52
遷移 EJB 計時器 53
爲叢集建立伺服器實例 53
配置叢集伺服器實例 54
爲叢集配置應用程式 54
爲叢集配置資源 55
刪除叢集 55
將多個叢集用於線上升級而不使服務受到損失 56

第 3 章 配置負載平衡和防故障備用 57
關於 HTTP 負載平衡和防故障備用 57
HTTP 負載平衡和防故障備用 58
HTTP 負載平衡的需求 58
瞭解指定的請求和未指定的請求 58
HTTP 負載平衡演算法 59
關於粘性循環負載平衡演算法 59
負載平衡和防故障備用範例應用程式 60
HTTP 負載平衡設定概述 60
爲 HTTP 負載平衡配置 Web 伺服器 61
關於 Web 伺服器配置 61
對 Sun Java System Web Server 的修改 61
對 Apache Web Server 的修改 62
安裝程式所作的修改 62
安裝後的修改 63
對 Microsoft Windows 的附加修改 63
對 Microsoft IIS 的修改 64
配置多個 Web 伺服器實例 65
HTTP 負載平衡器配置作業 66
建立 HTTP 負載平衡器配置 66
建立 HTTP 負載平衡器參考 67
啓用用於負載平衡的伺服器實例 68
啓用用於負載平衡的應用程式 68
建立 HTTP 運作狀態檢查程式 68
建立運作狀態檢查程式 68
正常實例的附加運作狀態檢查特性 69
匯出負載平衡器配置檔案 70

變更 HTTP 負載平衡器配置 70
啓用動態重新配置 71
停用 (靜止) 伺服器實例或叢集 71
停用 (靜止) 應用程式 72
配置 HTTP 和 HTTPS 階段作業防故障備用 72
關於 HTTPS 路由 73
配置 HTTPS 路由 73
有關負載平衡 HTTP/HTTPS 請求的已知問題 73
配置等冪 URL 74
配置 HTML 錯誤頁面 74
監視 HTTP 負載平衡器外掛程式 75
配置記錄訊息 75
記錄訊息類型 75
負載平衡器配置器記錄訊息 75
請求派送和執行階段記錄訊息 76
配置器錯誤訊息 76
配置監視 77
瞭解監視訊息 78
升級應用程式 79
關於捲動升級 79
在單一獨立叢集中進行升級 79
在兩個叢集中進行升級 80
關於 RMI-IIOP 負載平衡和防故障備用 82
RMI-IIOP 負載平衡和防故障備用的需求 82
RMI-IIOP 負載平衡和防故障備用演算法 83
RMI-IIOP 範例應用程式 83

第 4 章 配置節點代理程式 85
關於節點代理程式 85
節點代理程式 85
自動建立的節點代理程式 86
節點代理程式和伺服器實例管理 87
附加節點代理程式 87
節點代理程式萬用字元 87
部署節點代理程式 87
部署節點代理程式之前 87
線上部署 88
離線部署 88
節點代理程式和網域管理伺服器同步 90
節點代理程式同步 90
伺服器實例同步 90
同步大型應用程式 90
檢視節點代理程式記錄 91

可以通過管理主控台和 asadmin 工具執行的作業 91
用於節點代理程式的管理主控台作業 92
檢視一般節點代理程式資訊 92
建立節點代理程式萬用字元 93
刪除節點代理程式配置 94
編輯節點代理程式配置 94
編輯節點代理程式範圍 95
爲 JMX 編輯節點代理程式的偵聽程式 95
asadmin 工具中用於節點代理程式的作業 96
建立節點代理程式 96
啓動節點代理程式 97
停止節點代理程式 98
刪除節點代理程式 98

第 5 章 部署應用程式 99
關於部署 99
部署生命週期 99
J2EE 歸檔檔案的類型 101
命名慣例 102
有關部署應用程式的管理主控台作業 102
部署企業應用程式 103
部署 Web 應用程式 104
啓動已部署的 Web 應用程式 106
部署 EJB 模組 107
部署連接器模組 108
建立生命週期模組 110
部署應用程式用戶端模組 111
有關列示、取消部署以及啓用應用程式的管理主控台作業 113
列示已部署的應用程式 113
列示子元件 113
檢視已部署的應用程式的模組描述元 114
取消部署應用程式 114
啓用和停用應用程式 114
管理應用程式目標 115
部署在其他虛擬伺服器上 116
重新部署到多個目標 116
開發環境 116
生產環境 116
啓用和停用動態重新載入 116
適用於開發者的部署方法 117
使用自動部署 117
部署目錄中未封裝的應用程式 119
使用 deploytool 公用程式 119

使用部署規劃 119

第 6 章 JDBC 資源 121
關於 JDBC 資源 121
JDBC 資源 121
JDBC 連線池 122
JDBC 資源和連線池如何協同工作 122
設定資料庫存取 123
設定資料庫存取的一般步驟 123
整合 JDBC 驅動程式 123
關於 JDBC 連線池 124
建立 JDBC 連線池 124
編輯 JDBC 連線池 125
一般設定 126
池設定 126
連線驗證 127
作業事件隔絕 127
特性 128
驗證連線池設定 128
刪除 JDBC 連線池 128
關於 JDBC 資源 129
建立 JDBC 資源 129
編輯 JDBC 資源 130
刪除 JDBC 資源 130
啓用和停用 JDBC 資源 131
關於持續性管理程式資源 131
建立持續性管理程式資源 131
編輯持續性管理程式資源 132
管理資源目標 132
刪除持續性管理程式資源 132
啓用和停用持續性管理程式資源 132

第 7 章 配置可用性和階段作業持續性 133
關於可用性和階段作業持續性 133
需要階段作業持續性的原因 133
階段作業持續性配置概況 134
可用性層級 135
處於 HTTP 階段作業狀態的單次登入之可用性 135
應用程式範例 136
用於配置可用性的管理主控台作業 136
停用可用性時配置 SFSB 階段作業儲存 136
配置伺服器實例層級的可用性 137

配置 Web 容器層級的可用性 137
配置 EJB 容器層級的可用性 139

第 8 章 配置 Java 訊息服務資源 141
關於 JMS 資源 141
Application Server 中的 JMS 提供者 141
JMS 資源 142
JMS 資源與連接器資源之間的關係 143
用於 JMS 連線工廠的管理主控台作業 143
建立 JMS 連線工廠資源 143
編輯 JMS 連線工廠資源 146
刪除 JMS 連線工廠資源 147
用於 JMS 目標資源的管理主控台作業 147
建立 JMS 目標資源 147
編輯 JMS 目標資源 148
刪除 JMS 目標資源 149
用於 JMS 實體目標的管理主控台作業 150
建立 JMS 實體目標 150
刪除 JMS 實體目標 151
用於 JMS 提供者的管理主控台作業 152
配置 JMS 提供者的一般特性 152
建立 JMS 主機 156
編輯 JMS 主機 156
刪除 JMS 主機 157

第 9 章 配置 JavaMail 資源 159
關於 JavaMail 159
JavaMail API 159
有關 JavaMail 的管理主控台作業 160
建立 JavaMail 階段作業 160
編輯 JavaMail 階段作業 161
刪除 JavaMail 階段作業 162

第 10 章 JNDI 資源 163
關於 Java 命名與目錄介面 (JNDI) 163
JNDI 名稱和資源 163
J2EE 命名服務 164
命名參考與連結資訊 164
關於自訂資源 165
使用自訂資源 165
建立自訂資源 166
編輯自訂資源 166

刪除自訂資源 167
列示自訂資源 167
關於外部 JNDI 儲存庫和資源 167
使用外部 JNDI 儲存庫和資源 168
建立外部資源 168
編輯外部資源 169
刪除外部資源 170
列示外部資源 170

第 11 章 連接器資源 171
關於連接器 171
連接器模組、連線池和資源 171
連接器連線池作業 172
設定 EIS 存取的一般步驟 172
建立連接器連線池 172
編輯連接器連線池 173
刪除連接器連線池 175
連接器資源作業 175
建立連接器資源 175
編輯連接器資源 176
刪除連接器資源 176
配置連接器服務 177
受管理物件資源作業 177
建立受管理物件資源 177
編輯受管理物件資源 178
刪除受管理物件資源 179

第 12 章 管理已命名的配置 181
關於已命名的配置 181
已命名的配置 181
default-config 配置 182
建立實例或叢集時建立的配置 182
唯一連接埠號和配置 182
用於已命名的配置的管理主控台作業 183
建立已命名的配置 183
編輯已命名的配置的特性 184
編輯參考配置的實例的連接埠號 185
檢視已命名的配置的目標 185
刪除已命名的配置 185

第 13 章 J2EE 容器 187
關於 J2EE 容器 187

J2EE 容器的類型 187
Web 容器 187
EJB 容器 188
有關 J2EE 容器的管理主控台作業 188
配置一般 Web 容器設定 188
配置 Web 容器階段作業 188
配置管理員內容 189
配置儲存內容 190
配置一般 EJB 設定 191
階段作業儲存位置 191
池設定 191
快取設定 192
配置訊息導引 Bean 設定 193
配置 EJB 計時器服務設定 194
配置計時器服務 194
將外部資料庫與計時器服務一起使用 195

第 14 章 配置安全性 197
關於 Application Server 安全性 197
安全性概況 198
瞭解應用程式和系統安全性 198
管理安全性的工具 198
管理密碼安全性 199
指定安全性職責 202
關於認證與授權 203
認證實體 203
對使用者進行授權 204
指定 JACC 提供者 204
稽核認證與授權決策 204
配置訊息安全性 205
瞭解使用者、群組、角色和範圍 205
使用者 206
群組 206
角色 206
範圍 207
證書和 SSL 簡介 208
關於數位證書 208
關於安全套接字層 209
關於防火牆 210
使用管理主控台管理安全性 211
伺服器安全性設定 211
範圍和 file 範圍使用者 211
JACC 提供者 212

稽核模組 212
訊息安全性 212
HTTP 和 IIOP 偵聽程式安全性 213
管理服務安全性 213
安全性對映 213
用於安全性的管理主控台作業 214
配置安全性設定 214
控制對管理工具的存取 215
用於範圍的管理主控台作業 216
建立範圍 217
建立 ldap 範圍 218
建立 solaris 範圍 220
建立自訂範圍 220
編輯範圍 221
編輯 file 和 admin-realm 範圍 222
使用網路安全性服務 (NSS) 管理使用者 222
管理 file 範圍使用者 223
編輯 certificate 範圍 225
配置相互認證 225
刪除範圍 226
設定預設範圍 227
有關 JACC 提供者的管理主控台作業 227
建立 JACC 提供者 227
編輯 JACC 提供者 228
刪除 JACC 提供者 229
設定使用中的 JACC 提供者 229
有關稽核模組的管理主控台作業 230
建立稽核模組 230
編輯稽核模組 231
刪除稽核模組 232
啓用和停用稽核記錄 232
設定使用中稽核模組 233
使用預設稽核模組 233
有關偵聽程式和 JMX 連接器的管理主控台作業 234
配置 HTTP 偵聽程式的安全性 234
配置 IIOP 偵聽程式的安全性 235
配置管理服務的 JMX 連接器的安全性 235
設定偵聽程式安全性特性 236
有關虛擬伺服器的管理主控台安全性作業 237
配置單次登入 (SSO) 237
有關連接器連線池的管理主控台作業 238
關於連接器連線池 238
關於安全性對映 239

建立安全性對映 239
編輯安全性對映 240
刪除安全性對映 241
使用證書和 SSL 241
關於證書檔案 241
變更證書檔案的位置 242
關於 Keytool 公用程式 243
關於 CertUtil 公用程式 243
產生伺服器證書 244
對數位證書進行簽名 244
使用 CA 證書 244
刪除證書 244
更多資訊 245

第 15 章 配置訊息安全性 247
關於訊息安全性 247
訊息安全性概況 247
瞭解 Application Server 中的訊息安全性 248
指定訊息安全性職責 248
關於安全性記號和安全性機制 249
訊息安全性術語字彙表 251
確保 Web 服務的安全 252
配置特定於應用程式的 Web 服務安全性 252
確保範例應用程式的安全 253
配置 Application Server 以實現訊息安全性 253
配置 JCE 提供者 254
用於訊息安全性的管理主控台作業 255
啓用訊息安全性的提供者 256
配置訊息安全性提供者 257
建立訊息安全性提供者 260
請求策略配置和回應策略配置的動作 261
刪除訊息安全性配置 262
刪除訊息安全性提供者 263
啓用用戶端應用程式的訊息安全性 263
設定應用程式用戶端配置的請求策略和回應策略 264
詳細資訊 265

第 16 章 作業事件 267
關於作業事件 267
何爲作業事件？ 267
J2EE 技術中的作業事件 268
有關作業事件的管理主控台作業 269

配置作業事件 269
作業事件恢復 269
作業事件逾時 270
作業事件記錄 270

第 17 章 配置 HTTP 服務 273
關於 HTTP 服務 273
什麼是 HTTP 服務？ 273
虛擬伺服器 274
HTTP 偵聽程式 275
有關 HTTP 服務的管理主控台作業 277
配置 HTTP 服務 277
配置 HTTP 服務存取記錄 279
配置 HTTP 服務請求處理執行緒 280
配置 HTTP 服務的持續作用子系統 280
配置 HTTP 服務連線池 281
爲 HTTP 服務配置 HTTP 通訊協定 281
爲 HTTP 服務配置 HTTP 檔案快取 282
有關虛擬伺服器的管理主控台作業 283
建立虛擬伺服器 283
編輯虛擬伺服器 285
刪除虛擬伺服器 286
有關 HTTP 偵聽程式的管理主控台作業 287
建立 HTTP 偵聽程式 287
編輯 HTTP 偵聽程式 289
刪除 HTTP 偵聽程式 289

第 18 章 配置物件請求代理程式 291
關於物件請求代理程式 291
CORBA 291
什麼是 ORB？ 292
IIOP 偵聽程式 292
有關 ORB 的管理主控台作業 292
配置 ORB 292
有關 IIOP 偵聽程式的管理主控台作業 293
建立 IIOP 偵聽程式 293
編輯 IIOP 偵聽程式 294
刪除 IIOP 偵聽程式 295

第 19 章 執行緒池 297
關於執行緒池 297
Application Server 中的執行緒池 297

有關執行緒池的管理主控台作業 298
建立執行緒池 298
編輯執行緒池 299
刪除執行緒池 299

第 20 章 配置記錄 301
關於記錄 301
記錄記錄 301
記錄程式名稱空間階層結構 302
有關記錄的管理主控台作業 304
配置一般記錄設定 304
配置記錄層級 305
檢視伺服器記錄 306

第 21 章 監視元件和服務 309
關於監視 309
監視 Application Server 309
監視概述 310
關於可監視物件的樹狀結構 310
應用程式樹 311
HTTP 服務樹 312
資源樹 312
連接器服務樹 313
JMS 服務樹 313
ORB 樹 313
執行緒池樹 314
關於受監視的元件和服務的統計資訊 314
EJB 容器統計資訊 315
Web 容器統計資訊 318
HTTP 服務統計資訊 319
JDBC 連線池統計資訊 320
JMS/ 連接器服務統計資訊 322
ORB 中連線管理程式的統計資訊 323
執行緒池統計資訊 323
作業事件服務統計資訊 324
Java 虛擬機器 (JVM) 統計資訊 324
生產 Web 容器 (PWC) 統計資訊 329
有關啓用和停用監視功能的管理主控台作業 336
使用管理主控台配置監視層級 336
使用 asadmin 工具配置監視功能 337
有關檢視監視資料的管理主控台作業 338
在管理主控台中檢視監視資料 338

使用 asadmin 工具檢視監視資料 340
使用 asadmin 工具檢視監視資料 340
瞭解和指定含點名稱 341
list 和 get 指令的範例 342
Petstore 範例 345
list 和 get 指令在所有層級上的預期輸出 348
使用 JConsole 354

第 22 章 Java 虛擬機器和進階設定 355
有關 JVM™ 設定的管理主控台作業 355
配置 JVM 一般設定 355
配置 JVM 類別路徑設定 356
配置 JVM 選項 357
停用安全性管理程式 358
配置 JVM 效能評測器設定 359
有關進階設定的管理主控台作業 359
設定進階網域屬性 359

附錄 A 編譯和配置 Apache Web Server 361
最低需求 361
適用於 Apache 1.3 的最低需求 361
適用於 Apache 2 的最低需求 362
安裝 SSL 可識別 Apache 363
編譯和建置 OpenSSL 364
使用 mod_ssl 配置 Apache 364
編譯和建置 Apache 365
編譯和建置 Apache 1.3 365
編譯和建置 Apache 2 366
啓動和停止 Apache 367

附錄 B 自動重新啟動網域或節點代理程式 369
在 UNIX 平台上自動重新啓動 369
在 Microsoft Windows 平台上自動重新啓動 370
自動重新啓動的安全性 371

附錄 C domain.xml 的含點名稱屬性 373
頂層元素 373
不能別名化的元素 375

前言

本指南描述如何配置和管理 Application Server。本前言包含有關以下主題的資訊：

- 本書的讀者
- 閱讀本書之前
- 本書的結構
- 本書使用的慣例
- 相關文件
- 存取 Sun 線上資源
- 與 Sun 技術支援部門聯絡
- 相關的協力廠商網站參考
- Sun 歡迎您提出意見和建議

本書的讀者

本**管理指南**適用於生產環境中的資訊技術管理員。本指南假設您已經熟悉以下主題：

- 基本系統管理作業
- 安裝軟體
- 使用 Web 瀏覽器
- 啓動資料庫伺服器
- 在終端機視窗中發佈指令

閱讀本書之前

Application Server 是 Sun Java™ Enterprise System 的元件之一，Sun Java™ Enterprise System 是支援分佈在網路或網際網路環境中的企業應用程式的軟體基礎架構。您應該熟悉隨 Sun Java Enterprise System 一起提供的文件，可以從 http://docs.sun.com/app/docs/prod/entsys.05q1 和 http://docs.sun.com/app/docs/prod/entsys.05q1?l=zh_TW 位置線上存取該文件。

本書的結構

本指南的組織結構與管理主控台的佈局相對應，管理主控台是基於瀏覽器的工具，用來管理 Application Server。每章都以概念性資訊開頭，隨後再說明如何使用管理主控台執行特定的作業。

本書使用的慣例

本小節中的表格描述了本書中使用的慣例。

印刷排版慣例

下表描述了本書對印刷排版的變更。

表 1 印刷排版慣例

字體	含義	範例
AaBbCc123	API 和語言元素、HTML 標籤、網站 URL、指令名稱、檔案名稱、目錄路徑名稱、螢幕畫面輸出和範例碼。	編輯 `.login` 檔案。 使用 `ls -a` 列示所有檔案。 `% You have mail。`
AaBbCc123	您所鍵入的資訊，與螢幕畫面輸出相對應。	`% su` `Password:`
AaBbCc123	新術語和要強調的文字。 將用實際名稱或值替代的指令或路徑名稱中的萬用字元。	這些稱爲**類別**選項。 請**勿**儲存該檔案。 檔案位於 *install-dir*`/bin` 目錄中。

表 1 印刷排版慣例 (續)

字體	含義	範例
「」	用於書名及章節名稱。	請閱讀 「使用者手冊」中的第 6 章。

符號

下表描述本書中使用的符號慣例。

表 2 符號慣例

符號	描述	範例	含義
[ ]	包含選擇性指令選項。	`ls [-l]`	無需 `-l` 選項。
{ | }	包含為所需指令選項提供的一組選擇。	`-d {y|n}`	`-d` 選項要求您使用 `y` 引數或 `n` 引數。
-	連接需同時按下的多個按鍵。	Control-A	同時按 Control 鍵和 A 鍵。
+	連接需連續按下的多個按鍵。	Ctrl+A+N	按 Control 按鍵，然後鬆開再依次按後面的按鍵。
>	表示圖形使用者介面中的功能表項目選取。	[檔案] > [新建] > [範本]	從 [檔案] 功能表中，選擇 [新建]。從 [新建] 子功能表中，選擇 [範本]。

預設路徑和檔案名稱

下表描述在本書中使用的預設路徑和檔案名稱。

表 3 預設路徑和檔案名稱

術語	描述

install_dir	依預設，Application Server 的安裝目錄位於以下位置：
	• Solaris™ 平台上的 Sun Java Enterprise System 的安裝：
	`/opt/SUNWappserver/appserver`
	• Linux 平台上的 Sun Java Enterprise System 的安裝：
	`/opt/sun/appserver/`
	• 其他 Solaris 和 Linux 的安裝 (非 root 使用者)：
	使用者的主目錄 `/SUNWappserver`
	• 其他 Solaris 和 Linux 的安裝 (root 使用者)：
	`/opt/SUNWappserver`
	• Windows 的所有安裝：
	系統磁碟機 :`\Sun\AppServer`

domain_root_dir	依預設，包含所有網域的目錄位於以下位置：
	• Solaris 平台上的 Sun Java Enterprise System 的安裝：
	`/var/opt/SUNWappserver/domains/`
	• Linux 平台上的 Sun Java Enterprise System 的安裝：
	`/var/opt/sun/appserver/domains/`
	• 所有其他安裝：
	install_dir`/domains/`

domain_dir	依預設，每個網域目錄都位於以下位置： *domain_root_dir*`/`*domain_dir* 在配置檔案中，您可能會看到 *domain_dir* 顯示如下： `${com.sun.aas.instanceRoot}`
instance_dir	依預設，每個實例目錄都位於以下位置： *domain_dir*`/`*instance_dir*

Shell 提示符號

下表描述本書中使用的 shell 提示符號。

表 4 Shell 提示符號

Shell	提示符號
UNIX 或 Linux 上的 C shell	*machine-name*%
UNIX 或 Linux 上的 C shell 超級使用者	*machine-name*#
UNIX 或 Linux 上的 Bourne shell 和 Korn shell	$
UNIX 或 Linux 上的 Bourne shell 和 Korn shell 超級使用者	#
Windows 指令行	C:\

相關文件

http://docs.sun.com℠ 網站使您可以存取 Sun 的線上技術文件。您可以瀏覽歸檔檔案或搜尋特定書名或主題。

您可以在 *install_dir*/docs/index.htm 中找到有關官方規格的 URL 目錄。此外，以下資源可能會有用。

一般 J2EE 資訊：

「The J2EE 1.4 Tutorial：」

http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html

「J2EE Blueprints：」

http://java.sun.com/reference/blueprints/index.html

「Core J2EE Patterns:Best Practices and Design Strategies」由 Deepak Alur、John Crupi 和 Dan Malks 編寫，Prentice Hall 出版

「Java Security」，由 Scott Oaks 編寫，O'Reilly 出版

使用 Servlet 和 JSP 檔案程式設計：

「Java Servlet Programming」，由 Jason Hunter 編寫，O'Reilly 出版

「Java Threads, 2nd Edition」，由 Scott Oaks 和 Henry Wong 編寫，O'Reilly 出版

使用 **EJB** 元件程式設計：

「Enterprise JavaBeans」，由 Richard Monson-Haefel 編寫，O'Reilly 出版

使用 **JDBC** 程式設計：

「Database Programming with JDBC and Java」，由 George Reese 編寫，O'Reilly 出版

「JDBC Database Access With Java:A Tutorial and Annotated Reference (Java Series)」，由 Graham Hamilton、Rick Cattell 和 Maydene Fisher 編寫

此文件集中包含的文件

Sun Java System Application Server 手冊爲線上檔案，有可攜式文件格式 (PDF) 和超文件標示語言 (HTML) 兩種格式可供選擇。

下表總結了包含在 Application Server 核心文件集中的文件。

表 5 此文件集中包含的文件

書名	描述
版本說明	軟體與文件的最新資訊。其中包括以表格形式對所支援的硬體、作業系統、JDK 和 JDBC/RDBMS 所做的全面概括。
快速入門指南	如何開始使用 Sun Java System Application Server 產品。
Installation Guide	安裝 Application Server 軟體及其元件。
Deployment Planning Guide	評估系統需求和企業狀況，確定以最適合您的站台的方式部署 Sun Java System Application Server。此外還描述了部署應用伺服器時應該註意的常見問題。
Developer's Guide	建立和實作 Application Server 上執行的 Java™ 2 Platform, Enterprise Edition (J2EE™ 平台) 應用程式，這些應用程式遵循針對 J2EE 元件和 API 的開放式 Java 標準模型。其中包括有關開發者工具、安全性、組合、部署、除錯和建立生命週期模組的一般資訊。
J2EE 1.4 Tutorial	使用 J2EE 1.4 平台技術和 API 開發 J2EE 應用程式，並將這些應用程式部署到 Sun Java System Application Server。
管理指南	從管理主控台配置、管理和部署 Application Server 子系統和元件。
High Availability Administration Guide	配置和管理 Sun Java System Application Server 高可用性功能。
Administration Reference	編輯 Sun Java System Application Server 的配置檔案 `domain.xml`。
Upgrade and Migration Guide	將應用程式遷移到新的 Sun Java System Application Server 程式設計模型，特別是從 Application Server 6.x 和 7 進行遷移。該指南還描述可導致與產品規格不相容的相鄰產品版本和配置選項之間的差異。

表 5 此文件集中包含的文件 (續)

書名	描述
Performance Tuning Guide	微調 Sun Java System Application Server 以改善效能。
Troubleshooting Guide	解決 Sun Java System Application Server 問題。
Error Message Reference	解決 Sun Java System Application Server 錯誤訊息。
Reference Manual	可用於 Sun Java System Application Server 的公用程式指令，以線上說明手冊樣式編寫。其中包括 `asadmin` 指令行介面。

其他伺服器文件

如需有關伺服器的其他文件，請移至以下位置：

- Message Queue 文件
  http://docs.sun.com/db?p=prod/s1.s1msgqu
- Directory Server 文件
  http://docs.sun.com/coll/DirectoryServer_04q2
- Web Server 文件
  http://docs.sun.com/coll/S1_websvr61_en

存取 Sun 線上資源

如需有關產品下載、專業服務、修補程式及支援和其他開發者資訊，請移至以下位置：

- 下載中心
  http://wwws.sun.com/software/download/
- 專業服務
  http://www.sun.com/service/sunps/sunone/index.html
- Sun 企業服務、Solaris 修補程式和支援
  http://sunsolve.sun.com/
- 開發者資訊
  http://developers.sun.com/prodtech/index.html

與 Sun 技術支援部門聯絡

如果您有關於此產品的技術問題，並且在此產品文件中找不到其答案，請移至 http://www.sun.com/service/contacting。

相關的協力廠商網站參考

Sun 對本文件中提到的協力廠商網站的可用性不承擔任何責任。對於此類站台或資源中的 (或通過它們獲得的) 任何內容、廣告、產品或其他材料，Sun 並不表示認可，也不承擔任何責任。對於因使用或依靠此類站台或資源中的 (或通過它們獲得的) 任何內容、產品或服務而造成的、宣稱的或連帶產生的實際或名義損壞或損失，Sun 概不負責，也不承擔任何責任。

Sun 歡迎您提出意見和建議

Sun 非常重視改善其文件品質，歡迎您提出意見和建議。

若要共用您的意見和建議，請移至 http://docs.sun.com 並按一下 [Send Comments]。在線上表單中，提供了文件標題和文件號碼。文件號碼是一個七位或九位的數字，可以在書的標題頁面或文件的頂部找到。例如，本書的標題是「Sun Java System Application Server 2005Q1 管理指南」，文件號碼是 819-1554。在您提出意見時，可能需要在表單中輸入英文版書名和文件號碼，本書的英文版文件號碼和書名爲：819-0215 和 Sun Java System Application Server 2005Q1 Administration Guide。

第 1 章

入門

本章描述 Sun Java™ System Application Server，並描述基本的管理作業。它包含以下小節：

- 關於 Sun Java System Application Server
- Application Server 配置
- 應用程式伺服器實例
- 配置變更

關於 Sun Java System Application Server

- 什麼是 Application Server？
- Application Server 架構
- 管理工具

什麼是 Application Server？

Application Server 爲開發、部署和管理企業應用程式提供了牢固的 J2EE 平台。主要功能包括作業事件管理、效能、可延伸性、安全性以及整合。Application Server 支援的服務包含從 Web 發佈到企業範圍內的作業事件處理，同時可讓開發者基於 JavaServer Pages (JSP™)、Java Servlet 以及企業 JavaBeans™ (EJB™) 技術建立應用程式。

Application Server Enterprise Edition 提供了進階叢集和防故障備用技術。這些功能讓您可以執行可延伸的且具有高可用性的 J2EE 應用程式。

- **叢集** — 叢集是一組應用程式伺服器實例，它們作爲一個邏輯實體一起工作。叢集中的每個 Application Server 實例都擁有相同的配置，並部署了相同的應用程式。

  透過將 Application Server 實例新增到叢集以實現水平延伸，從而增加系統容量。可以在不中斷服務的情況下將 Application Server 實例新增到叢集。HTTP、RMI/IIOP 和 JMS 負載平衡系統會將請求發行到叢集中運作狀態良好的 Application Server 實例中。

- **高可用性** — 可用性允許對叢集中的 Application Server 實例進行防故障備用保護。如果一個 Application Server 實例出現故障，則其他 Application Server 實例將接管指定給該故障伺服器的階段作業。階段作業資訊儲存在高可用性資料庫 (HADB) 中。HADB 支援 HTTP 階段作業和有狀態階段作業 Bean 的持續性。

Application Server 架構

本小節描述圖 1-1，此圖顯示了 Application Server 的高層級架構。

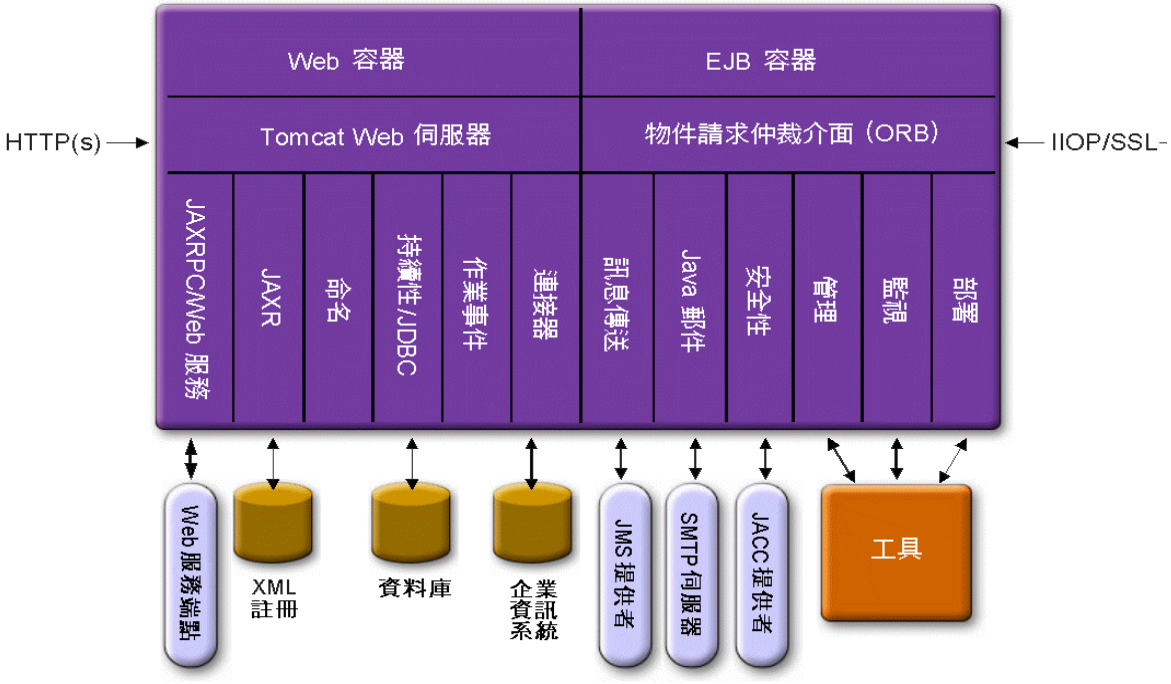


圖 1-1 Application Server 架構

- **容器** — 容器是一種執行階段環境，它爲 J2EE 元件提供安全性和作業事件管理等服務。圖 1-1 顯示了兩種類型的 J2EE 容器：Web 和 EJB。Web 元件 (例如 JSP 頁面和 Servlet) 在 Web 容器內執行。企業 Bean (EJB 技術的元件) 在 EJB 容器內執行。
- **用戶端存取** — 在執行階段瀏覽器用戶端透過 HTTP (在網際網路中使用的協定) 與 Web 伺服器進行通訊來存取 Web 應用程式。HTTPS 協定用於需要安全通訊的應用程式。企業 Bean 用戶端通過 IIOP 協定或 IIOP/SSL (安全) 協定與物件請求代理程式 (ORB) 進行通訊。Application Server 具有分別用於 HTTP 協定、HTTPS 協定、IIOP 協定和 IIOP/SSL 協定的偵聽程式。每個偵聽程式專用特定的連接埠號。
- **Web 服務** — 在 J2EE 平台上，可以部署一個 Web 應用程式，該應用程式可以提供由基於 XML 的 RPC 的 Java API (JAX-RPC) 實作的 Web 服務。J2EE 應用程式或元件還可以是其他 Web 服務的用戶端。應用程式通過用於 XML 登錄的 Java API (JAXR) 存取 XML 登錄。
- **用於應用程式的服務** — J2EE 平台旨在使容器爲應用程式提供服務。圖 1-1 顯示了以下服務：
  - 命名 — 命名和目錄服務將物件連結到名稱。J2EE 應用程式通過查找物件的 JNDI 名稱來找到物件。JNDI 代表 Java 命名和目錄介面 API。
  - 安全性 — Java 容器授權合約 (JACC) 是一組爲 J2EE 容器定義的安全性合約。基於用戶端的身份，容器限制對容器的資源和服務的存取。
- **作業事件管理** — 作業事件是不可分割的工作單元。例如，在銀行帳戶之間轉帳是一個作業事件。作業事件管理服務用於確定完全完成作業事件或將作業事件轉返。

存取外部系統

J2EE 平台使應用程式能夠存取應用程式伺服器之外的系統。應用程式通過稱爲資源的物件連線到這些系統。管理員的職責之一是資源配置。J2EE 平台可以通過以下 API 和元件存取外部系統：

- **JDBC** — 資料庫管理系統 (DBMS) 提供了用於儲存、組織和擷取資料的工具。大多數企業應用程式將資料儲存在關聯式資料庫中，這些應用程式透過 JDBC API 存取關聯式資料庫。由於資料庫中的資訊儲存在磁碟上並在應用程式結束之後仍然存在，因此通常將資料庫中的資訊稱爲持續性資訊。Application Server 束包括 PointBase DBMS。
- **訊息傳送** — 訊息傳送是軟體元件或應用程式之間的一種通訊方法。訊息傳送用戶端可以向任何其他用戶端傳送訊息，也可以從任何其他用戶端接收訊息。應用程式通過 Java 訊息傳送服務 (JMS) API 存取訊息傳送提供者。Application Server 包括一個 JMS 提供者。

- **連接器** — J2EE 連接器架構允許 J2EE 應用程式和現有企業資訊系統 (EIS) 之間的整合。應用程式通過稱爲連接器或資源介面的可攜式 J2EE 元件存取 EIS。
- **JavaMail** — 通過 JavaMail API，應用程式連線到 SMTP 伺服器以傳送和接收電子郵件。
- **伺服器管理** — 圖 1-1 的右下角顯示了由 Application Server 的管理員執行的一些作業。例如，管理員部署 (安裝) 應用程式並監視伺服器的效能。這些作業透過 Application Server 提供的管理工具來執行。
- 管理工具
- Application Server 包括三種管理工具：
  - 管理主控台
  - asadmin 公用程式
  - Application Server Management Extension (AMX)

管理主控台

管理主控台是一種基於瀏覽器的工具，具有易於導覽的介面和線上說明。本手冊提供使用管理主控台的逐步說明。必須執行管理伺服器才能使用管理主控台。

安裝 Application Server 時，您爲伺服器選擇了連接埠號，或使用的是預設連接埠號 4849。還指定了使用者名稱和主密碼。

若要啓動管理主控台，請在 Web 瀏覽器中鍵入以下特性：

```
https://hostname:port
```

例如：

```
https://kindness.sun.com:4949
```

如果管理主控台在安裝了 Application Server 的機器上執行，請將 `localhost` 指定爲主機名稱。

在 Windows 上，從 [開始] 功能表啓動 Application Server 管理主控台。

安裝程式將建立預設管理網域 (名爲 `domain1`)，並使用預設連接埠號 4849，還會建立一個獨立於網域管理伺服器 (DAS) 的實例。安裝之後，還可以建立其他管理網域。每個網域都具有自己的網域管理伺服器，該伺服器具有唯一的連接埠號。爲管理主控台指定 URL 時，請確定使用要管理的網域的連接埠號。

如果配置中包括遠端伺服器實例，請建立節點代理程式以便管理和簡化遠端伺服器實例。節點代理程式負責建立、啓動、停止和刪除伺服器實例。使用指令行介面 (CLI) 指令可以設定節點代理程式。

asadmin 公用程式

asadmin 公用程式是一個指令行工具。使用 asadmin 公用程式和與其關聯的指令可以執行在管理主控台中可執行的相同作業集。例如，啓動和停止網域、配置伺服器以及部署應用程式。

可以在 Shell 的指令提示符號下使用這些指令，也可以從其他程序檔和程式呼叫這些指令。使用這些指令可以自動執行重複的管理作業。

若要啓動 asadmin 公用程式，請使用：

```
$ asadmin
```

若要列示 asadmin 中的可用指令，請使用：

```
asadmin> help
```

也可以在 Shell 的指令提示符號下發佈 asadmin 指令：

```
$ asadmin help
```

若要檢視指令的語法和範例，請在指令名稱後面鍵入 help。例如：

```
asadmin> help create-jdbc-resource
```

給定指令的 asadmin help 資訊將顯示此指令的 Unix 線上說明手冊。也可以使用 HTML 格式檢視這些線上說明手冊。

Application Server Management Extension (AMX)

Sun Java System **A**pplication Server **M**anagement e**X**tension 是一個 API，它顯示所有 Application Server 配置並將 JMX 管理 Bean 當作實作 AMX 介面的、易於使用的用戶端動態代理程式來進行監視。

如需有關使用 Application Server Management Extension 的更多資訊，請參閱「Sun Java System Application Server Developer's Guide」中的「Using the Java Management Extensions (JMX) API」一章。

Application Server 配置

- 配置 Application Server
- 配置網域
- 啓動網域
- 重新啓動伺服器或網域
- 停止網域
- 重新建立網域管理伺服器

配置 Application Server

Application Server 網域是爲協助管理員管理系統配置而建立的邏輯或實體單元。一個網域分爲很多包括實例和節點代理程式的較小單元。伺服器實例是在單一實體機器上執行 Application Server 的單一 Java 虛擬機 (JVM)。每個網域都有一個或多個實例。網域還必須至少有一個相關聯的節點代理程式才能使實例正常工作。可以將網域群組在一起以建立一個叢集。叢集使管理員可以管理多組硬體和軟體。

配置網域

管理網域提供了基本的安全性結構，不同的管理員可以藉此管理應用程式伺服器實例的特定群組 (網域)。透過將伺服器實例分組到單獨的網域中，不同的組織和管理員可以共用單一 Application Server 安裝。每個網域都有自己的獨立於其他網域的配置、記錄檔和應用程式部署區域。如果變更某個網域的配置，其他網域的配置不會受到影響。

每個管理主控台階段作業均允許您配置和管理網域。如果建立了多個網域，則必須啓動另外一個管理主控台階段作業以管理每個網域。每個網域都具有自己的網域管理伺服器 (DAS)，該伺服器具有唯一的連接埠號。每個管理網域可以有多個應用程式伺服器實例。但是，一個應用程式伺服器實例只能屬於一個網域。安裝 Application Server 時，將自動建立名爲 `domain1` 的管理網域。

建立網域

網域是使用 create-domain 指令建立的。以下範例指令將建立名爲 mydomain 的網域。管理伺服器在連接埠 1234 上進行偵聽，管理使用者名爲 hanan。該指令提示您輸入管理密碼和主密碼。

```
$ asadmin create-domain --adminport 80 --adminuser hanan mydomain
```

若要爲 mydomain 網域啓動管理主控台，請在瀏覽器中輸入以下 URL：

http://*hostname*:80

對於前面的 create-domain 範例，網域的記錄檔、配置檔案和部署的應用程式現在位於以下目錄中：

install_dir/domains/mydomain

若要在其他位置建立網域的目錄，請指定 --domaindir 選項。如需完整的指令語法，請鍵入 asadmin help create-domain。

刪除網域

網域是使用 asadmin delete-domain 指令刪除的。僅具有網域管理權限的作業系統使用者 (或 root 使用者) 才能成功地執行該指令。例如，若要刪除名爲 mydomain 的網域，請鍵入以下指令：

```
$ asadmin delete-domain mydomain
```

列示網域

可以使用 asadmin list-domains 指令找到建立在機器中的網域。若要列示預設 *install_dir*/domains 目錄中的網域，請鍵入以下指令：

```
$ asadmin list-domains
```

若要列示在其他目錄中建立的網域，請指定 --domaindir 選項。

啓動網域

啓動網域時，將啓動管理伺服器和應用程式伺服器實例。啓動應用程式伺服器實例之後，應用程式伺服器實例將持續執行、偵聽並接受請求。必須單獨啓動各個網域。

若要啓動網域，請鍵入 asadmin start-domain 指令並指定網域名稱。例如，若要啓動預設網域 (domain1)，請鍵入以下特性：

```
$ asadmin start-domain domain1
```

如果只有一個網域，則可以省略網域名稱。如需完整的指令語法，請鍵入 asadmin help start-domain。如果省略了密碼資料，系統將提示您提供此資料。

若要在 Windows 中啟動預設網域，請執行以下步驟：

在 Windows [開始] 功能表中，依次選取 [程式集] -> [Sun Microsystems] -> [Application Server] -> [啓動管理伺服器]。

重新啟動伺服器或網域

重新啓動伺服器與重新啓動網域相同。若要重新啓動網域或伺服器，請停止然後再啓動網域。

停止網域

停止網域將關閉該網域的管理伺服器和應用程式伺服器實例。停止網域時，伺服器實例將停止接受新的連線，然後等待所有未完成的連線完成。由於伺服器實例必須完成其關閉程序，因此該程序需要幾秒鐘時間。停止網域時，管理主控台或大多數 asadmin 指令都無法使用。

若要停止網域，請鍵入 asadmin stop-domain 指令並指定網域名稱。例如，若要停止預設網域 (domain1)，請鍵入以下特性：

```
$ asadmin stop-domain domain1
```

如果只有一個網域，則網域名稱是選擇性的。如需完整的語法，請鍵入 asadmin help stop-domain。

若要使用管理主控台停止網域，請執行以下步驟：

- 在樹形元件中，選取 [獨立實例] 節點下的伺服器 (管理伺服器)。
- 在 [一般資訊] 頁面中，按一下 [停止伺服器]。

若要在 Windows 中停止預設網域，請執行以下步驟：

從 [開始] 功能表中，依次選取 [程式集] -> [Sun Microsystems] -> [Application Server] -> [停止管理伺服器]。

重新建立網域管理伺服器

若要進行鏡像並提供網域管理伺服器 (DAS) 的工作副本，您必須具有：

- 一台包含原始 DAS 的機器 (machine1)。
- 一台包含叢集的機器 (machine2)，該叢集具有執行應用程式並滿足用戶端需要的伺服器實例。該叢集是使用第一台機器上的 DAS 配置的。
- 一台備份機器 (machine3)，當第一台機器當機時，需要在該備份電腦上重新建立 DAS。

備註 必須保留一份第一台機器上的 DAS 的備份。請使用 `asadmin backup-domain` 來備份目前網域。

DAS 遷移的步驟

以下步驟用於將網域管理伺服器從第一台機器 (machine1) 遷移到第三台機器 (machine3)：

1. 透過在第三台機器上安裝與在第一台機器上安裝的應用程式伺服器相同的應用程式伺服器，設定第三台機器。

   爲了可以在第三台機器上正確地復原 DAS 並且不會發生路徑衝突，您必須執行此操作。

   - 使用指令行 (互動式) 模式來安裝應用程式伺服器管理套裝軟體。若要啓動互動式指令行模式，請使用 `console` 選項呼叫安裝程式：

     ***./ bundle_filename* `-console`**

     若要使用指令行介面進行安裝，您必須具有 root 許可權。

   - 若要安裝預設網域，請取消選取該選項。

     只有具有相同架構並具有**完全**相同的安裝路徑 (即使用相同的 *install_dir*) 的兩台機器才支援備份網域的復原。

2. 將第一台機器上的備份 ZIP 檔案複製到第三台機器上的 *`install_dir`*`/domains` 目錄中。也可以通過 FTP 方式複製檔案。

3. 執行 `asadmin restore-domain` 指令以將 ZIP 檔案復原到第三台機器：

   ```
   asadmin restore-domain --filename
   install_dir/domains/sjsas_backup_v00001.zip domain1
   ```

   可以備份任何網域。但是，在重新建立網域時，網域名稱應與原始網域名稱相同。

4. 變更第三台機器上的 *install_dir*/domains/domain1/generated/tmp 目錄的許可權以與第一台機器上相同目錄的許可權匹配。

   該目錄的預設許可權爲：?drwx------?(或 700)。

   例如：

   chmod 700 *install_root*/domains/*domain1*/generated/tmp

   以上範例假定您備份的是 *domain1*。如果備份的是其他名稱的網域，則應當用要備份的網域的名稱取代上面的 *domain1*。

5. 變更第三台機器的 domain.xml 中的主機特性值：

6. 更新第三台機器上的 *install_root*/domains/*domain1*/config/domain.xml。

   例如：

   搜尋 *machine1* 並將其替代爲 *machine3*。這樣，您就可以將：

   <jmx-connector><property name=*client-hostname* value=*machine1*/>...

   變更爲：

   <jmx-connector><property name=*client-hostname* value=*machine3*/>...

7. 將：

   <jms-service... host=*machine1*.../>

   變更爲：

   <jms-service... host=*machine3*.../>

8. 在 machine3 上啓動復原的網域：

   asadmin start-domain --user *admin_user* --password *admin_password* domain1

9. 在 machine2 上變更節點代理程式下的 DAS 主機特性值：

10. 變更 machine2 中的 *install_dir*/nodeagents/*nodeagent*/agent/config/das.properties 中的 agent.das.host 特性值。

11. 在 machine2 上重新啓動節點代理程式。

備註 使用 asadmin start-instance 指令啓動叢集實例，以使這些實例與復原網域同步。

應用程式伺服器實例

- 關於應用程式伺服器實例
- 定義 Application Server 實例
- 關於獨立實例
- 建立實例
- 啓動實例
- 恢復作業事件
- 停止實例

關於應用程式伺服器實例

Sun Java System Application Server 在安裝時將建立一個稱爲 server 的應用程式伺服器實例。如果需要，可以刪除此伺服器實例，並建立一個其他名稱的新實例。

每個 Sun Java System Application Server 實例都有自己的 J2EE 配置、J2EE 資源、應用程式部署區域和伺服器配置設定。對一個應用程式伺服器實例所做的變更不會影響其他應用程式伺服器實例。在一個管理網域內，您可以擁有多個應用程式伺服器實例。

對於許多使用者而言，一個應用程式伺服器實例就符合他們的需要了。不過，依據您的環境，您可能想建立一個或多個附加的應用程式伺服器實例。例如，在開發環境下，您可以使用不同的應用程式伺服器實例來測試不同的 Sun Java System Application Server 配置，或比對和測試不同的應用程式部署。由於您可以輕易地加入或刪除應用程式伺服器實例，因而可以在開發時透過這些實例建立暫時的「沙箱」區域進行試驗。

此外，對於每個應用程式伺服器實例，您也可以建立虛擬伺服器。在單一安裝的應用程式伺服器實例內，您可以爲公司或個人提供網域名稱、IP 位址以及某些管理功能。對於使用者而言，看起來好像使用者有自己的 Web 伺服器，但沒有硬體和基本的伺服器維護功能。這些虛擬伺服器不擴充應用程式伺服器實例。如需有關虛擬伺服器的更多資訊，請參閱配置 JVM 一般設定。

在作業部署中，您可以使用虛擬伺服器代替多重應用程式伺服器實例，用於多種目的。但是，如果虛擬伺服器不能滿足需求，您也可以使用多個應用程式伺服器實例。

Sun Java System Application Server 實例不會自動啓動。啓動某個實例後，該實例將一直執行，直至您將其停止。若您停止一個應用程式伺服器實例，該實例將停止接受新連線，然後等待所有未完成的連線完成。如果您的機器當機或離線，伺服器將結束，其正在處理的任何請求均可能遺失。

定義 Application Server 實例

應用程式伺服器實例構成了應用程式部署的基礎。每個實例均屬於單一網域，並有自己的目錄結構、配置和已部署的應用程式。每個伺服器實例還包含了 J2EE 平台的 Web 和 EJB 容器。每個新的伺服器實例必須包含對節點代理程式名稱的參考，該名稱定義實例將要駐留的機器。

可以建立三種類型的伺服器實例。每個伺服器實例只能是其中的一種類型：

- 在**獨立伺服器**實例中，其他任何伺服器實例或叢集不能共用其配置。
- 在**共用伺服器實例**中，其他實例或叢集可以共用其配置。
- 在**叢集伺服器實例**中，叢集中的其他實例可以共用其配置。

  圖 1-2 詳細顯示了一個應用程式伺服器實例。應用伺服器實例是 Application Server 企業版的叢集、負載平衡和階段作業持續性功能中的建置區段。

- **定義和使用叢集** — 叢集是一組共用相同的應用程式、資源和配置資訊集的伺服器實例。伺服器實例可以只屬於一個叢集。叢集用於通過在多台機器上分佈負載來增強負載平衡，並通過實例層級的防故障備用來提供高可用性。

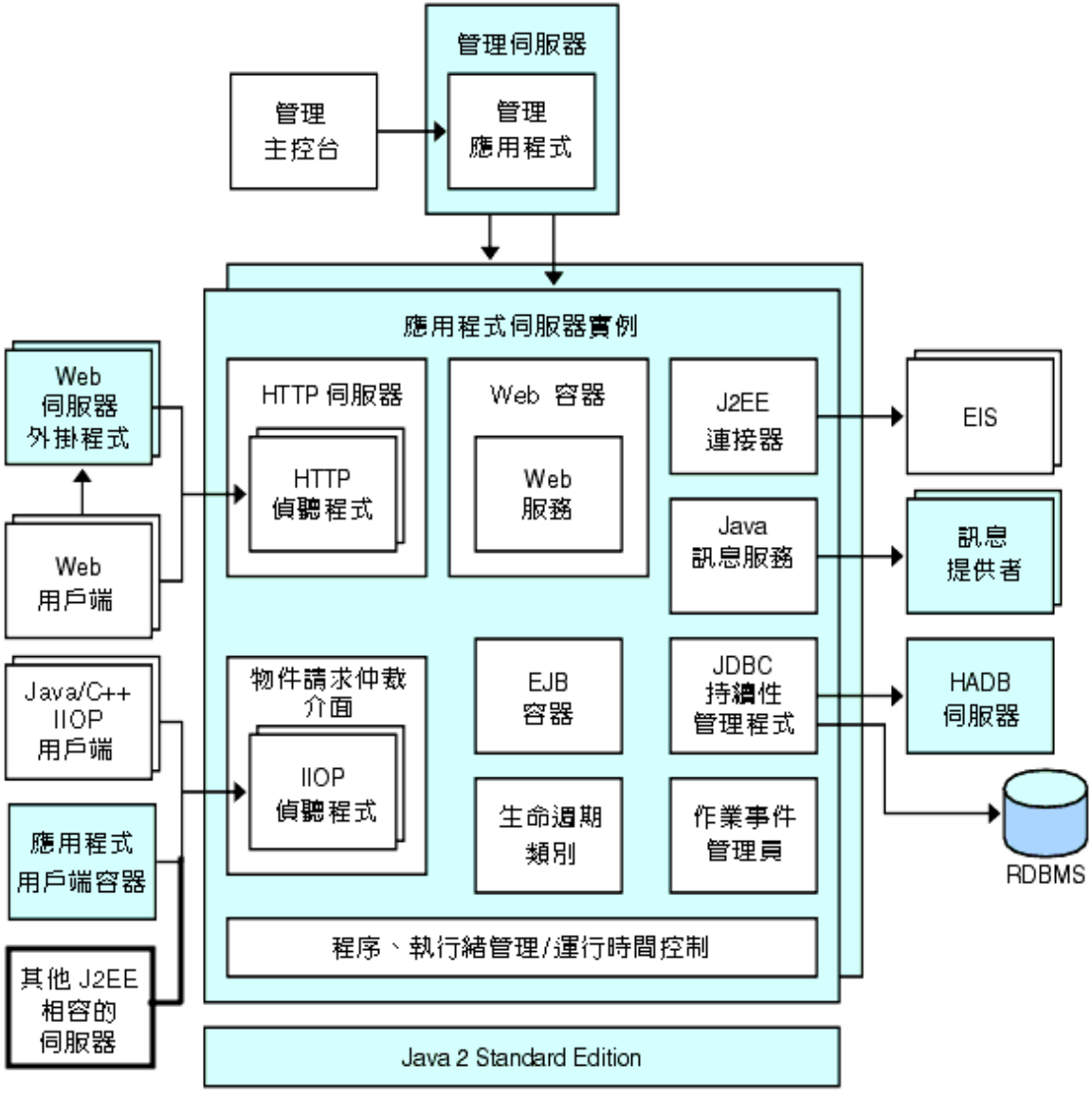


圖 1-2 Sun Java System Application Server 實例

關於獨立實例

Sun Java System Application Server 實例不會自動啓動。啓動某個實例後，該實例將一直執行，直至您將其停止。若您停止一個應用程式伺服器實例，該實例將停止接受新連線，然後等待所有未完成的連線完成。如果您的機器當機或離線，伺服器將結束，其正在處理的任何請求均可能遺失。

檢視一般伺服器資訊

透過 [一般] 標籤可以執行以下作業：

- 按一下 [啓動實例] 以啓動實例。
- 按一下 [停止實例] 以停止實例。
- 按一下 [檢視記錄檔]，以開啓伺服器記錄檢視器。
- 按一下 [自動重建記錄檔]，以自動重建實例的記錄檔。該動作將排程記錄檔以進行自動重建。實際的自動重建將在下一次向記錄檔寫入項目時發生。預設伺服器 (DAS) 的自動重建將立即發生，但其他獨立伺服器的自動重建將延遲。
- 按一下 [JNDI 瀏覽] 以瀏覽正在執行的實例的 JNDI 樹。
- 按一下 [恢復作業事件] 以恢復未完成的作業事件。

此外，您可以選取以下標籤以執行其他作業：

- [應用程式] 標籤：部署選取的應用程式。
- [資源] 標籤：管理選取的資源。
- [特性] 標籤：配置實例特定的特性。
- [監視] 標籤：檢視 JVM、伺服器、執行緒池、HTTP 服務和作業事件服務的監視資料。
- [進階] 標籤：設定用於部署應用程式的一般特性。

管理應用程式

通過 [應用程式] 標籤，您可以啓用、停用和部署與實例相關聯的選取應用程式。

若要部署應用程式，請執行以下步驟：

1. 選取所需應用程式的核取方塊。
2. 從 [部署] 下拉式功能表中，選取要部署的應用程式模組的類型：

- 企業應用程式：位於 EAR (企業應用程式歸檔) 檔案或目錄中的 J2EE 應用程式。
- Web 應用程式：封裝在 WAR (Web 應用程式歸檔) 檔案或目錄中的 Web 資源集合 (例如 JavaServer Pages (JSP)、servlet 和 HTML 頁面等)。
- EJB 模組：包含在 EJB JAR (Java 歸檔) 檔案或目錄中的一個或多個企業 JavaBean (EJB)。
- 連接器模組：連線至企業資訊系統 (EIS) 並封裝在 RAR (資源介面歸檔) 檔案或目錄中。
- 生命週期模組：在伺服器生命週期中有一個或多個事件觸發時會執行作業。
- 應用程式用戶端模組：也稱爲 J2EE 應用程式用戶端 JAR 檔案，它包含用戶端的伺服器端常式。

管理資源

通過 [資源] 標籤，您可以啓用、停用和建立與實例關聯的新資源類型。

若要建立新的資源類型，請執行以下步驟：

1. 選取所需資源的核取方塊。
2. 從 [新建] 下拉式功能表中，選取要建立並與該實例相關聯的資源類型：
   - JDBC：爲應用程式提供連線至資料庫的方法。
   - 持續性管理員：具有容器管理的持續性 Bean (用於向下相容) 的應用程式需要持續性管理員。
   - JMS 連線工廠：允許應用程式以程式化方式建立其他 JMS 物件的物件。
   - JMS 目標：表示 JavaMail API 中的郵件階段作業，JavaMail API 提供了一個獨立於平台和協定的框架來建置郵件和訊息傳送應用程式。
   - JavaMail：提供了一個獨立於平台和協定的框架來建置郵件和訊息傳送應用程式。
   - 自訂：表示具有已定義的 JNDI 子環境、資源類型和工廠類別的非標準資源。
   - 外部：使應用程式能夠查找位於簡易資料存取協定 (LDAP) 系統儲存庫中的外部資源物件。
   - 連接器：可爲應用程式提供企業資訊系統 (EIS) 連線的程式物件。
   - 管理物件：配置 JSR-160 相容的遠端 JMX 連接器

Administration Server 進階設定

Administration Server 進階設定使您可以設定用於部署應用程式的一般特性。這些特性可讓您確保並監視所部署應用程式的變更已被偵測出並且修改的類別已被重新載入。

設定應用程式配置

如果啓用了動態重新載入，伺服器將定期檢查已部署的應用程式檔案中的變更並自動重新載入包含變更的應用程式。動態重新載入在開發環境中非常有用，因爲它能快速測試程式碼變更。但在生產環境中，動態重新載入可能會使效能降低。

動態重新載入旨在用於開發環境。它與階段作業持續性 (一種生產環境功能) 不相容。如果啓用了動態部署，請勿啓用階段作業持續性。

備註 動態重新載入僅適用於預設伺服器實例。

若要在 [應用程式配置] 頁面中配置動態重新載入，請配置以下項目：

- 重新載入：使用 [已啓用] 核取方塊來啓用或停用動態重新載入。
- 重新載入輪詢間隔：指定伺服器檢查已部署的應用程式中的變更的頻率。
- 管理階段作業逾時：指定管理階段作業逾時且必須重新登入之前的時間。

設定自動部署

自動部署功能使您能夠透過將預先封裝的應用程式或模組複製到 *install_dir*/domains/*domain_dir*/autodeploy 目錄來部署該應用程式或模組。

例如，將名爲 hello.war 的檔案複製到 *install_dir*/domains/domain1/autodeploy 目錄。若要取消部署應用程式，請從 autodeploy 目錄中移除 hello.war 檔案。

自動部署功能旨在用於開發環境。它與階段作業持續性 (一種生產環境功能) 不相容。如果啓用了動態部署，請勿啓用階段作業持續性

備註 自動部署僅適用於預設伺服器實例。

若要在 [應用程式配置] 頁面中配置自動部署設定，請執行以下步驟：

1. 透過選取或取消選取 [已啓用] 核取方塊來啓用或停用自動部署。
2. 在 [自動部署輪詢間隔] 欄位中，指定伺服器檢查自動部署目錄中的應用程式檔案或模組檔案的頻率。變更輪詢間隔不會影響部署應用程式或模組所需的時間。
3. 在 [自動部署目錄] 中，如果指定建立應用程式的目錄，則不必將檔案複製到預設自動部署目錄中。

   預設目錄是伺服器實例的根目錄中名爲 `autodeploy` 的目錄。依預設，可使用變數自動變更多個伺服器實例的目錄。
4. 若要在部署之前執行檢驗器，請選取 [已啓用檢驗器] 核取方塊。檢驗器將檢查檔案的結構和特性。大型應用程式的檢驗通常會很費時。
5. 若要預編譯 JSP 頁面，請選取 [JSP] 核取方塊。如果未選取此核取方塊，則首次存取 JSP 頁面時會在執行階段編譯這些頁面。由於編譯通常很費時，因此在生產環境中請選取此核取方塊。

設定其他特性

按一下 [新增特性] 按鈕以指定其他設定。

設定網域屬性

包括以下網域屬性。

表 1-1 網域屬性值

特性	定義
com.sun.aas.installRoot	應用程式伺服器的安裝目錄。
com.sun.aas.instanceRoot	伺服器實例的頂層目錄。
com.sun.aas.hostName	主機 (機器) 的名稱。
com.sun.aas.javaRoot	.J2SE 安裝目錄。
com.sun.aas.imqLib	Sun Java System Message Queue 的程式庫目錄。
com.sun.aas.configName	伺服器實例正在使用的配置的名稱。
com.sun.aas.instanceName	伺服器實例的名稱。該特性對於 default-config 不可用，但可用於自訂的配置。
com.sun.aas.clusterName	叢集的名稱。僅在叢集的伺服器實例上設定了該特性。該特性對於 default-config 不可用，但可用於自訂的配置。
com.sun.aas.domainName	網域的名稱。該特性對於 default-config 不可用，但可用於自訂的配置。

實例特定的配置特性

實例特定的配置特性將置換此實例的值。

備註 預設值定義在與實例關聯的配置中。

若要將值復原爲預設值，請執行以下步驟：

1. 移除置換值。
2. 按一下 [儲存]。

如果未設定置換值，則使用預設值。

新增或刪除實例特性

若要新增特性，請執行以下步驟：

- 按一下 [新增特性] 按鈕以指定其他設定。

可以使用以下屬性名稱 / 值對來配置資源：

表 1-2 屬性名稱 / 值對

特性	定義
HTTP_LISTENER_PORT	此連接埠用於偵聽 HTTP 請求。此特性指定 http-listener-1 的連接埠號。有效值為 1 到 65535。在 UNIX 中，建立在 1 到 1024 連接埠上進行偵聽的插槽要求具有超級使用者權限。
HTTP_SSL_LISTENER_PORT	此連接埠用於偵聽 HTTPS 請求。此特性指定 http-listener-2 的連接埠號。有效值為 1 到 65535。在 UNIX 中，建立在 1 到 1024 連接埠上進行偵聽的插槽要求具有超級使用者權限。
IIOP_LISTENER_PORT	此特性指定 orb-listener-1 偵聽 IIOP 連接的 ORB 偵聽程式連接埠。
IIOP_SSL_LISTENER_PORT	此連接埠用於保護 IIOP 連線。
JMX_SYSTEM_CONNECTOR_PORT	此特性指定 JMX 連接器進行偵聽的連接埠號。有效值為 1 到 65535。在 UNIX 中，建立在 1 到 1024 連接埠上進行偵聽的插槽要求具有超級使用者權限。
IIOP_SSL_MUTUALAUTH_PORT	此特性指定稱為 SSL_MUTUALAUTH 的 IIOP 偵聽程式偵聽 IIOP 連線的 ORB 偵聽程式連接埠。

若要刪除特性，請執行以下步驟：

1. 按一下要刪除的特性。
2. 按一下 [刪除特性] 按鈕。

建立實例

若要建立實例，請執行以下步驟：

1. 在樹形元件中，選取 [獨立實例] 節點。
2. 在 [獨立伺服器實例] 頁面中，按一下 [新建]。
3. 在 [名稱] 欄位中，識別新實例的唯一名稱。
4. 選擇一個節點代理程式。

   必須在節點代理程式的主機上使用 `asadmin start-node-agent` 指令啓動節點代理程式，以便使正在建立的伺服器實例能夠與該節點代理程式相關聯。
5. 選取所需的配置。
   - 參考現有配置。不新增新配置。
   - 建立現有配置的副本。新增伺服器實例或叢集時，將新增新配置。
6. 依預設，在建立新實例時，其配置是從 default-config 配置中複製的。若要從其他配置進行複製，請在建立新實例時指定要複製的配置。
7. 對於伺服器實例，新配置的名稱爲 `instance_name-config`。

   default-config 配置爲預設配置，它用作建立獨立伺服器實例的範本。非叢集伺服器實例或叢集不允許參考 default-config 配置；只能對其進行複製以建立新配置。編輯預設配置，以確定從預設配置複製而來的新配置具有正確的初始設定。

等效的 `asadmin` 指令爲：`create-instance`。

啟動實例

若要啓動實例，請執行以下步驟：

1. 在樹形元件中，展開 [獨立實例] 節點。
2. 選取要啓動的實例。
3. 在 [一般] 標籤上，按一下 [啓動] 以啓動實例。

必須先使用 asadmin start-node-agent 指令啓動與實例相關聯的節點代理程式，才能成功啓動該實例。

啓動實例後，便可以從 [一般] 標籤上執行以下作業：

- 按一下 [停止實例] 以停止實例。
- 按一下 [JNDI 瀏覽] 以檢視該實例的 JNDI 項目。
- 按一下 [檢視記錄檔] 以檢視 [記錄檢視器] 並指定記錄選項。
- 按一下 [自動重建記錄檔]。
- 按一下 [恢復作業事件] 以恢復未完成的作業事件。

等效的 asadmin 指令爲：start-instance。

恢復作業事件

由於伺服器當機或資源管理員當機，作業事件可能未完成。完成這些中斷的作業事件並將其從故障中恢復至關重要。Application Server 旨在在伺服器啓動時從故障中恢復並完成這些作業事件。

如果選取的伺服器正在執行，則將由該伺服器執行恢復。如果選取的伺服器未執行，則將由選取的目標伺服器執行恢復。

停止實例

若要停止實例，請執行以下步驟：

1. 在樹形元件中，展開 [獨立實例] 節點。
2. 選擇要停止的實例。
3. 在 [一般] 標籤上，按一下 [停止] 以停止實例。

等效的 asadmin 指令爲：stop-instance。

關閉伺服器實例

若要關閉管理伺服器，請執行以下步驟：

1. 在樹形元件中，選取 [獨立實例] 節點。
2. 選取 [Administation Server 實例]

3. 按一下 [停止]。

將顯示確認對話方塊以確認您要關閉管理伺服器。

配置變更

- 變更 Application Server 配置
- Application Server 中的連接埠
- 檢視連接埠號
- 變更管理伺服器連接埠
- 變更 HTTP 連接埠
- 變更 IIOP 連接埠
- 使用管理服務配置 JMX 連接器
- 編輯 JMX 連接器配置
- 變更 J2SE 軟體

變更 Application Server 配置

在進行以下任何配置變更時，請重新啓動伺服器以使變更生效：

- 變更 JVM 選項
- 變更連接埠號
- 管理 HTTP 服務、IIOP 服務和 JMS 服務
- 管理執行緒池

若需說明，請參閱「重新啓動伺服器或網域」。

如果使用動態配置，大多數變更在伺服器執行時即可生效。若要進行以下配置變更，請勿重新啓動伺服器：

- 部署和取消部署應用程式
- 新增或移除 JDBC、JMS 與連接器資源和池
- 變更記錄層級
- 新增檔案範圍使用者

- 變更監視層級
- 啓用和停用資源和應用程式

請註意，asadmin reconfig 指令已停用，並且不再需要此指令。配置變更將動態套用至伺服器。

Application Server 中的連接埠

表 1-3 描述 Application Server 的連接埠偵聽程式。

表 1-3 使用連接埠的 Application Server 偵聽程式

偵聽程式	預設連接埠號	描述
管理伺服器	4848	透過管理主控台和 asadmin 公用程式存取網域的管理伺服器。對於管理主控台，請在瀏覽器的 URL 中指定連接埠號。遠端執行 asadmin 指令時，請使用 --port 選項指定連接埠號。
HTTP	8081	Web 伺服器偵聽連接埠上的 HTTP 請求。若要存取已部署的 Web 應用程式和服務，用戶端應連線到此連接埠。
HTTPS	8181	為安全通訊配置的 Web 應用程式在單獨的連接埠上進行偵聽。
IIOP		企業 Bean (EJB 元件) 的遠端用戶端通過 IIOP 偵聽程式存取 Bean。
IIOP、SSL		另一個連接埠由為安全通訊配置的 IIOP 偵聽程式使用。
IIOP、SSL 和相互認證		另一個連接埠由為相互 (用戶端和伺服器) 認證配置的 IIOP 偵聽程式使用。

檢視連接埠號

1. 在樹形元件中，選取 [獨立實例] 節點下的一個實例。
2. 選取 [特性] 標籤。
3. 在 [實例特定的] 頁面中，識別了預設連接埠號。可以設定配置以置換這些值。

變更管理伺服器連接埠

1. 在樹形元件中，展開 [配置] 節點。
2. 展開 [server-config (管理配置)] 節點

3. 展開 [HTTP 服務] 節點。
4. 展開 [HTTP 偵聽程式] 節點。
5. 選取 [admin-listener] 節點。
6. 在 [編輯 HTTP 偵聽程式] 頁面中，變更 [偵聽程式連接埠] 欄位的值。
7. 重新啓動伺服器。

變更 HTTP 連接埠

1. 在樹形元件中，展開 [HTTP 服務] 節點。
2. 展開 [HTTP 偵聽程式] 節點。
3. 選取要變更其連接埠號的 HTTP 偵聽程式。
4. 在 [編輯 HTTP 偵聽程式] 頁面中，變更 [偵聽程式連接埠] 欄位的值。
5. 按一下 [儲存]。
6. 重新啓動伺服器。

變更 IIOP 連接埠

1. 在樹形元件中，展開 [配置] 節點。
2. 展開 [server-config (管理配置)] 節點
3. 展開 [ORB] 節點。
4. 展開 [IIOP 偵聽程式] 節點。
5. 選擇要變更其連接埠號的偵聽程式。
6. 在 [編輯 IIOP 偵聽程式] 頁面中，變更 [偵聽程式連接埠] 欄位的值。
7. 按一下 [儲存]。
8. 重新啓動伺服器。

使用管理服務配置 JMX 連接器

使用管理服務爲遠端伺服器實例配置 JSR-160 相容遠端 JMX 連接器，該連接器處理網域管理伺服器和節點代理程式 (管理主機電腦上的伺服器實例) 之間的通訊。

管理服務確定伺服器實例是一般實例、網域管理伺服器 (DAS)，還是兼具兩者。DAS 與 J2EE 伺服器實例類似，只是使用者應用程式和資源不會被部署到 DAS (儘管它能夠處理使用者應用程式請求)。DAS 和 J2EE 伺服器實例之間唯一比較顯著的區別在於：前者不能是叢集的一部分，而是伺服器實例的同質單元。

若要配置 JMX 連接器，請執行以下步驟：

1. 從樹中選擇 [配置]。
2. 選取要配置的實例：
    a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例、伺服器，請選擇 [server-config] 節點。
    b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 [default-config] 節點。
3. 從樹中選取 [管理服務]。
4. 從 [類型] 下拉式功能表中，選取要管理服務配置的類型：DAS、DAS 和伺服器或者伺服器。選取 [DAS 和伺服器] 與選取 [DAS] 作用相同。[伺服器] 選項將選取非 DAS 伺服器實例。
5. 在 [JMX 連接器名稱] 欄位中，輸入內部使用的 JMX 連接器的名稱。連接器的名稱爲 system。

編輯 JMX 連接器配置

使用 [編輯 JMX 連接器] 螢幕，您可以編輯 JSR 160 相容的 JMX 連接器的配置。

1. 從樹中選擇 [配置]。
2. 選擇要配置的實例：
    a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例、伺服器，請選擇 [server-config] 節點。
    b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 [default-config] 節點。
3. 展開 [管理服務] 節點並按一下 [系統]，它是內部使用的 JMX 連接器。

4. 輸入 JMX 連接器伺服器的連接埠。JMX 服務 URL 是協定、連接埠和位址的函式，如 JSR 160 1.0 規格所定義
5. 輸入此 JMX 連接器應該支援的協定。Application Server 版本 8.1 僅支援 rmi_jrmp 協定。
6. 在 [範圍名稱] 欄位中，輸入表示特殊管理範圍的名稱。所有認證均將依此範圍處理。
7. 選取 [已啓用] 核取方塊以表示 JMX 連接器中應當使用傳輸層安全性。如果啓用傳輸層安全性，請按照配置管理服務的 JMX 連接器的安全性中的說明填寫 SSL 區段。

變更 J2SE 軟體

Application Server 依賴於 Java 2 Standard Edition (J2SE) 軟體。安裝 Application Server 時，指定 J2SE 軟體的目錄。如需有關變更 J2SE 軟體的說明，請參閱配置 JVM 一般設定。

使用線上說明

管理主控台的線上說明是上下文無關的：按一下右上角的 [說明] 連結時，說明瀏覽器視窗會顯示與目前管理主控台頁面相關的主題。如果目前頁面沒有說明資訊，則顯示 [使用線上說明] 主題。

線上說明包含了上下文有關的概念主題。若要檢視這些主題中的某一個主題，請從說明瀏覽器視窗的目錄中選取該主題。

若要返回到上一說明螢幕，請執行以下步驟：

1. 在說明瀏覽器視窗中，按一下滑鼠右鍵以顯示選取功能表。
2. 選擇 [後退]。

如果沒找到要查找的資訊，請參閱 「Application Server Administration Guide」，可從以下位址獲得：

```
http://docs.sun.com/
```

更多資訊

- Sun Microsystems 全球培訓 — Sun 及其授權中心每年通過基於 Web 的課程和遍佈 60 多個國家 / 地區的 250 多個培訓站點培訓超過 250,000 名學員。如需更多資訊，請參閱：

  `http://training.sun.com/`

- The J2EE 1.4 Tutorial — 該線上教程專爲開發者編寫，其中描述了配置 JMS、設定 JavaMail 資源和管理安全性的管理說明。若要存取此線上教程，請移至此 URL：

  `http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html`

- Application Server Developer's Guide — 該指南包含針對 Application Server 的開發資訊。Developer's Guide 可從以下位址獲得：

  `http://docs.sun.com/`

- `asadmin` 線上說明手冊 — 以 HTML 格式提供，這些線上說明手冊包含所有應用程式伺服器公用程式 (包括 `asadmin` 公用程式指令) 的語法和範例。這些 HTML 頁面發佈在以下 URL：

  `http://docs.sun.com/`

- Application Server 版本說明 — 線上位於：

  `http://docs.sun.com/`

- Getting Started With J2EE Connectors — 該文件含有配置連接器 (資源介面)、連線池和連結器資源的說明。

  `http://java.sun.com/j2ee/connector/`

- docs.sun.com：Sun 產品文件 — 從該站台您可以搜尋和存取我們產品的所有文件：

  `http://docs.sun.com/`

- J2EE 1.4 Documentation 頁面 — 位於我們的公共網站上，該頁面具有指向 J2EE 1.4 平台的技術文件的連結：

  `http://java.sun.com/j2ee/1.4/docs/`

- **快速入門指南** — 該文件說明如何部署和執行簡易 Web 應用程式。該指南位於 *install_dir*`/docs/QuickStart.html` 檔案中。

第 2 章

配置叢集

本章介紹如何使用管理主控台來配置叢集。它包含以下小節：

- 關於叢集
- 用於叢集的管理主控台作業

關於叢集

- 什麼是叢集？
- 叢集類型
- 叢集、實例、負載平衡器和階段作業

什麼是叢集？

叢集是作爲一個邏輯實體一起工作的零個或多個伺服器實例的集合。叢集爲一個或多個 J2EE 應用程式提供了執行階段環境。

在 Sun Java System Application Server Enterprise Edition 8.1 環境中使用叢集具有以下優點：

- 高可用性 (透過允許爲叢集中的伺服器實例提供防故障備用保護來實現)。如果一個伺服器實例當機，其他伺服器實例將接管該伺服器實例正在服務的請求。
- 可延伸性 (透過允許向叢集中新增伺服器實例從而增加系統的容量來實現)。Application Server 的負載平衡器會將請求分布到叢集中的可用伺服器實例。當管理員向叢集中新增更多伺服器實例時，無需中斷服務。

叢集具有以下特性：

- 叢集中的所有實例都參考相同的配置。

- 叢集中的所有實例均具有相同的一組已部署的應用程式 (例如，J2EE 應用程式 EAR 檔案、Web 模組 WAR 檔案或 EJB JAR 檔案)。
- 叢集中的所有實例均具有相同的資源集，從而產生相同的 JNDI 名稱空間。

網域中的每一個叢集都具有唯一的名稱；此外，該名稱在所有節點代理程式名稱、伺服器實例名稱、叢集名稱和配置名稱中也必須是唯一的。此名稱不能爲 `domain`。管理員在叢集上執行的作業與在非叢集伺服器實例上執行之作業相同 (例如，部署應用程式和建立資源)。

有關叢集、節點代理程式和伺服器實例的使用的概述，請參閱「Deployment Planning Guide」。如需有關負載平衡器的詳細資訊，請參閱「配置負載平衡和防故障備用」。

叢集類型

有兩種叢集類型：**獨立叢集**和**共用叢集**。

- 獨立叢集具有自己的配置，它不與其他伺服器實例或叢集共用其配置。依預設，此配置的名稱爲 *cluster_name*`-config`，其中 *cluster_name* 表示叢集的名稱。
- 共用叢集與一個或多個其他叢集或非叢集實例共用其配置。

叢集、實例、負載平衡器和階段作業

叢集、伺服器實例、負載平衡器和階段作業在 Application Server 中的相互關係如下：

- 伺服器實例可以不屬於叢集。但是，不屬於叢集的實例無法通過將階段作業狀態從一個實例傳送到其他實例來利用高可用性。
- 叢集中的伺服器實例可以位於不同的機器上，也可以位於同一機器上。也就是說，可以將不同機器上的伺服器實例群組爲一個叢集。
- 特定負載平衡器可以向多個叢集中的伺服器實例轉寄請求。您可以使用負載平衡器的此功能來執行線上升級，而不使服務受到損失。如需更多資訊，請參閱第 56 頁的「將多個叢集用於線上升級而不使服務受到損失」。
- 單一叢集可以從多個負載平衡器接收請求。如果叢集由多個負載平衡器提供服務，則必須以完全相同的方式在每個負載平衡器上配置叢集。
- 每個階段作業都與特定的叢集連結在一起。這意味著儘管可以將應用程式部署在多個叢集中，但階段作業將只能被防故障備用至同一叢集中的伺服器實例。

- Application Server 支援 HTTP 階段作業和有狀態階段作業 Bean (SFSB) 階段作業的防故障備用。也支援某些儲存在 HTTP 階段作業中的 J2EE 物件參考的防故障備用。

  如需有關 HTTP 階段作業防故障備用的更多資訊，請參閱第 133 頁的「配置可用性和階段作業持續性」

因此，對於叢集中的伺服器實例，叢集充當的是階段作業防故障備用的安全邊界。在 Application Server 中，您可以使用負載平衡器和升級元件，而不使服務受到任何損失。如需更多資訊，請參閱第 56 頁的「將多個叢集用於線上升級而不使服務受到損失」。

用於叢集的管理主控台作業

- 建立叢集
- 配置叢集
- 遷移 EJB 計時器
- 爲叢集建立伺服器實例
- 配置叢集伺服器實例
- 爲叢集配置應用程式
- 爲叢集配置資源
- 刪除叢集
- 將多個叢集用於線上升級而不使服務受到損失

建立叢集

若要建立叢集，請執行以下步驟：

1. 在樹形元件中，選擇 [叢集] 節點。
2. 在 [叢集] 頁面中，按一下 [新建]。將顯示 [建立叢集] 頁面。
3. 在 [名稱] 欄位中，鍵入叢集的名稱。此名稱
   - 只能由大寫字母和小寫字母、數字、底線、連字符和句點 (.) 組成
   - 在所有節點代理程式名稱、伺服器實例名稱、叢集名稱和配置名稱中都必須是唯一的

- 不能爲 domain

4. 在 [配置] 欄位中，從下拉式清單中選擇配置。

   若要建立獨立叢集，請選擇 default-config，並選取標有 [複製選取的配置] 的單選按鈕。預設配置的副本的名稱將爲 *cluster_name*-config。

   若要參考其他配置，請從下拉式清單中選擇配置並選取標有 [參考選取的配置] 的單選按鈕。如果另一個叢集正在使用該配置，則此動作將建立共用叢集。

5. 您可以現在新增伺服器實例，也可以在建立叢集後再新增服務器實例。爲叢集新增伺服器實例之前，請先建立一個或多個節點代理程式或節點代理程式預留位置字元。請參閱第 93 頁的「建立節點代理程式萬用字元」，以取得詳細資訊。

   若要建立伺服器實例，請執行以下步驟：

   a. 在 [要建立的伺服器實例] 區域，按一下 [新增]。

   b. 在 [實例名稱] 欄位中爲實例輸入名稱。

   c. 從 [節點代理程式] 下拉式清單中選擇節點代理程式。

6. 按一下 [確定]。

7. 在顯示的 [已成功建立叢集] 頁面中按一下 [確定]。

如需有關如何管理叢集、伺服器實例和節點代理程式的詳細資訊，請參閱第 87 頁的「部署節點代理程式」。

等效的 asadmin 指令爲：create-cluster

配置叢集

若要配置叢集，請執行以下步驟：

1. 在樹形元件中，展開 [叢集] 節點。
2. 選擇叢集的節點。在 [一般資訊] 頁面中，您可以執行以下作業：
   - 按一下 [啓動實例] 以啓動叢集伺服器實例。
   - 按一下 [停止實例] 以停止叢集伺服器實例。
   - 按一下 [遷移 EJB 計時器] 以將 EJB 計時器從已停止的伺服器實例遷移到叢集中的其他伺服器實例。

等效的 asadmin 指令爲：start-cluster, stop-cluster, migrate-timers

遷移 EJB 計時器

如果伺服器實例非正常或未預期地停止執行，則可能需要將該伺服器實例上安裝的 EJB 計時器移至叢集中正在執行的伺服器實例。若要完成此操作，請執行以下步驟：

1. 從 [源] 下拉式清單中，選擇要遷移的計時器所在的已停止的伺服器實例。
2. 從 [目標] 下拉式清單中，選擇要將計時器遷移到的正在執行的伺服器實例 (此操作可選)。如果將該欄位保留爲空，將隨機選擇一個正在執行的伺服器實例。
3. 按一下 [確定]。
4. 停止並重新啓動目標伺服器實例。

如果源伺服器實例正在執行或目標伺服器實例未執行，將顯示錯誤訊息。

等效的 asadmin 指令爲：`migrate-timers`

為叢集建立伺服器實例

爲叢集建立伺服器實例之前，必須先建立節點代理程式或節點代理程式預留位置字元。請參閱第 93 頁的「建立節點代理程式萬用字元」，以取得詳細資訊。

若要爲叢集建立伺服器實例，請執行以下步驟：

1. 在樹形元件中，展開 [叢集] 節點。
2. 選取叢集的節點。
3. 按一下 [實例] 標籤以顯示 [叢集伺服器實例] 頁面。
4. 按一下 [新建] 以顯示 [建立叢集伺服器實例] 頁面。
5. 在 [名稱] 欄位中，鍵入伺服器實例的名稱。
6. 從 [節點代理程式] 下拉式清單中選擇節點代理程式。
7. 按一下 [確定]。

等效的 asadmin 指令爲：`create-instance`

配置叢集伺服器實例

若要在建立叢集伺服器實例後對其進行變更，請執行以下步驟：

1. 在樹形元件中，展開 [叢集] 節點。
2. 展開包含伺服器實例的叢集的節點，然後選取要進行編輯的伺服器實例節點。
3. 在 [一般資訊] 頁面中，您可以執行以下作業：
    - 按一下 [啓動實例] 以啓動實例。
    - 按一下 [停止實例] 以停止正在執行的實例。
    - 按一下 [JNDI 瀏覽] 以瀏覽正在執行的實例的 JNDI 樹。
    - 按一下 [檢視記錄檔]，以開啓伺服器記錄檢視器。
    - 按一下 [自動重建記錄檔]，以自動重建實例的記錄檔。該動作將排程記錄檔以進行自動重建。實際的自動重建將在下一次向記錄檔寫入項目時發生。
    - 按一下 [恢復作業事件] 以恢復未完成的作業事件。
    - 按一下 [特性] 標籤，以修改實例的連接埠號。
    - 按一下 [監視] 標籤，以變更監視特性。

您也可以在伺服器實例上執行以下作業：

1. 在樹形元件中，展開 [叢集] 節點。
2. 展開包含伺服器實例的叢集的節點。
3. 按一下 [實例] 標籤以移至 [叢集伺服器實例] 頁面。在此頁面中，您可以執行以下作業：
    - 選取實例的核取方塊，並按一下 [刪除]、[啓動] 或 [停止]。
    - 按一下實例的名稱，以顯示 [一般資訊] 頁面。

若要從叢集中刪除叢集伺服器實例，請選取實例名稱旁邊的核取方塊並按一下 [刪除]。

為叢集配置應用程式

若要爲叢集配置應用程式，請執行以下步驟：

1. 在樹形元件中，展開 [叢集] 節點。
2. 選擇叢集的節點。

3. 按一下 [應用程式] 標籤，以顯示 [應用程式] 頁面。在此頁面中，您可以執行以下作業：
   - 選取應用程式旁邊的核取方塊，然後選擇 [啓用] 或 [停用] 以啓用或停用用於叢集的應用程式。
   - 從 [部署] 下拉式清單中，選取要部署的應用程式的類型。在顯示的 [部署] 頁面中，指定應用程式。
   - 從 [篩選器] 下拉式清單中，選取要在清單中顯示的應用程式的類型。

若要編輯應用程式，請按一下應用程式名稱。

為叢集配置資源

若要爲叢集配置資源，請執行以下步驟：

1. 在樹形元件中，展開 [叢集] 節點。
2. 選擇叢集的節點。
3. 按一下 [資源] 標籤，以顯示 [資源] 頁面。在此頁面中，您可以執行以下作業：
   - 選取資源旁邊的核取方塊，然後按一下 [啓用] 或 [停用] 以全域啓用或停用資源。該動作不會移除資源。
   - 從 [新建] 下拉式清單中，選取要建立的資源的類型。建立資源時，請確定將叢集指定爲目標。
   - 從 [篩選器] 下拉式清單中，選取要在清單中顯示的資源的類型。

若要編輯資源，請按一下資源名稱。

刪除叢集

若要刪除叢集，請執行以下步驟：

1. 在樹形元件中，選擇 [叢集] 節點。
2. 在 [叢集] 頁面中，選取要刪除的叢集的名稱旁邊的核取方塊。
3. 按一下 [刪除]。

等效的 `asadmin` 指令爲：`delete-cluster`

將多個叢集用於線上升級而不使服務受到損失

在 Application Server 中，您可以使用負載平衡器和多個叢集來升級元件，而不使服務受到任何損失。例如，元件可以是 JVM、Application Server 或 Web 應用程式。

若要執行此作業，請執行以下操作：

1. 使用叢集的 [一般資訊] 頁面上的 [停止叢集] 按鈕來停止其中一個叢集。
2. 升級該叢集中的元件。
3. 使用叢集的 [一般資訊] 頁面上的 [啓動叢集] 按鈕來啓動叢集。
4. 對其他叢集逐個重複此程序。

由於一個叢集中的階段作業將決不會防故障備用至另一個叢集中的階段作業，因此階段作業在從執行某版本元件的伺服器實例防故障備用至執行其他版本元件的伺服器實例 (位於其他業集中) 時不會出現版本不匹配的情況。這樣，對於叢集中的伺服器實例，叢集充當的是階段作業防故障備用的安全邊界。

備註 在以下情況下不能使用此方法：

- 當您變更高可用性資料庫 (HADB) 的模式時。如需更多資訊，請參閱「Sun Java System Application Server High Availability Administration Guide」中的「Administering High Availability Database」一章。
- 當您執行涉及對應用程式資料庫模式的變更的應用程式升級時。

注意 請同時升級叢集中的所有伺服器實例。否則，可能會出現由從執行不同版本的元件的一個實例到另一個實例的階段作業防故障備用引起的版本不匹配。

第 3 章

配置負載平衡和防故障備用

本章介紹如何設定 Sun Java System Application Server 中的 HTTP 請求的負載平衡。並說明如何配置負載平衡器所控制的伺服器實例之間的防故障備用。此外，還討論 RMI-IIOP 負載平衡和防故障備用。

它包含以下小節：

- 關於 HTTP 負載平衡和防故障備用
- 爲 HTTP 負載平衡配置 Web 伺服器
- HTTP 負載平衡器配置作業
- 升級應用程式
- 關於 RMI-IIOP 負載平衡和防故障備用

關於 HTTP 負載平衡和防故障備用

- HTTP 負載平衡和防故障備用
- HTTP 負載平衡的需求
- 瞭解指定的請求和未指定的請求
- HTTP 負載平衡演算法
- HTTP 負載平衡設定概述

HTTP 負載平衡和防故障備用

負載平衡的目的是在多個 Sun Java System Application Server 實例 (獨立或形成叢集的) 之間平均分布工作量，從而提高系統的整體流量。

使用負載平衡器還可以滿足從一個伺服器實例防故障備用到另一個伺服器實例的請求。對於要持續的 HTTP 階段作業資訊，請配置 HTTP 階段作業持續性。如需更多資訊，請參閱 「配置可用性和階段作業持續性」

使用 asadmin 工具而非管理主控台配置 HTTP 負載平衡。

HTTP 負載平衡的需求

在使用負載平衡器外掛程式處理 HTTP 請求之前，必須滿足以下需求：

- 已安裝 Sun Java System Application Server Enterprise Edition。
- 已安裝並配置 Web 伺服器。

  如需更多資訊，請參閱第 61 頁的 「爲 HTTP 負載平衡配置 Web 伺服器」。
- 已安裝負載平衡器外掛程式。
- 已配置 Application Server。
  - 已建立參與負載平衡的 Application Server 實例或叢集。
  - 應用程式已部署到所有參與負載平衡的 Application Server 實例或叢集中。
  - 參與負載平衡的伺服器實例和叢集必須具有同質環境。通常，這意味著伺服器實例將參考相同的伺服器配置，並且部署到伺服器實例中的應用程式也相同。

瞭解指定的請求和未指定的請求

在請求首次從 HTTP 用戶端傳入負載平衡器時，此請求爲新階段作業的請求。新階段作業的請求稱爲**未指定的**請求。負載平衡器會根據循環演算法將此請求路由到叢集中的應用程式伺服器實例。如需更多資訊，請參閱第 59 頁的 「HTTP 負載平衡演算法」。

在某個應用程式伺服器實例中建立階段作業後，負載平衡器會將此階段作業的所有後續請求都路由到該特定實例而且僅路由到該實例。現有階段作業的請求稱爲**指定的**或**居留式**請求。

HTTP 負載平衡演算法

Sun Java System Application Server 負載平衡器使用居留式循環演算法對進來的 HTTP 和 HTTPS 請求進行負載平衡。給定階段作業的所有請求都將會傳送到同一個應用程式伺服器實例。使用居留式負載平衡器，階段作業資料將快取在單一應用程式伺服器上，而不會分布到叢集中的所有實例。

因此，居留式循環方案能夠帶來顯著的效能優勢，這種優勢通常超過了使用純循環方案所獲得的使負載更加平均分布的優勢。

關於居留式循環負載平衡演算法

當新的 HTTP 請求傳送到負載平衡器外掛程式時，系統將基於簡單的循環方案將該請求轉寄到某個應用程式伺服器實例。隨後，透過使用 Cookie 或明確的 URL 重寫將該請求「居留」在此特定應用程式伺服器實例上。

從居留式資訊中，負載平衡器外掛程式將首先確定請求先前轉寄到的實例。如果發現該實例正常工作，負載平衡器外掛程式會將請求轉寄至該特定應用程式伺服器實例。因此，給定階段作業的所有請求都將會傳送到同一個應用程式伺服器實例。

負載平衡器外掛程式使用以下方法來確定階段作業居留性：

- 基於 Cookie 的方法
- 明確的 URL 重寫方法

基於 *Cookie* 的方法

在基於 Cookie 的方法中，負載平衡器外掛程式使用一個單獨的 Cookie 來記錄路由資訊。

備註	若要使用基於 Cookie 的方法，HTTP 用戶端必須支援 Cookie。

明確的 *URL* 重寫方法

在明確的 URL 重寫方法中，居留式資訊會附加至 URL。即使 HTTP 用戶端不支援 Cookie，也可以使用此方法。

負載平衡和防故障備用範例應用程式

以下目錄包含用於演示負載平衡和防故障備用的範例應用程式：

install_dir/samples/ee-samples/highavailability
install_dir/samples/ee-samples/failover

ee-samples 目錄還包含有關設定環境以執行範例的資訊。

HTTP 負載平衡設定概述

使用 asadmin 工具可以在您的環境中配置負載平衡。請依照下列步驟執行：

1. 完成第 58 頁的 「HTTP 負載平衡的需求」，包括安裝和配置 Web 伺服器和 Application Server 實例和/或叢集。
2. 使用 asadmin 指令 create-http-lb-config 建立負載平衡器配置。
3. 使用 asadmin create-http-lb-ref 爲要管理的負載平衡器增加對叢集和獨立伺服器實例的參考。

   如果您建立了具有目標的負載平衡器配置，並且該目標是負載平衡器參考的唯一叢集或獨立伺服器實例，請跳過此步驟。
4. 使用 asadmin enable-http-lb-server 啓用負載平衡器參考的叢集或獨立伺服器實例。
5. 使用 asadmin enable-http-lb-application 啓用要用於負載平衡的應用程式。

   這些應用程式必須已在負載平衡器所參考的叢集或獨立實例上部署並可以使用。啓用應用程式以用於負載平衡與啓用以使用這些應用程式是兩個獨立的步驟。
6. 使用 asadmin create-health-checker 建立運作狀態檢查程式。

   運作狀態檢查程式監視工作異常的伺服器實例，以便在這些伺服器實例重新正常工作時，負載平衡器可以向它們傳送新的請求。
7. 使用 asadmin export-http-lb-config 產生負載平衡器配置檔案。

   此指令產生要同 Sun Java System Application Server 隨附的負載平衡器外掛程式一起使用的配置檔案。
8. 將負載平衡器配置檔案複製到 Web 伺服器的 config 目錄，該目錄中儲存了負載平衡器外掛程式配置檔案。

為 HTTP 負載平衡配置 Web 伺服器

- 關於 Web 伺服器配置
- 對 Sun Java System Web Server 的修改
- 對 Apache Web Server 的修改
- 對 Microsoft IIS 的修改
- 配置多個 Web 伺服器實例

關於 Web 伺服器配置

負載平衡器外掛程式安裝程式將對 Web 伺服器的配置檔案進行一些修改。所作的變更取決於該 Web 伺服器。

備註 負載平衡器外掛程式可隨 Sun Java System Application Server Enterprise Edition 一起安裝，也可在執行支援的 Web 伺服器的機器上單獨安裝。

如需有關安裝程序的完整詳細資訊，請參閱「Sun Java System Application Server Installation Guide」。

對 Sun Java System Web Server 的修改

安裝程式將對 Sun Java System Web Server 的配置檔案進行以下變更：

1. 將以下負載平衡器外掛程式的特定項目增加到 Web 伺服器實例的 `magnus.conf` 檔案中：

```
##EE lb-plugin
Init fn="load-modules"
shlib="web_server_install_dir/plugins/lbplugin/bin/libpassthrough.so"
funcs="init-passthrough,service-passthrough,name-trans-passthrough"
Thread="no"

Init fn="init-passthrough"

##end addition for EE lb-plugin
```

2. 將特定於負載平衡器外掛程式的以下項目增加到 Web 伺服器實例的 `obj.conf` 檔案中：

```
<Object name=default>

NameTrans fn="name-trans-passthrough" name="lbplugin"
config-file="web_server_install_dir/web_server_instance/config/loadbalancer.xml"

<Object name="lbplugin">
ObjectType fn="force-type" type="magnus-internal/lbplugin"
PathCheck fn="deny-existence" path="*/WEB-INF/*"
Service type="magnus-internal/lbplugin" fn="service-passthrough"
Error reason="Bad Gateway" fn="send-error" uri="$docroot/badgateway.html"
</object>
```

lbplugin 是唯一識別 Object 的名稱；*web_server_install_dir*/*web_server_instance*/config/loadbalancer.xml 是在其上配置並執行負載平衡器的虛擬伺服器的 XML 配置檔案所在的位置。

安裝後，請按照第 60 頁的 「HTTP 負載平衡設定概述」中所述配置負載平衡器。

對 Apache Web Server 的修改

在 Apache 上安裝負載平衡器外掛程式之前，請參閱附錄 A，「編譯和配置 Apache Web Server」中有關編譯和配置 Apache 的資訊。

安裝程式所作的修改

負載平衡器外掛程式安裝程式將所需檔案擷取到 Web 伺服器根目錄下的 libexec (Apache 1.3) 或 modules (Apache 2.0) 資料夾中。它將特定於負載平衡器外掛程式的以下項目增加到 Web 伺服器實例的 httpd.conf 檔案中：

```
<VirtualHost machine_name:443>

##Addition for EE lb-plugin

LoadFile /usr/lib/libCstd.so.1

LoadModule apachelbplugin_module libexec/mod_loadbalancer.so
#AddModule mod_apachelbplugin.cpp
<IfModule mod_apachelbplugin.cpp>
        config-file webserver_instance/conf/loadbalancer.xml
locale en
</IfModule>

<VirtualHost machine_ip_address>
DocumentRoot "webserver_instance/htdocs"
ServerName server_name
</VirtualHost>

##END EE LB Plugin ParametersVersion 7
```

備註

- 在 Apache 1.3 上，當多個 Apache 子程序執行時，每個程序都有自己的負載平衡循環序列。

  例如，如果有兩個正在執行的 Apache 子程序，並且負載平衡外掛程式對兩個應用程式伺服器實例進行負載平衡，則第一個請求將傳送給實例 1，第二個請求也將傳送給實例 1，而第三個請求將傳送給實例 2，第四個請求也將傳送給實例 2。系統將重複執行這種式樣 (實例 1、實例 1、實例 2、實例 2 等)。

  此運作方式可能與您預期的運作方式 (即，實例 1、實例 2、實例 1、實例 2 等) 不同。在 Sun Java System Application Server 中，用於 Apache 的負載平衡外掛程式將為每個 Apache 程序實例創設一個負載平衡器實例，建立獨立的負載平衡序列。

- 如果使用 `--with-mpm=worker` 選項進行編譯，則 Apache 2.0 將具有多重執行緒運作方式。

安裝後的修改

Apache Web 伺服器必須在安全模式中執行以確定它使用負載平衡器外掛程式時可以正常工作。請在 *apache_install_dir* 下建立名爲 `sec_db_files` 的目錄並將 *application_server_domain_dir*`/config/`*security_db_files* 複製到 *apache_install_dir*`/`*sec_db_files*。

對 Microsoft Windows 的附加修改

如果是在 Microsoft Windows 上執行 Apache，則安裝外掛程式後，需要變更某些環境變數：

透過按一下 [開始] -> [設定] -> [主控台] -> [系統] -> [進階] -> [環境變數] -> [系統變數] 將新路徑增加到 Path 環境變數中。編輯 Path 變數使之包含以下路徑：

application_server_install_dir`/bin`

此外，在啓動 Apache Web 伺服器之前，請將環境變數 NSPR_NATIVE_THREADS_ONLY 設定爲 1。

在 [環境變數] 視窗中的 [系統變數] 下，按一下 [新建]。輸入以下名稱和值對：

變數名稱：NSPR_NATIVE_THREADS_ONLY

變數值：1

重新啓動機器。

對 Microsoft IIS 的修改

若要配置 Microsoft 網際網路資訊服務 (IIS) 以使用負載平衡器外掛程式，請修改 Windows 網際網路服務管理程式中的某些特性。網際網路服務管理程式位於 [控制台] 資料夾內的 [管理工具] 資料夾中。

請在安裝 Sun Java System Application Server 後執行這些修改。

1. 開啓網際網路服務管理程式。
2. 選取要爲其啓用外掛程式的網站。此網站通常命名爲「預設網站」。
3. 在網站上按一下滑鼠右鍵，選取 [特性] 以開啓 [特性] 筆記本。
4. 若要增加新的 ISAPI 篩選器，請開啓 [ISAPI 篩選器] 標籤，按一下 [增加]，然後執行以下步驟：
   a. 在 [篩選器名稱] 欄位中，輸入 Application Server
   b. 在 [可執行] 欄位中，鍵入 `C:\Inetpub\wwwroot\sun-passthrough\sun-passthrough.dll`
   c. 按一下 [確定]，關閉 [特性] 筆記本。
5. 建立並配置新的虛擬目錄：
   a. 在預設網站上按一下滑鼠右鍵，選取 [新建]，然後選取 [虛擬目錄]。

      [虛擬目錄建立精靈] 將開啓。
   b. 在 [別名] 欄位中，鍵入 `sun-passthrough`。
   c. 在 [目錄] 欄位中，鍵入 `C:\Inetpub\wwwroot\sun-passthrough`
   d. 核取 [執行許可權] 核取方塊。使與許可權相關的所有其他核取方塊保持未核取狀態。
   e. 按一下 [完成]。
6. 將 `sun-passthrough.dll` 檔案的路徑和 *application_server_install_dir*`/bin` 增加到系統的 `PATH` 環境變數。重新啓動機器。
7. 停止然後啓動 Web 伺服器以使新設定生效。

   若要停止 Web 伺服器，請在此網站上按一下滑鼠右鍵，然後選取 [停止]。若要啓動 Web 伺服器，請在此網站上按一下滑鼠右鍵，然後選取 [啓動]。

   然後，在 Web 瀏覽器中鍵入以下內容，以存取 Web 應用程式環境根：

   `http://`*webserver_name*`/`*web_application*

其中，*webserver_name* 是 Web 伺服器的主機名稱或 IP 位址，*/web_application* 是 `C:\Inetpub\wwwroot\sun-passthrough\sun-passthrough.properties` 檔案中列示的環境根。檢驗 Web 伺服器、負載平衡器外掛程式和 Application Server 是否正常作業。

安裝程式將自動配置 `sun-passthrough.properties` 中的下列特性。您可以變更預設值。

表 3-1 爲 Microsoft IIS 自動配置的 sun-passthrough.properties

特性	定義	預設值
lb-config-file	負載平衡器配置檔案的路徑	*IIS_www_root*\sun-passthrough\loadbalancer.xml
log-file	負載平衡器記錄檔的路徑	*IIS_www_root*\sun-passthrough\lb.log
log-level	Web 伺服器的記錄層級	資訊

配置多個 Web 伺服器實例

Sun Java System Application Server 安裝程式不允許在單一機器上安裝多個負載平衡器外掛程式。若要在單一叢集或多個叢集中的單一機器上安裝多個帶有負載平衡器外掛程式的 Web 伺服器，需要手動執行一些步驟來配置負載平衡器外掛程式。

1. 配置新的 Web 伺服器實例以使用負載平衡器外掛程式 (如第 61 頁的 「對 Sun Java System Web Server 的修改」、第 62 頁的 「對 Apache Web Server 的修改」或第 64 頁的 「對 Microsoft IIS 的修改」所述)。
2. 將 `sun-loadbalancer_1_1.dtd` 檔案從現有 Web 伺服器實例的 `config` 目錄複製到新實例的 `config` 目錄。
3. 若要使用相同的負載平衡器配置，請將 `loadbalancer.xml` 檔案從現有 Web 伺服器實例的 `config` 目錄複製到新實例的 `config` 目錄。
4. 若要使用其他負載平衡器配置，請執行以下步驟：
   a. 使用 `asadmin create-http-lb-config` 建立新的負載平衡器配置。
   b. 使用 `asadmin export http-lb-config` 將新配置匯出到 `loadbalancer.xml` 檔案中。
   c. 將該 `loadbalancer.xml` 檔案複製到新 Web 伺服器的 `config` 目錄。

   如需有關建立負載平衡器配置並將其匯出到 `loadbalancer.xml` 檔案的資訊，請參閱 「HTTP 負載平衡器配置作業」。

HTTP 負載平衡器配置作業

- 建立 HTTP 負載平衡器配置
- 建立 HTTP 負載平衡器參考
- 啓用用於負載平衡的伺服器實例
- 啓用用於負載平衡的應用程式
- 建立 HTTP 運作狀態檢查程式
- 匯出負載平衡器配置檔案
- 變更 HTTP 負載平衡器配置
- 啓用動態重新配置
- 停用 (靜止) 伺服器實例或叢集
- 停用 (靜止) 應用程式
- 配置 HTTP 和 HTTPS 階段作業防故障備用
- 配置等冪 URL
- 配置 HTML 錯誤頁面

建立 HTTP 負載平衡器配置

負載平衡器配置是定義負載平衡器的 `domain.xml` 檔案中的命名配置。

負載平衡配置非常靈活：

- 儘管每個負載平衡器只有一個負載平衡器配置，但每個負載平衡器配置可以關聯多個負載平衡器。
- 儘管一個網域可以關聯多個負載平衡器，但一個負載平衡器只爲一個網域服務。

使用 `asadmin` 指令 `create-http-lb-config` 建立配置。指定下列參數：

- response timeout

  伺服器實例必須傳回回應的時間段 (以秒爲單位)。如果在該時間段內未收到任何回應，則認爲伺服器處於異常狀態。預設值爲 60。

- HTTPS routing

  指定對負載平衡器的 HTTPS 請求是否生成對伺服器實例的 HTTPS 或 HTTP 請求。

如需更多資訊，請參閱第 72 頁的 「配置 HTTP 和 HTTPS 階段作業防故障備用」。

- reload interval

  對負載平衡器配置檔案 `loadbalancer.xml` 所作的變更進行檢查的時間間隔。當檢查偵測到變更時，系統將重新載入配置檔案。0 值停用重新載入。

  如需更多資訊，請參閱第 71 頁的 「啓用動態重新配置」。

- monitor

  指定是否對負載平衡器啓用監視功能。

  如需更多資訊，請參閱第 75 頁的 「監視 HTTP 負載平衡器外掛程式」。

- routecookie

  指定負載平衡器外掛程式用於記錄路由資訊的 Cookie 的名稱。HTTP 用戶端必須支援 Cookie。如果您的瀏覽器設定爲在儲存 Cookie 前詢問，則 Cookie 的名稱爲 JROUTE。

- target

  爲負載平衡器配置指定目標。如果指定目標，則相當於增加對該目標的參考。目標可爲叢集或獨立實例。

如需更多資訊，請參閱 `create-http-lb-config`、`delete-http-lb-config` 和 `list-http-lb-configs` 的文件。

建立 HTTP 負載平衡器參考

當您在負載平衡器中建立對獨立伺服器或叢集的參考時，會將該伺服器或叢集增加到負載平衡器控制的目標伺服器和叢集的清單中。仍需要先啓用所參考的伺服器或叢集 (使用 `enable-http-lb-server`)，然後才能對向該伺服器或叢集發出的請求執行負載平衡。如果建立了帶有目標的負載平衡器配置，則系統已增加了作爲參考的該目標。

使用 `create-http-lb-ref` 建立參考。您必須提供負載平衡器配置名稱和目標伺服器實例或叢集。

若要刪除參考，請使用 `delete-http-lb-ref`。必須先使用 `disable-http-lb-server` 停用已參考的伺服器或叢集，才能刪除此參考。

如需更多資訊，請參閱 `create-http-lb-ref` 和 `delete-http-lb-ref` 的文件。

啟用用於負載平衡的伺服器實例

建立對伺服器實例或叢集的參考後，請使用 `enable-http-lb-server` 啓用伺服器實例或叢集。如果在建立負載平衡器配置時使用某伺服器實例或叢集作爲目標，則必須啓用該伺服器實例或叢集。

如需更多資訊，請參閱 `enable-http-lb-server` 的文件。

啟用用於負載平衡的應用程式

負載平衡器管理的所有伺服器都必須具有同質配置，包括部署到這些伺服器的相同應用程式集。部署某個應用程式並啓用 (在部署期間或之後) 此應用程式以進行存取後，您必須啓用該應用程式以進行負載平衡。如果沒有爲負載平衡啓用應用程式，則將無法對該應用程式的請求執行負載平衡和防故障備用，即使已對該應用程式部署到的伺服器的請求執行了負載平衡和防故障備用。

啓用應用程式時，請指定應用程式名稱和目標。如果負載平衡器管理了多個目標 (例如，兩個叢集)，請在所有目標上啓用該應用程式。

如需更多資訊，請參閱 `enable-http-lb-application` 的線上說明。

如果部署了新的應用程式，您也必須啓用該應用程式以進行負載平衡並再次匯出負載平衡器配置。

建立 HTTP 運作狀態檢查程式

負載平衡器的運作狀態檢查程式將定期檢查所有標示爲異常的已配置 Application Server 實例。運作狀態檢查程式不是必需的，但如果沒有運作狀態檢查程式，或者停用了運作狀態檢查程式，異常實例的定期運行狀況檢查就不會執行。

負載平衡器的運作狀態檢查機制使用 HTTP 與應用程式伺服器實例進行通訊。運作狀態檢查程式將 HTTP 請求傳送給指定的 URL 並等待回應。HTTP 回應標頭中的狀態碼在 100 到 500 之間時表示實例處於正常狀態。

建立運作狀態檢查程式

若要建立運作狀態檢查程式，請使用 `asadmin` 的 `create-http-health-checker` 指令。指定下列參數：

- url

  指定負載平衡器檢查的偵聽程式的 URL 以確定其運作狀態。預設值爲 「/」。

- interval

  指定實例的運作狀態檢查發生的間隔 (以秒爲單位)。其預設值爲 30 秒。指定爲 0 將停用運作狀態檢查程式。

- timeout

  指定逾時間隔 (以秒爲單位)，必須在該時間間隔內獲得回應才能認爲偵聽程式運作正常。預設值爲 10 秒。

如果應用程式伺服器實例標示爲異常，運作狀態檢查程式將輪詢異常實例以確定實例的狀態是否已變爲正常。運作狀態檢查程式使用指定的 URL 來檢查所有異常的應用程式伺服器實例，以確定這些異常的應用程式伺服器實例是否已返回到正常狀態。

如果運作狀態檢查程式發現某個異常實例已變爲正常，該實例將被增加到正常實例的清單中。

如需更多資訊，請參閱 `create-http-health-checker` 和 `delete-http-health-checker` 的文件。

正常實例的附加運作狀態檢查特性

`create-http-health-checker` 建立的運作狀態檢查程式僅檢查異常實例。若要定期檢查正常實例，請在匯出的 `loadbalancer.xml` 檔案中設定某些附加特性。

備註 只能在匯出 `loadbalancer.xml` **之後**對該檔案進行手動編輯來設定這些特性。沒有等效的 asadmin 指令可以使用。

若要檢查正常的實例，請設定以下特性：

表 3-2 運作狀態檢查程式特性

特性	定義
active-healthcheck-enabled	True/False 標幟，用於表示是否要 Ping 正常伺服器實例以確定這些實例是否正常。若要 Ping 伺服器實例，請將標幟設定為 True。
number-healthcheck-retries	指定在將未回應的伺服器實例標示為異常之前，負載平衡器的運作狀態檢查程式 Ping 該伺服器實例的次數。有效範圍在 1 到 1000 之間。設定的預設值為 3。

編輯 loadbalancer.xml 檔案來設定特性。例如：

```
<property name="active-healthcheck-enabled" value="true"/>
<property name="number-healthcheck-retries" value="3"/>
```

如果增加了這些特性，隨後編輯並再次匯出了 loadbalancer.xml 檔案，您必須將這些特性再次增加到該檔案中，因爲新匯出的配置不會包含這些特性。

匯出負載平衡器配置檔案

Sun Java System Application Server 隨附的負載平衡外掛程式使用名爲 loadbalancer.xml 的配置檔案。使用 asadmin 工具可以在 domain.xml 檔案中建立負載平衡器配置。配置負載平衡環境後，請將其匯出至檔案：

1. 使用 asadmin 指令 export-http-lb-config 匯出 loadbalancer.xml 檔案。

   匯出用於特定負載平衡器配置的 loadbalancer.xml 檔案。您可以指定路徑和其他檔案名稱。如果不指定檔案名稱，則該檔案命名爲 loadbalancer.xml.*load_balancer_config_name*。如果不指定路徑，將在 *application_server_install_dir*/domains/*domain_name*/generated 目錄中建立該檔案。

   若要在 Windows 上指定路徑，請將路徑加上引號。例如，"c:\sun\AppServer\loadbalancer.xml"。

2. 將已匯出的負載平衡器配置檔案複製到 Web 伺服器的配置目錄。

   例如，對於 Sun Java System Web Server，該位置可能爲 *web_server_root*/config。

   Web 伺服器配置目錄中的負載平衡器配置檔案必須命名爲 loadbalancer.xml。如果您的檔案使用其他名稱 (例如 loadbalancer.xml.*load_balancer_config_name*)，則必須重新命名它。

變更 HTTP 負載平衡器配置

如果您透過建立或刪除對伺服器的參考、部署新的應用程式、啓用或停用伺服器或應用程式等方法來變更 HTTP 負載平衡器配置，請再次匯出負載平衡器配置檔案並將該檔案複製到 Web 伺服器的 config 目錄。如需更多資訊，請參閱第 70 頁的「匯出負載平衡器配置檔案」。

負載平衡器外掛程式將基於在負載平衡器配置中指定的重新載入間隔定期檢查已更新的配置。在指定的時間值後，如果負載平衡器發現新的配置檔案，則它將開始使用新配置。

啟用動態重新配置

啓用動態重新配置後，負載平衡器外掛程式將定期檢查已更新的配置。若要啓用動態重新配置，請執行以下步驟：

- 若要在建立負載平衡器配置時啓用動態重新配置，請在執行 `asadmin create-http-lb-config` 時使用 `--reloadinterval` 選項。

  此選項用於設定對負載平衡器配置檔案 `loadbalancer.xml` 所作的變更進行檢查的時間間隔。0 值停用重新載入。依預設，將啓用動態重新載入，並且時間間隔設定爲 60 秒。

- 若要在停用動態重新載入後重新啓用它，或變更重新載入間隔，請使用 `asadmin set` 指令。

在變更這些設定後，請再次匯出負載平衡器配置檔案並將其複製到 Web 伺服器的 `config` 目錄。

如果啓用了先前已停用的動態重新配置，您還必須重新啓動 Web 伺服器。

備註

- 如果嘗試進行自身重新配置時，負載平衡器遇到了硬碟讀取錯誤，它將使用記憶體中的目前配置。負載平衡器還確定了在覆寫現有配置之前，已修改的配置資料與 DTD 相容。

  遇到磁碟讀取錯誤後，相關的警告訊息將記錄到 Web 伺服器的錯誤記錄檔中。

  Sun Java System Web Server 的錯誤記錄位於以下位置：*web_server_install_dir*`/`*webserver_instance*`/logs/`。

停用（靜止）伺服器實例或叢集

在因任何原因停止應用程式伺服器之前，您希望實例完成正在處理的請求。正常停用伺服器實例或叢集的程序被稱爲靜止。

負載平衡器使用以下策略來靜止應用程式伺服器實例：

- 如果已停用某個實例 (獨立實例或叢集的一部分)，並且逾時尚未到期，居留式請求將繼續發送到該實例。但是，新請求將不會傳送到已停用的實例。
- 逾時到期後，該實例將停用。從負載平衡器到該實例的所有開啓的連線將關閉。負載平衡器不會將任何請求傳送到該實例，即使居留在該實例的所有階段作業都還有效。相反，負載平衡器會將居留式請求防故障備用到另一個正常實例上。

若要停用某個伺服器實例或叢集，請執行以下步驟：

1. 執行 `asadmin disable-http-lb-server`，設定逾時 (以分鐘爲單位)。

2. 使用 `asadmin export-http-lb-config` 匯出負載平衡器配置檔案。
3. 將匯出的配置複製到 Web 伺服器的 `config` 目錄。
4. 停止該伺服器實例或叢集。

停用 (靜止) 應用程式

在取消部署 Web 應用程式之前，您希望該應用程式完成正在處理的請求。正常停用應用程式的程序稱爲靜止。

負載平衡器使用以下策略來靜止應用程式：

- 如果停用了某個應用程式，並且逾時尚未到期，負載平衡器將不會轉寄對已停用的應用程式的新請求。這些請求將傳回 Web 伺服器。居留式請求將會繼續轉寄，直至逾時到期。
- 逾時到期後，該應用程式將停用。負載平衡器將不接受對該應用程式的任何請求 (包括居留式請求)。

當您從負載平衡器參考的每個伺服器實例或叢集中停用應用程式時，已停用的應用程式的使用者將遭受服務損失，直到再次啓用該應用程式。

如果您從一個伺服器實例或叢集中停用應用程式而使該應用程式在其他伺服器實例或叢集中保持啓用狀態，則使用者仍可存取該應用程式。

若要停用應用程式，請執行以下步驟：

1. 執行 `asadmin disable-http-lb-application`，指定逾時 (以分鐘爲單位)、要停用的應用程式的名稱和停用該應用程式的目標叢集或實例。
2. 使用 `asadmin export-http-lb-config` 匯出負載平衡器配置檔案。
3. 將匯出的配置複製到 Web 伺服器的 `config` 目錄。

配置 HTTP 和 HTTPS 階段作業防故障備用

如果 HTTP/HTTPS 階段作業所連線的原始應用程式伺服器實例變爲不可用，負載平衡器外掛程式會將這些階段作業防故障備用到其他應用程式伺服器實例上。本小節介紹如何配置負載平衡器外掛程式以啓用 HTTP/HTTPS 路由和階段作業防故障備用。

如需有關配置 HTTP 階段作業持續性的資訊，請參閱「配置可用性和階段作業持續性」。

本小節論述以下主題：

- 關於 HTTPS 路由
- 配置 HTTPS 路由
- 有關負載平衡 HTTP/HTTPS 請求的已知問題

關於 HTTPS 路由

所有進來的請求 (無論是 HTTP 請求還是 HTTPS 請求) 都是透過負載平衡器外掛程式路由到應用程式伺服器實例。但是，如果啓用了 HTTPS 路由，則負載平衡器外掛程式將把 HTTPS 請求僅轉寄給使用 HTTPS 連接埠的應用程式伺服器。請註意，HTTPS 路由是針對新請求和居留式請求而執行的。

如果收到了 HTTPS 請求且沒有正在進行的階段作業，負載平衡器外掛程式將選取使用已配置的 HTTPS 連接埠的可用應用程式伺服器實例，並將請求轉寄到該實例。

在正在進行的 HTTP 階段作業中，如果收到對同一個階段作業的新 HTTPS 請求，則使用在 HTTP 階段作業期間儲存的階段作業和居留式資訊來路由 HTTPS 請求。新的 HTTPS 請求將路由到在此 HTTPS 連接埠上處理上一個 HTTP 請求的同一伺服器。

配置 HTTPS 路由

`create-http-lb-config` 指令的 `httpsrouting` 選項用於控制是爲正在參與負載平衡的所有應用程式伺服器開啓還是關閉 HTTPS 路由。如果此選項設定爲 false，所有 HTTP 和 HTTPS 請求都將作爲 HTTP 請求進行轉寄。建立新的負載平衡器配置時，請將此選項設定爲 true，或者以後使用 `asadmin set` 指令變更此選項的設定。

備註

- 若要使 HTTPS 路由工作，必須配置一個或多個 HTTPS 偵聽程式。
- 如果 `https-routing` 設定爲 `true`，而新請求或居留式請求傳入的叢集中沒有正常的 HTTPS 偵聽程式，則該請求將產生一個錯誤。

有關負載平衡 HTTP/HTTPS 請求的已知問題

以下內容討論有關負載平衡器對 HTTP/HTTPS 請求處理的限制。

- 如果某個階段作業使用 HTTP 和 HTTPS 請求的組合，則第一個請求必須是 HTTP 請求。如果第一個請求是 HTTPS 請求，則其後不能接 HTTP 請求。這是因爲與 HTTPS 階段作業關聯的 Cookie 不是由瀏覽器傳回的。瀏覽器將兩個不同的協定解釋爲兩個不同的伺服器，並啓動新的階段作業。

  僅當 `httpsrouting` 設定爲 `true` 時，此限制才有效。

- 如果某個階段作業具有 HTTP 和 HTTPS 請求的組合，則必須將應用程式伺服器實例配置爲同時具有 HTTP 和 HTTPS 偵聽程式。

  僅當 `httpsrouting` 設定爲 `true` 時，此限制才有效。

- 如果某個階段作業具有 HTTP 和 HTTPS 請求的組合，則必須將應用程式伺服器實例配置爲具有使用標準連接埠號 (即，對於 HTTP 爲 80，對於 HTTPS 爲 443) 的 HTTP 和 HTTPS 偵聽程式。

  無論爲 `httpsrouting` 設定了何值，此限制都有效。

配置等冪 URL

若要增強已部署應用程式的可用性，請在負載平衡器所服務的所有應用程式伺服器實例上將環境配置爲重試失敗的等冪 HTTP 請求。此選項用於唯讀請求 (例如，重試搜尋請求)。

等冪請求是一種在重試時不會在應用程式中造成任何變更或不一致的請求。在 HTTP 中，某些方法 (例如 GET) 是等冪的，而其他方法 (例如 POST) 則不是。重試等冪 URL 不會導致伺服器上或資料庫中的值發生變更。唯一的差異在於使用者收到的回應會有所不同。

等冪請求的範例包括搜尋引擎查詢和資料庫查詢。基礎原則是重試不會導致資料的更新或修改。

請在 `sun-web.xml` 檔案中配置等冪 URL。當您匯出負載平衡器配置時，等冪 URL 資訊將自動增加到 `loadbalancer.xml` 檔案中。

如需有關配置等冪 URL 的更多資訊，請參閱 「Developer's Guide」。

配置 HTML 錯誤頁面

您可以將自己的錯誤頁面或錯誤頁面的 URL 指定爲顯示給一般使用者。指定錯誤頁面將會置換所有其他爲錯誤報告配置的機制。

請在 sun-web.xml 檔案中配置 HTML 錯誤頁面。在匯出負載平衡器配置時，HTML 錯誤頁面資訊將自動從 sun-web.xml 檔案增加到 loadbalancer.xml 檔案中。

如需有關配置 HTML 錯誤頁面的更多資訊，請參閱「Developer's Guide」。

監視 HTTP 負載平衡器外掛程式

- 配置記錄訊息
- 記錄訊息類型
- 配置監視
- 瞭解監視訊息

配置記錄訊息

負載平衡器外掛程式使用 Web 伺服器的記錄機制寫入記錄訊息。Application Server 上的預設記錄層級設定爲 Sun Java System Web Server (INFO)、Apache Web Server (WARN) 和 Microsoft IIS (INFO) 上的預設記錄層級。應用程式伺服器記錄層級 (FINE、FINER 和 FINEST) 對映到 Web 伺服器上的 DEBUG 層級。

這些記錄訊息將寫入 Web 伺服器記錄檔，並且以可使用程序檔進行剖析或匯入試算表以計算所需公制的原始資料形式進行。

記錄訊息類型

負載平衡器外掛程式產生以下三組不同的記錄訊息：

- 負載平衡器配置器記錄訊息
- 請求派送和執行階段記錄訊息
- 配置器錯誤訊息

負載平衡器配置器記錄訊息

使用等冪 URL 和錯誤頁面設定時，將記錄這些訊息。

等冪 URL 式樣配置的輸出包含以下資訊：

- 當記錄層級設定爲 FINE 時：

```
CONFxxxx:IdempotentUrlPattern configured <url-pattern>
<no-of-retries> for web-module :<web-module>
```

- 當記錄層級設定爲 SEVERE 時：

  ```
  CONFxxxx:Duplicate entry of Idempotent URL element <url-pattern> for
  webModule <web-module> in loadbalancer.xml."
  ```

- 當記錄層級設定爲 WARN 時：

  ```
  CONFxxxx:Invalid IdempotentUrlPatternData <url-pattern> for
  web-module <web-module>
  ```

錯誤頁面 URL 配置的輸出包含以下資訊 (記錄層級設定爲 WARN)：

```
CONFxxxx:Invalid error-url for web-module <web-module>
```

請求派送和執行階段記錄訊息

這些記錄訊息在負載平衡和派送請求時產生。

- 每個方法開始的標準記錄的輸出均包含以下資訊 (記錄層級設定爲 FINE)：

  ```
  ROUTxxxx:Executing Router method <method_name>
  ```

- 每個方法開始的路由器記錄的輸出均包含以下資訊 (記錄層級設定爲 INFO)：

  ```
  ROUTxxxx:Successfully Selected another ServerInstance for idempotent
  request <Request-URL>
  ```

- 執行階段記錄的輸出包含以下資訊 (記錄層級設定爲 INFO)：

  ```
  RNTMxxxx:Retrying Idempotent <GET/POST/HEAD> Request <Request-URL>
  ```

配置器錯誤訊息

如果存在配置問題 (例如，缺少參考的自訂錯誤頁面)，將顯示這些錯誤。

- 記錄層級設定爲 INFO：

  ROUTxxxx: 無法重試非等冪請求 <Request-URL>

  範例：`ROUTxxxx:Non Idempotent Request http://sun.com/addToDB?x=11&abc=2 cannot be retried`

- 記錄層級設定爲 FINE：

  RNTMxxxx:Invalid / Missing Custom error-url / page:<error-url> 用於 Web 模組：<web-module>

範例：RNTMxxxx:Invalid / Missing Custom error-url / page:myerror1xyz for web-module:test

配置監視

執行以下步驟可以開啓負載平衡器外掛程式記錄訊息：

1. 設定 Web 伺服器中的記錄選項。
   a. 從 Sun Java System Web Server 的管理主控台中，移至 [Magnus 編輯器] 標籤。

      將 [log verbose] 選項設定爲 On。

   b. 對於 Apache Web Server，請將記錄層級設定爲 DEBUG。

   c. 對於 Microsoft IIS，請在 sun-passthrough.properties 檔案中將記錄層級設定爲 FINE。
2. 將負載平衡器配置的 [monitor] 選項設定爲 true。

   使用 asadmin create-http-lb-config 指令可以在最初建立負載平衡器配置時將監視設定爲 true，也可以稍後使用 asadmin set 指令將其設定爲 true。依預設，停用監視。

負載平衡器外掛程式記錄以下資訊：

- 每個請求的請求開始/停止資訊。
- 請求從異常實例防故障備用到正常實例時的防故障備用請求資訊。
- 每個運作狀態檢查週期結束時的異常實例清單。

備註 在負載平衡器外掛程式上啓用記錄後，如果將 Web 伺服器記錄層級設定爲 DEBUG 或設定爲列印詳細訊息，負載平衡器會將 HTTP 階段作業 ID 寫入 Web 伺服器記錄檔中。因此，如果託管負載平衡器外掛程式的 Web 伺服器位於 DMZ 中，請勿在生產環境中使用 DEBUG 或類似的記錄層級。

如果必須使用 DEBUG 記錄層級，請在 loadbalancer.xml 中將 require-monitor-data 特性設定爲 false，以關閉負載平衡器記錄。

瞭解監視訊息

負載平衡器外掛程式記錄訊息的格式如下所示：

- HTTP 請求的開頭處包含以下資訊：

  ```
  RequestStart Sticky(New) <req-id> <time-stamp> <URL>
  ```

  時間戳記值是從 1970 年 1 月 1 日開始的毫秒數。例如：

  ```
  RequestStart New 123456 602983
  http://austen.sun.com/Webapps-simple/servlet/Example1
  ```

- HTTP 請求的結尾處包含 `RequestExit` 訊息，如下所示：

  ```
  RequestExit Sticky(New) <req-id> <time-stamp> <URL> <listener-id>
  <response-time> Failure-<reason for error>(incase of a failure)
  ```

  例如：

  ```
  RequestExit New 123456 603001
  http://austen.sun.com/Webapps-simple/servlet/Example1
  http://austen:2222 18
  ```

  備註 在 RequestExit 訊息中，<response-time> 表示從負載平衡器外掛程式方面，返回請求的總時間 (以毫秒為單位)。

- 異常實例清單，如下所示：

  ```
  UnhealthyInstances <cluster-id> <time-stamp> <listener-id>,
  <listener-id>...
  ```

  例如：

  ```
  UnhealthyInstances cluster1 701923 http://austen:2210, http://austen:3010
  ```

- 防故障備用請求清單，如下所示：

  ```
  FailedoverRequest <req-id> <time-stamp> <URL> <session-id>
  <failed-over-listener-id> <unhealthy-listener-id>
  ```

  例如：

  ```
  FailedoverRequest 239496 705623
  http://austen.sun.com/Apps/servlet/SessionTest 16dfdac3c7e80a40
  http://austen:4044 http://austen:4045
  ```

升級應用程式

- 關於捲動升級
- 在單一獨立叢集中進行升級
- 在兩個叢集中進行升級

關於捲動升級

若要升級應用程式而不使使用者遭受任何服務損失，請每次在一個伺服器或叢集上升級應用程式。叢集透明地維護一個混合版本環境，使用者不會察覺到正在進行升級。這種類型的升級稱爲捲動升級。

只有在舊版本和新版本的應用程式相容並且能夠同時執行的情況下，才能執行捲動升級。階段作業資訊必須可以相容。可以在單一獨立叢集或在多個叢集中執行混合模式的捲動升級。

如果應用程式發生重大變更 (例如，對資料庫模式的變更)，則無法執行混合模式環境中的捲動升級。在這種情況下，升級時請關閉此應用程式。取消部署此應用程式，然後使用相同的名稱重新部署已升級的應用程式。

在單一獨立叢集中進行升級

若要在單一獨立叢集 (即不與任何其他叢集共用配置的叢集) 中升級應用程式，請執行以下步驟：

1. 儲存舊版本的應用程式或備份網域。

   若要備份網域，請使用 `asadmin backup-domain` 指令。

2. 關閉叢集的動態重新配置 (如果已啓用)。

   透過管理主控台：

   a. 展開 [配置] 節點。

   b. 按一下叢集配置的名稱。

   c. 在 [配置系統特性] 頁面中，取消核取 [已啓用動態重新配置] 方塊。

   d. 按一下 [儲存]

   等效的 `asadmin` 指令爲 `asadmin set`。語法爲：

   ```
   asadmin set --user user --passwordfile password_file
   cluster_name-config.dynamic-reconfiguration-enabled=false
   ```

3. 將已升級的應用程式重新部署到目標 domain 中。如果使用管理主控台進行重新部署，網域將自動成爲目標。由於已停用動態重新配置，因此舊應用程式將繼續在叢集上執行。
4. 使用 asadmin enable-http-lb-application 爲實例啓用已重新部署的應用程式。
5. 使用 asadmin disable-http-lb-server 停用一個伺服器實例。
6. 使用 asadmin export-http-lb-config 匯出負載平衡器配置檔案。
7. 將已匯出的配置檔案複製到 Web 伺服器實例的配置目錄。例如，對於 Sun Java System Web Server，其位置爲 *web_server_install_dir*/https-*host-name*/config/loadbalancer.xml
8. 請等待，直至逾時值到期。監視負載平衡器的記錄檔以確定實例已離線。
9. 在叢集中的其他實例仍處於執行狀態的情況下，重新啓動已停用的伺服器實例。重新啓動操作將使伺服器與網域同步，並更新應用程式。
10. 測試重新啓動的伺服器上的應用程式，以確定應用程式執行正常。
11. 使用 asadmin enable-http-lb-server 啓用伺服器實例。
12. 使用 asadmin export-http-lb-config 匯出負載平衡器配置檔案。
13. 將配置檔案複製到 Web 伺服器的配置目錄。
14. 對叢集中的每個實例重複步驟 5 至步驟 13。
15. 當所有伺服器實例都包含新的應用程式並且處於執行狀態時，請再次爲叢集啓用動態重新配置。

在兩個叢集中進行升級

1. 儲存舊版本的應用程式或備份網域。

   若要備份網域，請使用 asadmin backup-domain 指令。
2. 關閉兩個叢集的動態重新配置 (如果已啓用)。

   透過管理主控台：

   a. 展開 [配置] 節點。
   b. 按一下一個叢集配置的名稱。
   c. 在 [配置系統特性] 頁面中，取消核取 [已啓用動態重新配置] 方塊。
   d. 按一下 [儲存]

e. 對第二個叢集重複此程序。

等效的 `asadmin` 指令爲 `asadmin set`。語法爲：

`asadmin set --user` *user* `--passwordfile` *password_file* *cluster_name*`-config.dynamic-reconfiguration-enabled=false`

3. 將已升級的應用程式重新部署到目標 `domain` 中。如果使用管理主控台進行重新部署，網域將自動成爲目標。由於已停用動態重新配置，因此舊應用程式將繼續在叢集上執行。
4. 使用 `asadmin enable-http-lb-application` 爲叢集啓用已重新部署的應用程式。
5. 使用 `asadmin disable-http-lb-server` 從負載平衡器上停用其中一個叢集。
6. 使用 `asadmin export-http-lb-config` 匯出負載平衡器配置檔案。
7. 將已匯出的配置檔案複製到 Web 伺服器實例的配置目錄。例如，對於 Sun Java System Web Server，其位置爲 *web_server_install_dir*`/https-`*host-name*`/config/loadbalancer.xml`
8. 請等待，直至逾時值到期。監視負載平衡器的記錄檔以確定叢集已離線。
9. 在另一個叢集仍處於執行狀態的情況下，重新啓動已停用的叢集。重新啓動操作將使叢集與網域同步，並更新應用程式。
10. 測試重新啓動的叢集上的應用程式，以確定應用程式執行正常。
11. 使用 `asadmin enable-http-lb-server` 啓用叢集。
12. 使用 `asadmin export-http-lb-config` 匯出負載平衡器配置檔案。
13. 將配置檔案複製 Web 伺服器的配置目錄。
14. 對另一個叢集重複步驟 5 至步驟 13。
15. 當所有伺服器實例都包含新的應用程式並且處於執行狀態時，請爲兩個叢集再次啓用動態重新配置。

關於 RMI-IIOP 負載平衡和防故障備用

- RMI-IIOP 負載平衡和防故障備用的需求
- RMI-IIOP 負載平衡和防故障備用演算法

RMI-IIOP 負載平衡和防故障備用的需求

Sun Java™ System Application Server 透過 RMI-IIOP 提供遠端 EJB 參考和名稱服務物件的高可用性。在使用這些功能之前，環境必須滿足以下需求：

- Sun Java System 已安裝 Application Server Enterprise Edition。
- 存在一個至少包括兩個應用程式伺服器實例的叢集。

  如需有關叢集的更多資訊，請參閱 「配置叢集」
- J2EE 應用程式已部署到所有將參與負載平衡的應用程式伺服器實例和叢集。
- 已啓用 RMI-IIOP 用戶端應用程式以用於負載平衡。

  系統支援以下兩種 RMI/IIOP 用戶端的負載平衡：

  - 在應用程式用戶端容器 (ACC) 中執行且存取部署在 Application Server 實例上的 EJB 的 Java 應用程式。
  - 不在 ACC 中執行、但存取部署在 Application Server 實例上的 EJB 的 Java 應用程式。

  啓用基於 RMI-IIOP 的應用程式所需的配置設定取決於用戶端的類型。如需有關配置 RMI-IIOP 用戶端應用程式以進行負載平衡的更多資訊，請參閱 「Sun Java System Application Server Developer's Guide」。

  如需有關 RMI-IIOP 防故障備用和負載平衡的附加資訊，請參閱 「Sun Java System Application Server High Availability Administration Guide」。

備註 不支援透過 SSL 進行 RMI-IIOP 防故障備用。

RMI-IIOP 負載平衡和防故障備用演算法

Sun Java System Application Server 採用了隨機演算法和循環演算法，用於 RMI-IIOP 路徑上的遠端 EJB 參考和名稱服務物件的負載平衡。

在 RMI-IIOP 用戶端首次建立新的 `InitialContext` 物件時，可用的 Application Server IIOP 終點的清單對於該用戶端是隨機的。對於該 `InitialContext` 物件，負載平衡器會將查找請求和其他 `InitialContext` 作業導向至清單中的第一個終點。如果第一個終點不可用，則使用清單中的第二個終點，依此類推。

隨後每次用戶端建立新的 `InitialContext` 物件時，將自動重建終點清單，從而將不同的 IIOP 終點用於 `InitialContext` 作業。

在您從透過 `InitialContext` 物件獲得的參考中獲取或建立 Bean 時，將在服務於指定給 `InitialContext` 物件的 IIOP 終點的 Application Server 實例上建立這些 Bean。對這些 Bean 的參考包括叢集中的所有 Application Server 實例的 IIOP 終點位址。

主終點是與用於查找或建立 Bean 的 `InitialContext` 終點相對應的 Bean 終點。叢集中的其他 IIOP 終點將指定爲**替代終點**。如果 Bean 的主終點變得不可用，則該 Bean 上的其他請求將防故障備用到其中一個替代終點。

RMI-IIOP 範例應用程式

以下目錄包含用於演示透過和不透過 ACC 使用 RMI-IIOP 防故障備用的範例應用程式。

install_dir`/samples/ee-samples/sfsbfailover`

請參閱該範例附帶的 `index.html` 檔案，以取得有關透過和不透過 ACC 來執行應用程式的說明。`ee-samples` 目錄還包含有關設定執行範例的環境的資訊。

第 4 章

配置節點代理程式

本章介紹 Application Server 中的節點代理程式。它包含以下各節：

- 關於節點代理程式
- 用於節點代理程式的管理主控台作業
- asadmin 工具中用於節點代理程式的作業

關於節點代理程式

- 節點代理程式
- 節點代理程式萬用字元
- 部署節點代理程式
- 節點代理程式和網域管理伺服器同步
- 檢視節點代理程式記錄
- 可以通過管理主控台和 asadmin 工具執行的作業

節點代理程式

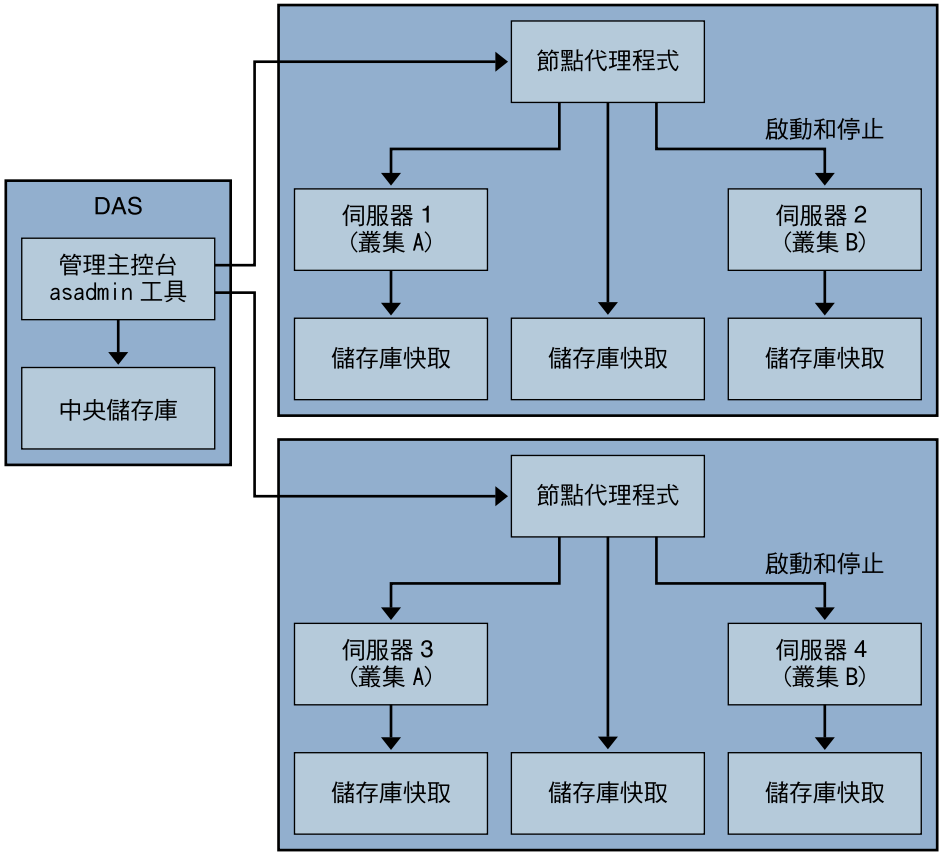
節點代理程式是託管伺服器實例的每台機器 (包括託管網域管理伺服器 [DAS] 的機器) 上都需要的輕型代理程式。節點代理程式可以：

- 按照網域管理伺服器的指示啓動、停止、建立和刪除伺服器實例。
- 重新啓動有故障的伺服器實例。
- 提供有故障的伺服器的記錄檔檢視。

- 使每個伺服器實例的本機配置儲存庫與網域管理伺服器的中央儲存庫同步。每個本機儲存庫只包含與該伺服器實例或節點代理程式相關的資訊。

下圖說明了節點代理程式的整體架構：

圖 4-1 節點代理程式架構



自動建立的節點代理程式

安裝 Application Server 時，依預設將使用該機器的主機名稱建立節點代理程式。必須先在本地機器上手動啓動節點代理程式，該節點代理程式才能執行。

節點代理程式和伺服器實例管理

即使未執行節點代理程式，您也可以建立和刪除伺服器實例。但是，節點代理程式必須處於執行狀態，您才能用它來啓動和停止伺服器實例。

如果停止節點代理程式，則該節點代理程式所管理的伺服器實例也將被停止。

附加節點代理程式

一個節點代理程式服務單一網域。如果某台機器託管在多個網域中執行的實例，則該機器必須執行多個節點代理程式。

節點代理程式萬用字元

您可以使用節點代理程式萬用字元建立和刪除不具有現有節點代理程式的伺服器實例。萬用字元是在節點代理程式的本機系統中建立節點代理程式本身之前，在網域管理伺服器 (DAS) 上建立的節點代理程式配置。

備註 當您建立萬用字元節點代理程式之後，即可使用該萬用字元節點代理程式在網域中建立實例。但是，在啓動實例之前，您必須先使用 `asadmin` 指令在實例將要常駐的機器上本機建立並啓動實際的節點代理程式。請參閱第 96 頁的「建立節點代理程式」和第 97 頁的「啓動節點代理程式」，以取得更多資訊。

部署節點代理程式

您可以使用以下兩種方式之一來配置和部署節點代理程式：

- 線上部署。如果您在配置網域之前，已經知道其拓樸並具有其硬體，請使用線上部署
- 離線部署。如果是在設定整個環境之前配置網域和伺服器實例，請使用離線部署

部署節點代理程式之前

部署節點代理程式之前，請執行以下步驟：

1. 安裝網域管理伺服器。
2. 啓動網域管理伺服器。

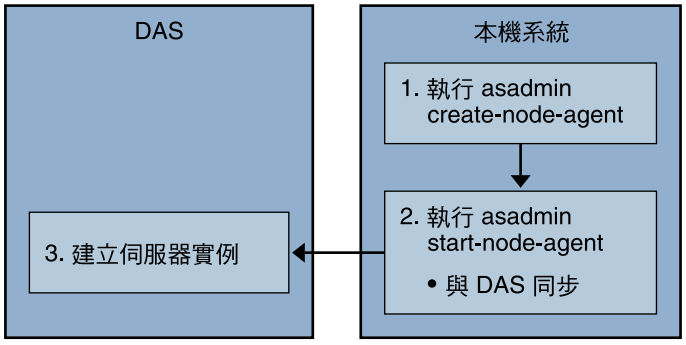
啓動並執行網域管理伺服器之後，便可以開始進行線上或離線部署。

線上部署

如果您在開始配置網域之前已經知道其拓樸並具有其硬體，則可以通過線上部署來配置該網域。

下圖概括了節點代理程式的線上部署：

圖 4-2 線上節點代理程式部署



若要配置線上部署，請執行以下步驟：

1. 在將要託管伺服器實例的每台機器上安裝節點代理程式。

   使用安裝程式或 `asadmin` 指令 `create-node-agent`。如果某台機器需要多個節點代理程式，請使用 `asadmin` 指令 `create-node-agent` 來建立這些節點代理程式。

   請參閱第 96 頁的「建立節點代理程式」，以取得更多資訊。

2. 使用 `asadmin` 指令 `start-node-agent` 啟動節點代理程式。

   啟動之後，節點代理程式將與網域管理伺服器 (DAS) 進行通訊。當節點代理程式到達 DAS 時，將在 DAS 上建立該節點代理程式的配置。具備了配置之後，即可在管理主控台中檢視該節點代理程式。

   請參閱第 97 頁的「啟動節點代理程式」，以取得更多資訊。

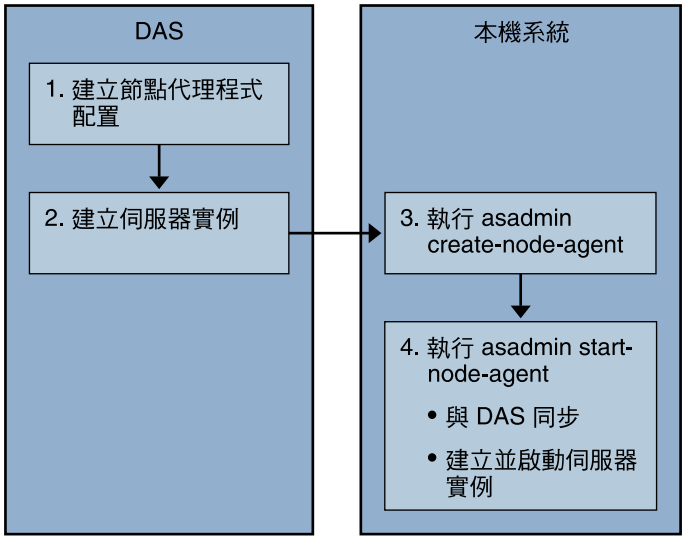
3. 配置網域，包括建立伺服器實例和叢集以及部署應用程式。

離線部署

離線方法使您可以在配置各個本地機器之前輕鬆地定義和重新安排網域中的項目。

下圖概括了離線部署的步驟。

圖 4-3 離線節點代理程式部署



若要在本地機器中設定節點代理程式之前配置網域和伺服器實例 (離線配置)，請執行以下步驟：

1. 在網域管理伺服器中建立萬用字元節點代理程式。

   請參閱第 93 頁的「建立節點代理程式萬用字元」，以取得更多資訊。

2. 建立伺服器實例和叢集並部署應用程式。

3. 在將要託管伺服器實例的每台機器上安裝節點代理程式。

   使用安裝程式或 `asadmin` 指令 `create-node-agent`。節點代理程式的名稱必須與先前建立的萬用字元節點代理程式的名稱相同。

   請參閱第 96 頁的「建立節點代理程式」，以取得更多資訊。

4. 使用 `asadmin` 指令 `start-node-agent` 啓動節點代理程式。

   節點代理程式啓動之後，將連結到網域管理伺服器並建立先前已與該節點代理程式相關聯的所有伺服器實例。

   請參閱第 97 頁的「啓動節點代理程式」，以取得更多資訊。

節點代理程式和網域管理伺服器同步

由於配置資料既儲存在網域管理伺服器的儲存庫 (中央儲存庫) 中，又快取在節點代理程式的本地機器中，因此這兩者必須同步。

節點代理程式同步

第一次啓動節點代理程式時，該節點代理程式將向網域管理伺服器 (DAS) 傳送一個請求，以獲得中央儲存庫中的最新資訊。當節點代理程式成功地與 DAS 取得聯絡並獲得配置資訊時，該節點代理程式即被**連結**到該 DAS。

如果您在 DAS 上建立了萬用字元節點代理程式，則第一次啓動節點代理程式時，該節點代理程式將從 DAS 的中央儲存庫中獲取其配置。

初始啓動節點代理程式期間，如果由於沒有執行 DAS 而使該節點代理程式無法連結 DAS，該節點代理程式將停止並保持**未連結**狀態。

如果在網域中變更了節點代理程式的配置，這些變更將在節點代理程式執行時自動傳送給本地機器中的節點代理程式。

如果在 DAS 中刪除了節點代理程式的一個配置，則下次該節點代理程式進行同步時將自行停止並會將其自身標記爲待刪除狀態。可以使用本機 `asadmin` 指令 `delete-node-agent` 手動刪除該節點代理程式。

伺服器實例同步

如果使用管理主控台或 `asadmin` 工具明確啓動了伺服器實例，該伺服器實例將與中央儲存庫同步。如果此同步失敗，則伺服器實例不會啓動。

如果節點代理程式不是通過向管理主控台或 `asadmin` 工具傳送明確的請求來啓動伺服器實例，將不同步該伺服器實例的儲存庫快取。該伺服器實例將以儲存在其快取中的配置執行。

同步大型應用程式

當環境中有大型應用程式要進行同步或者可用記憶體受到限制時，您可以調整 JVM 選項以限制記憶體的使用。這種調整將減少收到記憶體不足錯誤的可能性。實例同步 JVM 使用的是預設設定，但您可以配置 JVM 選項來變更這些設定。

使用 `INSTANCE-SYNC-JVM-OPTIONS` 特性設定 JVM 選項。用於設定特性的指令爲：

```
asadmin set domain.node-agent.node_agent_name.property.INSTANCE-SYNC-JVM-OPTIONS="JVM_options"
```

例如：

```
asadmin set domain.node-agent.node0.property.INSTANCE-SYNC-JVM-OPTIONS="-Xmx32m -Xss2m"
```

在此範例中，節點代理程式爲 node0，JVM 選項爲 -Xmx32m -Xss2m。

如需有關 JVM 選項的更多資訊，請參閱：

http://java.sun.com/docs/hotspot/VMOptions.html

備註 變更 INSTANCE-SYNC-JVM-OPTIONS 特性後，請重新啓動節點代理程式，因爲在節點代理程式的配置中新增或變更特性時節點代理程式並不會自動同步。

檢視節點代理程式記錄

每個節點代理程式都有自己的記錄檔。如果使用節點代理程式時遇到問題，請參閱位於以下位置的記錄檔：

node_agent_dir/*node_agent_name*/agent/logs/server.log.

有時，節點代理程式記錄會指示您檢視伺服器的記錄以獲得關於所出現的問題的詳細資訊。

伺服器記錄位於：

node_agent_dir/*node_agent_name*/*server_name*/logs/server.log

node_agent_dir 的預設位置爲 *install_dir*/nodeagents。

可以通過管理主控台和 asadmin 工具執行的作業

對於節點代理程式，有些作業必須從執行該節點代理程式的系統上本機執行，而其他作業則可以在網域管理伺服器中執行。需要本機執行的作業只能在節點代理常駐的機器中通過執行 asadmin 工具來執行。而在網域管理伺服器中執行的作業通過管理主控台或是 asadmin 工具都可以執行。

下表概括了這些作業以及執行這些作業的位置：

表 4-1 通過管理主控台或是 asadmin 指令都可以執行的作業

作業	管理主控台	asadmin 指令
在網域管理伺服器中建立節點代理程式萬用字元/配置	建立 [節點代理程式萬用字元] 頁面	create-node-agent-config
建立節點代理程式	不可用	create-node-agent
啟動節點代理程式	不可用	start-node-agent

表 4-1 通過管理主控台或是 asadmin 指令都可以執行的作業

作業	管理主控台	asadmin 指令
停止節點代理程式	不可用	stop-node agent
從網域管理伺服器中刪除節點代理程式配置	[節點代理程式] 頁面	delete-node-agent-config
從本地機器中刪除節點代理程式	不可用	delete-node-agent
編輯節點代理程式配置	[節點代理程式] 頁面	set
列出節點代理程式	[節點代理程式] 頁面	list-node-agents

用於節點代理程式的管理主控台作業

- 檢視一般節點代理程式資訊
- 建立節點代理程式萬用字元
- 刪除節點代理程式配置
- 編輯節點代理程式配置
- 編輯節點代理程式範圍
- 爲 JMX 編輯節點代理程式的偵聽程式

檢視一般節點代理程式資訊

建立節點代理程式或節點代理程式預留位置字元後，若要檢視其設定，請執行以下步驟：

1. 在樹形元件中，選取 [節點代理程式] 節點。
2. 按一下節點代理程式的名稱。

   如果節點代理程式已經存在但並未在此處顯示，請在節點代理程式的主機電腦上使用 `asadmin start-node-agent` 來啓動該節點代理程式。請參閱第 97 頁的「啓動節點代理程式」，以取得更多資訊。
3. 檢查節點代理程式的主機名稱。

   如果主機名稱爲 [未知主機]，則該節點代理程式尚未同網域管理伺服器 (DAS) 進行初始聯絡。

4. 檢查節點代理程式的狀態。

   正在執行。節點代理程式已正確建立，並且目前處於執行狀態。

   未執行。存在以下情形之一：

   - 已在本地機器中建立了節點代理程式，但從未啓動過該節點代理程式。
   - 以前啓動過節點代理程式，但現在已停止。

   等待會合。節點代理程式是從未在本地機器中建立的萬用字元。

   如需有關建立和啓動節點代理程式的更多資訊，請參閱第 96 頁的「建立節點代理程式」和第 97 頁的「啓動節點代理程式」。

5. 選擇啓動節點代理程式時是否啓動實例。

   選擇 [是]，將在啓動節點代理程式時自動啓動與該節點代理程式關聯的伺服器實例。選擇 [否] 將手動啓動這些實例。

6. 確定節點代理程式是否已同網域管理伺服器進行了聯絡。

   如果節點代理程式從未與網域管理伺服器進行過聯絡，則表明該節點代理程式從未成功啓動過。

7. 管理與節點代理程式關聯的伺服器實例。

   如果節點代理程式正在執行，則可以透過按一下實例名稱旁邊的核取方塊並按一下 [啓動] 或 [停止] 來啓動或停止實例。

建立節點代理程式萬用字元

由於必須從託管節點代理程式的機器上本機建立節點代理程式，因此通過管理主控台您只能爲節點代理程式建立萬用字元。此萬用字元是尚不存在的節點代理程式的節點代理程式配置。

建立一個萬用字元之後，請在託管節點代理的機器中使用 `asadmin` 指令 `create-node-agent` 完成節點代理程式的建立。如需更多資訊，請參閱第 96 頁的「建立節點代理程式」。

1. 在樹形元件中，選取 [節點代理程式] 節點。
2. 在 [節點代理程式] 頁面中，按一下 [新建]。

3. 在 [目前節點代理程式萬用字元] 頁面中，爲新節點代理程式輸入一個名稱。

   對於網域中的所有節點代理程式名稱、伺服器實例名稱、叢集名稱和配置名稱中，此名稱必須是唯一的。

4. 按一下 [確定]。

   新節點代理程式的萬用字元將列示在 [節點代理程式] 頁面中。

等效的 asadmin 指令爲：create-node-agent-config。

刪除節點代理程式配置

通過管理主控台，您只能刪除網域中的節點代理程式配置，而不能刪除實際的節點代理程式。若要刪除節點代理程式本身，請在該節點代理程式的本地機器中執行 asadmin 指令 delete-node-agent。如需更多資訊，請參閱第 98 頁的 「刪除節點代理程式」。

刪除節點代理程式配置之前，該節點代理程式必須停止並且不能有任何關聯的實例。若要停止節點代理程式，請使用 asadmin 指令 stop-node-agent。請參閱第 98 頁的 「停止節點代理程式」，以取得更多資訊。

1. 在樹形元件中，選取 [節點代理程式] 節點。
2. 在 [節點代理程式] 頁面中，選取要刪除的節點代理程式旁邊的核取方塊。
3. 按一下 [刪除]。

等效的 asadmin 指令爲：delete-node-agent-config。

編輯節點代理程式配置

若要編輯節點代理程式配置，請執行以下步驟：

1. 在樹形元件中，展開 [節點代理程式] 節點。
2. 選取要編輯的節點代理程式配置。
3. 在 [節點代理程式一般資訊] 頁面中，選擇啓動代理程式時是否啓動代理程式的伺服器實例。您還可以從此頁面手動啓動和停止實例。

如果此配置是萬用字元節點代理程式的配置，則使用 asadmin create-node-agent 建立實際的節點代理程式時，實際的節點代理程式將使用此配置。如需有關建立節點代理程式的資訊，請參閱第 96 頁的 「建立節點代理程式」。

如果此配置是現有節點代理程式的配置，則將自動同步此節點代理程式配置資訊。

編輯節點代理程式範圍

爲連線到節點代理程式的使用者設定認證範圍。只有管理使用者才能存取節點代理程式。

1. 在樹形元件中，展開 [節點代理程式] 節點。
2. 選取要編輯的節點代理程式配置。
3. 按一下 [認證範圍] 標籤。
4. 在 [節點代理程式編輯範圍] 頁面中，輸入一個範圍。

   預設範圍爲 `admin-realm`，它是您在建立節點代理程式時建立的。若要使用其他範圍，請用新範圍替代由網域控制的**所有**元件中的範圍，否則這些元件將無法正常通訊。
5. 在 [類別名稱] 欄位中，指定實作該範圍的 Java 類別。
6. 新增需要的所有特性。

   認證範圍需要特定於供應商的特性，這些特性會因特定實作的需求不同而有所不同。

為 JMX 編輯節點代理程式的偵聽程式

節點代理程式使用 JMX 與網域管理伺服器進行通訊。因此，該節點代理程式必須具有偵聽 JMX 請求的連接埠以及其他偵聽程式資訊。

1. 在樹形元件中，展開 [節點代理程式] 節點。
2. 選取要編輯的節點代理程式配置。
3. 按一下 [JMX] 標籤。
4. 如果偵聽程式將偵聽伺服器的所有 IP 位址，請在 [位址] 欄位中鍵入 0.0.0.0 並使用唯一的連接埠值。否則，請鍵入此伺服器的有效 IP 位址。
5. 在 [連結埠] 欄位中，鍵入節點代理程式的 JMX 連接器要偵聽的連接埠值。如果 IP 位址爲 0.0.0.0，則連接埠號必須唯一。
6. 在 [JMX 協定] 欄位中，鍵入 JMX 連接器支援的協定。

   預設值爲 rmi_jrmp。
7. 按一下 [接受所有位址] 旁邊的核取方塊以允許連線到所有 IP 位址。

   節點代理程式將偵聽與網路卡相關聯的特定 IP 位址或偵聽所有 IP 位址。如果接受所有位址，值 `0.0.0.0` 將會放入 [偵聽主機位址] 特性中。

8. 在 [範圍名稱] 欄位中，鍵入爲偵聽程式處理認證的範圍的名稱。

在此頁面的 [安全性] 區段中，您可以將偵聽程式配置爲使用 SSL 安全性和/或 TLS 安全性

若要設定安全偵聽程式，請執行以下操作：

1. 在 [安全性] 欄位中核取 [啓用] 方塊。

   依預設已啓用 [安全性]。

2. 若要使用戶端在使用此偵聽程式時自行向伺服器進行認證，請在 [用戶端認證] 欄位中核取 [啓用] 方塊。
3. 在 [證書暱稱] 欄位中輸入現有伺服器金鑰組和證書的名稱。請參閱 「安全性」一章以取得更多資訊。
4. 在 [SSL3/TLS] 區段：
   a. 核取要在偵聽程式上啓用的安全性協定。必須核取 SSL3 和/或 TLS。
   b. 選取協定所使用的密碼組。若要啓用所有密碼組，請核取 [所有支援的密碼組]。
5. 按一下 [儲存]。

asadmin 工具中用於節點代理程式的作業

- 建立節點代理程式
- 啓動節點代理程式
- 停止節點代理程式
- 刪除節點代理程式

建立節點代理程式

若要建立節點代理程式，請在執行該節點代理程式的機器中本機執行 `asadmin` 指令 `create-node-agent`。

例如：

```
$ asadmin create-node-agent --host myhost --port 4849 ---user admin
nodeagent1
```

其中，myhost 是網域管理伺服器 (DAS) 主機名稱，4849 是 DAS 連接埠號，admin 是 DAS 使用者，nodeagent1 是要建立的節點代理程式的名稱。

節點代理程式的預設名稱爲該節點代理程式建立時所在的主機的名稱。

如果已建立節點代理程式萬用字元，請使用與節點代理程式萬用字元相同的名稱來建立關聯的節點代理程式。如果您尚未建立節點代理程式萬用字元，而 DAS 已啓動並且可進行連線，則 create-node-agent 指令還將在 DAS 上建立節點代理程式配置 (萬用字元)。

如需有關指令語法的完整描述，請參閱該指令的線上說明。

備註 在以下情況下，必須指定一個可連線 DNS 的主機名稱：

1. 網域跨越了子網路邊界 (即，節點代理程式和網域管理伺服器 [DAS] 處於不同的網域中，例如 sun.com 和 java.com)
2. 使用的是主機名稱未在 DNS 中註冊的 DHCP 機器

建立網域和節點代理程式時，可透過明確指定網域和節點代理程式的主機名稱來指定一個可連線 DNS 的主機名稱：

```
create-domain --domainproperties domain.hostName=DAS-host-name
```

```
create-node-agent --host DAS-host-name --agentproperties
remoteclientaddress=node-agent-host-name
```

另外一種解決方案是更新特定於平台的 hosts 主機名稱 /IP 解析檔案，從而將主機名稱解析為正確的 IP 位址。但是，使用 DHCP 重新連線時，可能會指定給您一個不同的 IP 位址。在這種情況下，您必須更新每個伺服器中的主機解析檔案。

啟動節點代理程式

節點代理程式必須處於執行狀態，然後才能管理伺服器實例。透過在節點代理程式常駐的系統中本機執行 asadmin 指令 start-node-agent 來啓動節點代理程式。

例如：

```
$ asadmin start-node-agent --user admin nodeagent1
```

其中，admin 是管理使用者，nodeagent1 是要啓動的節點代理程式。

如需有關指令語法的完整描述，請參閱該指令的線上說明。

停止節點代理程式

若要停止正在執行的節點代理程式，請在該節點代理程式常駐的系統中執行 asadmin 指令 stop-node-agent。stop-node-agent 指令將停止該節點代理程式所管理的所有伺服器實例。

例如：

```
$ asadmin stop-node-agent nodeagent1
```

其中的 nodeagent1 是節點代理程式。

如需有關指令語法的完整描述，請參閱該指令的線上說明。

刪除節點代理程式

若要刪除節點代理程式檔案，請在該節點代理程式常駐的系統中執行 asadmin 指令 delete-node-agent。

刪除節點代理程式之前，必須先停止節點代理程式。您還可以刪除從未啓動過或者從未成功地與網域管理伺服器聯絡 (即尚未連結) 的節點代理程式。

例如：

```
$ asadmin delete-node-agent nodeagent1
```

其中的 nodeagent1 是節點代理程式。

如需有關指令語法的完整描述，請參閱該指令的線上說明。

刪除某個節點代理程式時，您還必須使用管理主控台或 asadmin delete-node-agent-config 指令從網域管理伺服器中刪除該節點代理程式的配置。

第 5 章

部署應用程式

本章說明如何在 Application Server 上部署 (安裝) J2EE 應用程式。它包含以下小節：

- 「關於部署」
- 「有關部署應用程式的管理主控台作業」
- 「有關列示、取消部署以及啓用應用程式的管理主控台作業」
- 「適用於開發者的部署方法」

關於部署

- 「部署生命週期」
- 「J2EE 歸檔檔案的類型」
- 「命名慣例」

部署生命週期

安裝 Application Server 並啓動網域之後，您可以部署 (安裝) J2EE 應用程式和模組。在部署期間和變更應用程式時，應用程式或模組可能會經過以下階段：

1. 初始部署

   部署應用程式或模組之前，請啓動網域。

   將應用程式或模組部署 (安裝) 到特定的獨立伺服器實例或叢集。由於應用程式和模組封裝在歸檔檔案中，因此在部署期間應指定歸檔檔案名稱。預設爲部署到預設伺服器實例 `server`。

如果部署到伺服器實例或叢集，則應用程式或模組將存在於網域的中央儲存庫中，並由部署到的所有目標叢集或伺服器實例參考。

您還可以使用 `asadmin deploy` 指令 (而非管理主控台) 部署到網域。如果將應用程式或模組僅部署到網域，則應用程式或模組將存在於網域的中央儲存庫中，但要在新增參考之後才會有伺服器實例或叢集參考該應用程式或模組 (如步驟 3 中所述)。

部署是動態的：部署應用程式或模組後，無需重新啓動伺服器實例即可使用應用程式。如果重新啓動了伺服器實例，所有已部署的應用程式和模組仍將處於部署狀態並且可用。

2. 啓用或停用

   依預設，將啓用已部署的應用程式或模組，這表示如果應用程式或模組已部署到可存取的伺服器實例或叢集，則可以執行它並且可以藉由用戶端對其進行存取。若要防止存取，請停用應用程式或模組。在部署之後，已停用的應用程式或模組並未從網域中解除安裝，而且可以輕鬆地將其啓用。

3. 新增或刪除已部署應用程式或模組的目標

   部署後，應用程式或模組將存在於中央儲存庫中，並可以由多個伺服器實例和/或叢集參考。最初，作爲目標部署到的伺服器實例或叢集將參考應用程式或模組。

   在部署應用程式或模組後，若要變更參考應用程式或模組的伺服器實例和叢集，請使用管理主控台變更應用程式或模組的目標，或使用 asadmin 工具變更應用程式參考。由於應用程式本身儲存在中央儲存庫中，因此新增或刪除目標將新增或刪除不同目標上同一版本的應用程式。但是，可以在一個目標上啓用而在另一個目標上停用部署到多個目標的應用程式，因此即使應用程式被一個目標參考，也只有在該目標上啓用它時使用者才能對其進行使用。

4. 重新部署

   若要替代已部署的應用程式或模組，請將其重新部署。重新部署將自動取消部署先前已部署的應用程式或模組，並用新的應用程式或模組對其進行替代。

   當通過管理主控台重新部署時，重新部署的應用程式或模組將部署到網域中，並且所有對其進行參考的獨立或叢集伺服器實例將自動接收新的版本 (如果已啓用動態重新配置)。如果使用 `asadmin deploy` 指令來重新部署，請將 `domain` 指定爲目標。

   對於生產環境，請使用捲動升級 (升級應用程式而不中斷服務)。如需更多資訊，請參閱第 79 頁的「關於捲動升級」。

5. 取消部署

   若要解除安裝應用程式或模組，請取消部署應用程式或模組。

J2EE 歸檔檔案的類型

軟體供應商將應用程式或模組封裝在歸檔檔案中。若要部署應用程式或模組，請指定歸檔檔案名稱。歸檔檔案的內容和結構是按照 J2EE 平台的規格定義的。J2EE 歸檔檔案的類型有以下幾種：

- Web 應用程式歸檔檔案 (WAR)：WAR 檔案由 Servlet 和 JSP 等 Web 元件以及靜態 HTML 頁面、JAR 檔案、標籤檔案庫和公用程式類別組成。WAR 檔案名稱具有 `.war` 副檔名。
- EJB JAR：EJB JAR 檔案包含一個或多個企業 Bean (用於 EJB 技術的元件)。EJB JAR 檔案還包括企業 Bean 所需的任何公用程式類別。EJB JAR 檔案的名稱具有 `.jar` 副檔名。
- J2EE 應用程式用戶端 JAR：該 JAR 檔案包含藉由 RMI/IIOP 存取伺服器端元件 (如企業 Bean) 的 J2EE 應用程式用戶端的程式碼。在管理主控台中，J2EE 應用程式用戶端被稱爲「應用程式用戶端」。J2EE 應用程式用戶端 JAR 檔案的名稱具有 `.jar` 副檔名。
- 資源介面歸檔檔案 (RAR)：RAR 檔案存有資源介面。資源介面是按照 J2EE 連接器架構規格定義的，它是允許企業 Bean 和 Web 元件和應用程式用戶端存取資源和外部企業系統的可攜式元件。資源介面經常稱爲連接器。RAR 檔案名稱具有 `.rar` 副檔名。
- 企業應用程式歸檔檔案 (EAR)：EAR 檔案存有一個或多個 WAR 檔案、EJB JAR 檔案、RAR 檔案或 J2EE 應用程式用戶端 JAR 檔案。EAR 檔案名稱具有 `.ear` 副檔名。

軟體供應商可以將應用程式組譯爲單一 EAR 檔案或多個獨立的 WAR 檔案、EJB JAR 檔案和應用程式用戶端 JAR 檔案。在管理工具中，用於所有類型檔案的部署頁面和指令都是類似的。

命名慣例

在給定網域中，已部署的應用程式名稱和模組名稱必須唯一。

- 如果使用管理主控台進行部署，請在 [應用程式名稱] 欄位中指定名稱。
- 如果使用 `asadmin deploy` 指令進行部署，則應用程式或模組的預設名稱爲要部署的 JAR 檔案的字首。例如，如果部署 `hello.war` 檔案，則 Web 應用程式的名稱爲 `hello`。若要置換預設名稱，請指定 `--name` 選項。

在一個應用程式中，不同類型的模組可以具有相同的名稱。部署應用程式時，將使用 `_jar`、`_war` 和 `_rar` 字尾來命名存有各個模組的目錄。一個應用程式內，類型相同的模組必須具有唯一的名稱。此外，在一個應用程式內，資料庫綱目檔的名稱必須是唯一的。

建議將類似於 Java 套裝軟體的命名機制用於在 `ejb-jar.xml` 檔案的 `<module-name>` 部分中找到的模組檔名、EAR 檔名、模組名稱，以及在 `ejb-jar.xml` 檔案的 `<ejb-name>` 部分找到的 EJB 名稱。使用這種類似於套裝軟體的命名機制可以確保不會發生名稱衝突。該命名慣例的優勢不僅適用於 Sun Java System Application Server，也適用於其他 J2EE 應用程式伺服器。

EJB 的 JNDI 查找名稱也必須是唯一的。建立連續的命名慣例可能會非常有用。例如，將應用程式名稱和模組名稱附加到 EJB 名稱中是保證名稱唯一性的方式。在這種情況下，mycompany.pkging.pkgingEJB.MyEJB 就是模組 pkgingEJB.jar 內 EJB 的 JNDI 名稱，該模組封裝於應用程式 pkging.ear 中。

請確定該套裝軟體和檔案的名稱不含有空格或作業系統不支援的非法字元。

有關部署應用程式的管理主控台作業

- 「部署企業應用程式」
- 「部署 Web 應用程式」
- 「啓動已部署的 Web 應用程式」
- 「部署 EJB 模組」
- 「部署應用程式用戶端模組」
- 「部署連接器模組」
- 「建立生命週期模組」
- 「部署應用程式用戶端模組」

部署企業應用程式

企業應用程式封裝在 EAR 檔案中，EAR 檔案是包含任意 J2EE 獨立模組 (例如 WAR 檔案和 EJB JAR 檔案) 的一種歸檔檔案。

若要部署 (安裝) 企業應用程式，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。
2. 選取 [企業應用程式] 節點。
3. 在 [企業應用程式] 頁面中，按一下 [部署]。
4. 在 [部署] 頁面中，指定要部署的 EAR 檔案的位置。

   伺服器是執行應用程式伺服器和網域管理伺服器的主機。用戶端是可以用於透過瀏覽器來檢視管理主控台的主機。

   a. 如果檔案位於用戶端上或可以從用戶端存取，則按一下單選按鈕以指定套裝軟體檔案以上傳到 Application Server。

      按一下 [瀏覽] 以找到檔案，或鍵入檔案的完整路徑。

   b. 如果檔案位於伺服器上，或者要從展開的目錄部署未封裝的應用程式，則按一下單選按鈕以指定必須可以從伺服器上存取的套裝軟體檔案或目錄路徑。

      鍵入檔案或目錄的完整路徑名稱。從展開的目錄部署適用於進階開發者，但不建議在生產環境下進行此操作。

5. 按一下 [下一步]，以顯示 [部署企業應用程式] 頁面。
6. 在 [部署企業應用程式] 頁面中，請指定應用程式的設定。

   a. 在 [應用程式名稱] 欄位中，既可以保留預設名稱 (即檔案名稱的字首)，也可以鍵入其他名稱。(如果您選擇上傳檔案，將顯示預設名稱。) 應用程式名稱必須唯一。

   b. 依預設，部署應用程式之後即可使用該應用程式。若要在部署之後停用應用程式以使其不可用，請選取 [已停用] 單選按鈕。

   c. 如果已部署應用程式，請選取 [重新部署] 核取方塊以對其進行重新部署；否則將顯示錯誤。您也可以選擇其他應用程式名稱，並以新名稱對其進行部署。

   d. 若要在部署之前檢驗檔案的結構和內容，請選取 [檢驗器] 核取方塊。大型應用程式的檢驗會很費時。如果懷疑檔案已毀壞或爲不可攜式，請檢驗檔案。

e. 若要預編譯 JSP 頁面，請選取 [JSP] 核取方塊。如果未選取此核取方塊，則首次存取 JSP 頁面時會在執行階段編譯這些頁面。由於編譯通常很費時，因此在生產環境中請選取此核取方塊。

f. 選擇高可用性設定。

若要啓用應用程式的高可用性，請選取 [可用性] 核取方塊。如果啓用了應用程式的可用性，則必須也在所有更高層級 (指配置和 Web 容器或 EJB 容器) 啓用可用性。

g. 選擇要將應用程式部署到的目標。

從可用目標的清單中選擇目標，並按一下 [新增]。目標可以是叢集或獨立伺服器實例。如果不選取目標，應用程式將被部署到預設伺服器實例 `server`。

如果您要重新部署，請勿選取目標。此時您所作的任何選取都將被忽略。參考已部署的應用程式的所有目標叢集或獨立伺服器實例都將自動參考新的、重新部署的應用程式 (如果已啓用叢集或獨立實例的動態重新配置)。如需有關如何在不中斷服務的情況下重新部署應用程式的更多資訊，請參閱第 79 頁的 「升級應用程式」。

h. 選擇是否產生 RMI 存根。

如果選擇產生 RMI 存根，將產生靜態 RMI-IIOP 存根並將其新增到 `client.jar`。

7. 按一下 [確定] 以部署應用程式。

等效的 `asadmin` 指令爲：`deploy`

部署 Web 應用程式

Web 應用程式封裝在 WAR 檔案中，WAR 檔案是包含元件 (例如 servlet 和 JSP 頁面) 的一種歸檔檔案。

若要部署 (安裝) Web 應用程式，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。

2. 選取 [Web 應用程式] 節點。

3. 在 [Web 應用程式] 頁面中，按一下 [部署]。

4. 在 [部署] 頁面中，指定要部署的 WAR 檔案的位置。

伺服器是執行應用程式伺服器和網域管理伺服器的主機。用戶端是可以用於透過瀏覽器來檢視管理主控台的主機。

a. 如果檔案位於用戶端上或可以從用戶端存取，則按一下單選按鈕以指定套裝軟體檔案以上傳到 Application Server。

按一下 [瀏覽] 以找到檔案，或鍵入檔案的完整路徑。

b. 如果檔案位於伺服器上，或者要從展開的目錄部署未封裝的應用程式，則按一下單選按鈕以指定必須可以從伺服器上存取的套裝軟體檔案或目錄路徑。

鍵入檔案或目錄的完整路徑名稱。從展開的目錄部署適用於進階開發者，但不建議在生產環境下進行此操作。

5. 按一下 [下一步] 以顯示 [部署 Web 應用程式] 頁面。

6. 在 [部署 Web 應用程式] 頁面中，請指定應用程式的設定。

a. 在 [應用程式名稱] 欄位中，既可以保留預設名稱 (即檔案名稱的字首)，也可以鍵入其他名稱。(如果您選擇上傳檔案，將顯示預設名稱。) 應用程式名稱必須唯一。

b. 在 [環境根] 欄位中，輸入識別 Web 應用程式的字串。在 Web 應用程式的 URL 中，環境根要緊跟著連接埠號 (`http://host:port/context-root/...`)。確定環境根以正斜線開頭，例如：`/hello`

c. 依預設，部署應用程式之後即可使用該應用程式。若要在部署之後停用應用程式以使其不可用，請選取 [已停用] 單選按鈕。

d. 如果已部署應用程式，請選取 [重新部署] 核取方塊以對其進行重新部署；否則將顯示錯誤。您也可以選擇其他應用程式名稱，並以新名稱對其進行部署。

e. 若要在部署之前檢驗檔案的結構和內容，請選取 [檢驗器] 核取方塊。大型應用程式的檢驗通常會很費時。如果懷疑檔案已毀壞或爲不可攜式，請檢驗檔案。

f. 若要預編譯 JSP 頁面，請選取 [JSP] 核取方塊。如果未選取此核取方塊，則首次存取 JSP 頁面時會在執行階段編譯這些頁面。由於編譯通常很費時，因此在生產環境中請選取此核取方塊。

g. 選擇高可用性設定。

若要啓用應用程式的高可用性，請選取 [可用性] 核取方塊。如果啓用了應用程式的可用性，則必須也在所有更高層級 (指配置和 Web 容器或 EJB 容器) 啓用可用性。

h. 選擇要將應用程式部署到的目標。

從可用目標的清單中選擇目標，並按一下 [新增]。目標可以是叢集或獨立伺服器實例。如果不選取目標，應用程式將被部署到預設伺服器實例 `server`。

如果您要重新部署，請勿選取目標。此時您所作的任何選取都將被忽略。參考已部署的應用程式的所有目標叢集或獨立伺服器實例都將自動參考新的、重新部署的應用程式 (如果已啓用叢集或獨立實例的動態重新配置)。如需有關如何在不中斷服務的情況下重新部署應用程式的更多資訊，請參閱第 79 頁的 「關於捲動升級」。

i. 選擇是否產生 RMI 存根。

如果選擇產生 RMI 存根，將產生靜態 RMI-IIOP 存根並將其新增到 `client.jar`。

7. 按一下 [確定] 以部署應用程式。

等效的 `asadmin` 指令爲：`deploy`

啟動已部署的 Web 應用程式

部署完應用程式之後，您可以從管理主控台啓動該應用程式。

1. 在樹形元件中，展開 [應用程式] 節點。

2. 按一下 [Web 應用程式]。

3. 按一下 Web 應用程式的 [啓動] 連結。

4. 在 [Web 應用程式連結] 頁面中按一下連結以啓動應用程式。

伺服器和 HTTP 偵聽程式必須正在執行，應用程式才能啓動。

部署 EJB 模組

EJB 模組也稱爲 EJB JAR 檔案，它包含企業 Bean。

若要部署 (安裝) EJB 模組，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。
2. 選取 [EJB 模組] 節點。
3. 在 [EJB 模組] 頁面中，按一下 [部署]。
4. 在 [部署] 頁面中，指定要部署的 JAR 檔案的位置。

   伺服器是執行應用程式伺服器和網域管理伺服器的主機。用戶端是可以用於透過瀏覽器來檢視管理主控台的主機。

   a. 如果檔案位於用戶端上或可以從用戶端存取，則按一下單選按鈕以指定套裝軟體檔案以上傳到 Application Server。

      按一下 [瀏覽] 以找到檔案，或鍵入檔案的完整路徑。

   b. 如果檔案位於伺服器上，或者要從展開的目錄部署未封裝的應用程式，則按一下單選按鈕以指定必須可以從伺服器上存取的套裝軟體檔案或目錄路徑。

      鍵入檔案或目錄的完整路徑名稱。從展開的目錄部署適用於進階開發者，但不建議在生產環境下進行此操作。
5. 按一下 [下一步] 以顯示 [部署 EJB 模組] 頁面。
6. 在 [部署 EJB 模組] 頁面中，請指定模組的設定。

   a. 在 [應用程式名稱] 欄位中，既可以保留預設名稱 (即檔案名稱的字首)，也可以鍵入其他名稱。(如果您選擇上傳檔案，將顯示預設名稱。) 應用程式名稱必須唯一。

   b. 依預設，部署模組之後即可使用該模組。若要在部署之後停用模組以使其不可用，請選取 [已停用] 單選按鈕。

   c. 如果已部署模組，請選取 [重新部署] 核取方塊以對其進行重新部署；否則將顯示錯誤。您也可以選擇其他應用程式名稱，並以新名稱對其進行部署。

   d. 若要在部署之前檢驗檔案的結構和內容，請選取 [檢驗器] 核取方塊。大型應用程式的檢驗會很費時。如果懷疑檔案已毀壞或爲不可攜式，請檢驗檔案。

e. 選擇高可用性設定。

若要啓用模組的高可用性，請選取 [可用性] 核取方塊。如果啓用了模組的可用性，則必須也在所有更高層級 (指配置和 Web 容器或 EJB 容器) 啓用可用性。

f. 選擇要將模組部署到的目標。

從可用目標的清單中選擇目標，並按一下 [新增]。目標可以是叢集或獨立伺服器實例。如果不選取目標，模組將被部署到預設伺服器實例 `server`。

如果您要重新部署，請勿選取目標。此時您所作的任何選取都將被忽略。參考已部署的模組的所有目標叢集或獨立伺服器實例都將自動參考新的、重新部署的模組 (如果已啓用叢集或獨立實例的動態重新配置)。如需有關如何在不中斷服務的情況下重新部署模組的更多資訊，請參閱第 79 頁的「關於捲動升級」。

g. 選擇是否產生 RMI 存根。

如果選擇產生 RMI 存根，將產生靜態 RMI-IIOP 存根並將其新增到 `client.jar`。

7. 按一下 [確定] 以部署模組。

等效的 `asadmin` 指令爲：`deploy`

部署連接器模組

連接器也稱爲資源介面，它封裝在一種稱爲 RAR 檔案的歸檔檔案中。

若要部署 (安裝) 連接器模組，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。

2. 選取 [連接器模組] 節點。

3. 在 [連接器模組] 頁面中，按一下 [部署]。

4. 在 [部署] 頁面中，指定要部署的 RAR 檔案的位置。

伺服器是執行應用程式伺服器和網域管理伺服器的主機。用戶端是可以用於透過瀏覽器來檢視管理主控台的主機。

a. 如果檔案位於用戶端上或可以從用戶端存取，則按一下單選按鈕以指定套裝軟體檔案以上傳到 Application Server。

按一下 [瀏覽] 以找到檔案，或鍵入檔案的完整路徑。

b. 如果檔案位於伺服器上，或者要從展開的目錄部署未封裝的模組，則按一下單選按鈕以指定必須可以從伺服器上存取的套裝軟體檔案或目錄路徑。

鍵入檔案或目錄的完整路徑名稱。從展開的目錄部署適用於進階開發者，但不建議在生產環境下進行此操作。

5. 按一下 [下一步] 以顯示 [部署連接器模組] 頁面。

6. 在 [部署連接器模組] 頁面中，指定模組的設定。

a. 在 [應用程式名稱] 欄位中，既可以保留預設名稱 (即檔案名稱的字首)，也可以鍵入其他名稱。(如果您選擇上傳檔案，將顯示預設名稱。) 應用程式名稱必須唯一。

b. 在 [執行緒池 ID] 欄位中，爲要部署的資源介面指定執行緒池。

依預設，Sun Java System Application Server 處理其預設執行緒池中所有資源介面的工作請求。使用該欄位可以關聯特定使用者建立的執行緒池以處理資源介面的工作請求。

c. 依預設，部署模組之後即可使用該模組。若要在部署之後停用模組以使其不可用，請選取 [已停用] 單選按鈕。

啓用或停用連接器模組時，您也同時啓用或停用了指向該模組的連接器資源和連線池。

d. 如果已部署模組，請選取 [重新部署] 核取方塊以對其進行重新部署；否則將顯示錯誤。您也可以選擇其他應用程式名稱，並以新名稱對其進行部署。

e. 若要在部署之前檢驗檔案的結構和內容，請選取 [檢驗器] 核取方塊。大型應用程式的檢驗通常會很費時。如果懷疑檔案已毀壞或爲不可攜式，請檢驗檔案。

f. 如果資源介面被指定了附加特性，將顯示這些特性。

使用此表可以修改這些特性的預設值。

g. 選擇要將模組部署到的目標。

從可用目標的清單中選擇目標，並按一下 [新增]。目標可以是叢集或獨立伺服器實例。如果不選取目標，模組將被部署到預設伺服器實例 `server`。

如果您要重新部署，請勿選取目標。此時您所作的任何選取都將被忽略。參考已部署的模組的所有目標叢集或獨立伺服器實例都將自動參考新的、重新部署的模組 (如果已啓用叢集或獨立實例的動態重新配置)。如需有關如何在不中斷服務的情況下重新部署模組的更多資訊，請參閱第 79 頁的「關於捲動升級」。

7. 按一下 [確定] 以部署模組。

等效的 `asadmin` 指令爲：`deploy`

建立生命週期模組

生命週期模組在伺服器生命週期中有一個或多個事件觸發時會執行作業。這些伺服器事件包含：

- 初始化
- 啓動
- 服務請求準備就緒
- 關閉

生命週期模組不符合 J2EE 規格，但該模組是 Sun Java System Application Server 的增強功能。

若要建立生命週期模組，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。
2. 選取 [生命週期模組] 節點。
3. 在 [生命週期模組] 頁面中，按一下 [新建]。
4. 在 [建立生命週期模組] 頁面中，指定以下設定：
   a. 在 [名稱] 欄位中，鍵入指定模組功能的名稱。
   b. 在 [類別名稱] 欄位中，鍵入生命週期模組的類別檔案的完全合格名稱。
   c. 如果包含生命週期的 JAR 檔案位於伺服器的類別路徑中，則請將 [類別名稱] 欄位保留爲空。否則，鍵入完全合格的路徑。

      如果不指定類別路徑，則必須在 *application_server_home*/domains/*domain*/applications/lifecycle-module/*module_name* 中解壓縮類別。如果指定類別路徑，則無需其他操作。
   d. 在 [載入次序] 欄位中，鍵入大於 100 小於作業系統的 `MAXINT` 值的整數。

      該整數將確定伺服器啓動時載入生命週期模組的次序。具有較小整數的模組可以更快地載入。
   e. 啓動伺服器時，伺服器將載入已部署的生命週期模組。依預設，如果載入失敗，伺服器仍將繼續進行啓動作業。若要在載入失敗時防止伺服器啓動，請選取 [載入失敗時] 核取方塊。
   f. 依預設，部署模組之後即可使用該模組。若要在部署之後停用模組以使其不可用，請選取 [已停用] 單選按鈕。

g. 選擇要將模組部署到的目標。

從可用目標的清單中選擇目標，並按一下 [新增]。目標可以是叢集或獨立伺服器實例。如果不選取目標，模組將被部署到預設伺服器實例 `server`。

如果您要重新部署，請勿選取目標。此時您所作的任何選取都將被忽略。參考已部署的模組的所有目標叢集或獨立伺服器實例都將自動參考新的、重新部署的模組 (如果已啓用叢集或獨立實例的動態重新配置)。如需有關如何在不中斷服務的情況下重新部署模組的更多資訊，請參閱第 79 頁的「關於捲動升級」。

5. 按一下 [確定]。

等效的 `asadmin` 指令爲：`create-lifecycle-module`

部署應用程式用戶端模組

應用程式用戶端模組也稱爲 J2EE 應用程式用戶端 JAR 檔案，它包含用戶端的伺服器端常式。

若要部署 (安裝) 應用程式用戶端模組，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。

2. 選取 [應用程式用戶端模組] 節點。

3. 在 [應用程式用戶端模組] 頁面中，按一下 [部署]。

4. 在 [部署] 頁面中，指定要部署的 JAR 檔案的位置。

伺服器是執行應用程式伺服器和網域管理伺服器的主機。用戶端是可以用於透過瀏覽器來檢視管理主控台的主機。

a. 如果檔案位於用戶端上或可以從用戶端存取，則按一下單選按鈕以指定套裝軟體檔案以上傳到 Application Server。

按一下 [瀏覽] 以找到檔案，或鍵入檔案的完整路徑。

b. 如果檔案位於伺服器上，或者要從展開的目錄部署未封裝的模組，則按一下單選按鈕以指定必須可以從伺服器上存取的套裝軟體檔案或目錄路徑。

鍵入檔案或目錄的完整路徑名稱。從展開的目錄部署適用於進階開發者，但不建議在生產環境下進行此操作。

5. 按一下 [下一步] 以顯示 [部署應用程式用戶端模組] 頁面。

6. 在 [部署應用程式用戶端模組] 頁面中，請指定模組的設定。

a. 在 [應用程式名稱] 欄位中，既可以保留預設名稱 (即檔案名稱的字首)，也可以鍵入其他名稱。(如果您選擇上傳檔案，將顯示預設名稱。) 應用程式名稱必須唯一。

b. 如果已部署模組，請選取 [重新部署] 核取方塊以對其進行重新部署；否則將顯示錯誤。您也可以選擇其他應用程式名稱，並以新名稱對其進行部署。

c. 若要在部署之前檢驗檔案的結構和內容，請選取 [檢驗器] 核取方塊。大型應用程式的檢驗會很費時。如果懷疑檔案已毀壞或爲不可攜式，請檢驗檔案。

d. 選擇要將模組部署到的目標。

從可用目標的清單中選擇目標，並按一下 [新增]。目標可以是叢集或獨立伺服器實例。如果不選取目標，模組將被部署到預設伺服器實例 `server`。

如果您要重新部署，請勿選取目標。此時您所作的任何選取都將被忽略。參考已部署的模組的所有目標叢集或獨立伺服器實例都將自動參考新的、重新部署的模組 (如果已啓用叢集或獨立實例的動態重新配置)。如需有關如何在不中斷服務的情況下重新部署模組的更多資訊，請參閱第 79 頁的「關於捲動升級」。

e. 選擇是否產生 RMI 存根。

如果選擇產生 RMI 存根，將產生靜態 RMI-IIOP 存根並將其新增到 `client.jar`。

7. 按一下 [確定] 以部署模組。

對於用戶端常式：

- 通常，應用程式供應商會發行包含用戶端常式的 JAR 檔案。
- 應用程式供應商透過指定 `asadmin deploy` 指令的 `--retrieve` 選項來獲取用戶端存根。

等效的 `asadmin` 指令爲：`deploy`

有關列示、取消部署以及啟用應用程式的管理主控台作業

- 「列示已部署的應用程式」
- 「列示子元件」
- 「檢視已部署的應用程式的模組描述元」
- 「取消部署應用程式」
- 「啓用和停用應用程式」
- 「啓用和停用動態重新載入」

列示已部署的應用程式

若要列示已部署的應用程式，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。
2. 展開應用程式或模組類型的節點。

若要檢視已部署的應用程式或模組的詳細資訊，請執行以下操作之一：

- 在樹形元件中，選取應用程式或模組的節點。
- 在頁面中選取 [應用程式名稱] 欄中的項目。

等效的 asadmin 指令爲：`list-components`

列示子元件

企業應用程式、Web 應用程式、EJB 模組和連接器模組都包含子元件。例如，Web 應用程式可能包含一個或多個 Servlet。

若要列示應用程式或模組的子元件，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。
2. 展開要檢視其描述元的應用程式或模組類型的節點。
3. 選取已部署的應用程式或模組的節點。
4. 在 [應用程式或模組] 頁面中，檢視 [子元件] 表的內容。

等效的 asadmin 指令爲：`list-sub-components`

檢視已部署的應用程式的模組描述元

對於企業應用程式、Web 應用程式、EJB 模組、連接器模組和應用程式用戶端模組，您可以檢視模組部署描述元。

1. 在樹形元件中，展開 [應用程式] 節點。
2. 選取要檢視其描述元的應用程式或模組類型的節點。
3. 選取已部署的應用程式或模組的節點。
4. 選取 [描述元] 標籤。
5. 若要查看描述元檔案的文字，請按一下檔案名稱。

   頁面將顯示檔案內容。此資訊是唯讀的。

取消部署應用程式

取消部署應用程式或模組將從網域中將其解除安裝並移除所有實例對它的參考。

若要取消部署應用程式或模組，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。
2. 選取要取消部署的應用程式或模組的類型的節點。
3. 在列示已部署的應用程式的表中，選取要取消部署的應用程式或模組的核取方塊。
4. 按一下 [取消部署]。

等效的 `asadmin` 指令爲：`undeploy`

啟用和停用應用程式

如果啓用了已部署的應用程式或模組，則可以透過用戶端對其進行存取。如果將其停用，它將仍然處於部署狀態，但不能透過用戶端對其進行存取。依預設，在部署應用程式或模組時，由於依預設選取了 [在所有目標上啓用] 單選按鈕，因此將啓用該應用程式或模組。

若要啓用已部署的應用程式或模組，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。
2. 展開應用程式類型的節點。

3. 選取已部署的應用程式或模組旁邊的核取方塊。
4. 按一下 [啓用] 或 [停用]。

   這些按鈕用於在所有目標上啓用或停用應用程式。

若要在單一目標上啓用應用程式，請執行以下步驟：

1. 在樹形元件中，展開 [應用程式] 節點。
2. 展開應用程式類型的節點。
3. 選取應用程式的節點。
4. 按一下 [目標] 標籤。
5. 選取已部署的應用程式或模組旁邊的核取方塊。
6. 按一下 [啓用] 或 [停用]。

等效的 asadmin 指令爲：enable 和 disable

管理應用程式目標

部署應用程式或模組後，可透過管理目標來管理參考該應用程式或模組的伺服器實例和叢集。

1. 在樹形元件中，展開 [應用程式] 節點。
2. 展開應用程式類型的節點。
3. 選取已部署的應用程式的節點。
4. 選取 [目標] 標籤。
5. 若要在特定目標實例或叢集上啓用或停用應用程式，請按一下目標旁邊的核取方塊並按一下 [啓用] 或 [停用]。
6. 若要新增或刪除應用程式的目標，則請選擇 [管理目標]。
7. 新增或移除目標並按一下 [確定]。

   此時，已修訂的目標清單中將包含該應用程式。

等效的 asadmin 指令爲：create-application-ref 和 delete-application-ref。

部署在其他虛擬伺服器上

將應用程式或模組部署到目標伺服器實例或叢集之後，可以將其與其他虛擬伺服器關聯。

1. 在已部署的應用程式或模組的 [目標] 頁面中，按一下目標旁邊的 [管理虛擬伺服器] 連結。
2. 在可用虛擬伺服器的清單中，新增或移除虛擬伺服器目標。
3. 按一下 [確定]。

重新部署到多個目標

如果將應用程式部署到多個目標 (獨立伺服器實例或叢集)，則可以透過兩種方式來重新部署到多個目標。使用以下方法之一可以確定參考應用程式的所有伺服器實例接收到最新版本。

開發環境

在開發環境中，只重新部署應用程式。應用程式將被重新部署到網域，並且參考它的所有目標將自動接收到新的版本 (如果已啓用目標伺服器實例的動態重新配置)。依預設，將啓用動態重新配置。如果未啓用伺服器實例的動態重新配置，將繼續使用舊版本，直至重新啓動伺服器實例。

生產環境

在生產環境中，請按第 79 頁的 「關於捲動升級」中所述的步驟進行操作。

啟用和停用動態重新載入

如果啓用了動態重新載入，伺服器將定期檢查已部署的應用程式中的變更並自動重新載入包含變更的應用程式。變更將通過您手動建立的名爲 `.reload` 的檔案的日期變更來表明。應用程式必須安裝在 *server_root*`/domain/domain1/applications/`*j2ee-module_or_j2ee-apps*`/` *app_or_module_name* 中。

例如： `AppServer/domain/domain1/applications/j2ee-module/webapps-simple`

動態重新載入在開發環境中非常有用，因爲它能快速測試程式碼變更。但在生產環境中，動態重新載入可能會使效能降低。

備註 動態重新載入僅適用於預設伺服器實例。

動態重新載入旨在用於開發環境。它與階段作業持續性 (一種生產環境功能) 不相容。如果啓用了動態部署，請勿啓用階段作業持續性。

若要配置動態重新載入，請執行以下步驟：

1. 在樹形元件中，展開 [獨立實例] 節點。
2. 按一下 [server] (管理伺服器)。
3. 按一下 [進階]。
4. 在 [應用程式配置] 頁面中，配置以下項目：
   - 重新載入：使用 [已啓用] 核取方塊來啓用或停用動態重新載入。
   - 重新載入輪詢間隔：指定伺服器檢查已部署的應用程式中的變更的頻率。
   - 管理階段作業逾時：指定管理階段作業逾時且必須重新登入之前的時間。

將系統配置爲使用動態重新載入之後，請爲每個要動態重新載入的應用程式建立一個名爲 .reload 的檔案並將其置於應用程式的目錄中。此檔案沒有任何內容。變更應用程式時，將變更此檔案的日期 (例如，使用 UNIX `touch` 指令)，並且自動重新載入變更。

適用於開發者的部署方法

- 「使用自動部署」
- 「部署目錄中未封裝的應用程式」
- 「使用 deploytool 公用程式」
- 「使用部署規劃」

使用自動部署

自動部署功能使您能夠透過將預先封裝的應用程式或模組複製到 *domain_root_dir*/*domain_dir*/`autodeploy` 目錄來部署該應用程式或模組。

例如，將名爲 `hello.war` 的檔案複製到 *domain_root_dir*/`domain1/autodeploy` 目錄。若要取消部署應用程式，請從 `autodeploy` 目錄中移除 `hello.war` 檔案。

您也可以使用管理主控台或 asadmin 工具來取消部署應用程式。在這種情況下，歸檔檔案將保留。

自動部署功能旨在用於開發環境。它與階段作業持續性 (一種生產環境功能) 不相容。如果啓用了動態部署，請勿啓用階段作業持續性

備註 自動部署僅適用於預設伺服器實例。

若要配置自動部署功能，請執行以下步驟：

1. 在樹形元件中，展開 [獨立實例] 節點。
2. 按一下 [server] (管理伺服器)。
3. 按一下 [進階]。
4. 在 [應用程式配置] 頁面中，配置以下項目：
   a. 透過選取或取消選取 [已啓用] 核取方塊來啓用或停用自動部署。
   b. 在 [自動部署輪詢間隔] 欄位中，指定伺服器檢查自動部署目錄中的應用程式檔案或模組檔案的頻率。變更輪詢間隔不會影響部署應用程式或模組所需的時間。
   c. 在 [自動部署目錄] 中，如果指定建立應用程式的目錄，則不必將檔案複製到預設自動部署目錄中。

      預設目錄是伺服器實例的根目錄中名爲 `autodeploy` 的目錄。依預設，可使用變數自動變更多個伺服器實例的目錄。如需有關這些變數的更多資訊，請參閱 「設定網域屬性」。
   d. 若要在部署之前執行檢驗器，請選取 [已啓用檢驗器] 核取方塊。檢驗器將檢查檔案的結構和內容。大型應用程式的檢驗通常會很費時。
   e. 若要預編譯 JSP 頁面，請選取 [JSP] 核取方塊。如果未選取此核取方塊，則首次存取 JSP 頁面時會在執行階段編譯這些頁面。由於編譯通常很費時，因此在生產環境中請選取此核取方塊。

部署目錄中未封裝的應用程式

此功能適用於進階開發者。

使用目錄部署僅部署到預設伺服器實例 (server)。您不能使用它來部署到叢集或獨立伺服器實例。

包含未封裝的應用程式或模組的目錄有時稱爲展開的目錄。目錄的內容必須與對應的 J2EE 歸檔檔案的內容匹配。例如，如果部署某一目錄中的 Web 應用程式，則該目錄的內容必須與對應的 WAR 檔案的內容相同。如需有關必需的目錄內容的資訊，請參閱相應的規格。

您可以直接在展開的目錄中變更部署描述元檔案。

如果您的環境配置爲使用動態重新載入，則還可以從目錄中動態重新載入已部署的應用程式。如需更多資訊，請參閱第 116 頁的「啓用和停用動態重新載入」。

若要部署目錄中未封裝的應用程式，請執行以下步驟：

1. 在管理主控台中開始部署程序。請參閱 第 104 頁的「部署 Web 應用程式」。
2. 在 [部署] 頁面中，指定以下內容：
   a. 按一下單選按鈕以指定必須可以從伺服器上存取的套裝軟體檔案或目錄路徑。
   b. 在 [檔案或目錄] 欄位中，輸入展開的目錄的名稱。

等效的 `asadmin` 指令爲：`deploydir`

使用 deploytool 公用程式

爲軟體開發者設計的 `deploytool` 公用程式可以封裝並部署 J2EE 應用程式和模組。如需有關如何使用 `deploytool` 的說明，請參閱「The J2EE 1.4 Tutorial」。

使用部署規劃

此功能適用於進階開發者。

部署規劃是指僅包含特定於 Application Server 的部署描述元的 JAR 檔案。有關這些部署描述元 (例如 `sun-application.xml`) 的說明，請參閱「Application Server Developer's Guide」。部署規劃是 JSR 88: J2EE Application Deployment 的實作的一部分。使用部署規劃可以部署不包含特定於 Application Server 的部署描述元的應用程式或模組。

若要使用部署規劃進行部署，請指定 asadmin deploy 指令中的 --deploymentplan 選項。例如，以下指令將根據 mydeployplan.jar 檔案中指定的規劃來部署 myrosterapp.ear 檔案中的企業應用程式。

```
$ asadmin deploy --user admin ---deploymentplan mydeployplan.jar
myrosterapp.ear
```

在企業應用程式 (EAR) 的部署規劃檔案中，sun-application.xml 檔案位於根目錄下。根據以下語法來儲存每個模組的部署描述元：*module-name.sun-dd-name*，其中的 *sun-dd-name* 取決於模組類型。如果模組包括 CMP 對映檔案，則該檔案命名爲 *module-name*.sun-cmp-mappings.xml。.dbschema 檔案儲存在根層級目錄下，並用磅符號 (#) 替代每個正斜線字元 (/)。下面列示的內容顯示了企業應用程式 (EAR) 的部署規劃檔案的結構。

```
$ jar -tvf mydeployplan.jar
420 Thu Mar 13 15:37:48 PST 2003 sun-application.xml
370 Thu Mar 13 15:37:48 PST 2003 RosterClient.war.sun-web.xml
418 Thu Mar 13 15:37:48 PST 2003 roster-ac.jar.sun-application-client.xml
1281 Thu Mar 13 15:37:48 PST 2003 roster-ejb.jar.sun-ejb-jar.xml
2317 Thu Mar 13 15:37:48 PST 2003 team-ejb.jar.sun-ejb-jar.xml
3432 Thu Mar 13 15:37:48 PST 2003 team-ejb.jar.sun-cmp-mappings.xml
84805 Thu Mar 13 15:37:48 PST 2003 team-ejb.jar.RosterSchema.dbschema
```

在 Web 應用程式或模組檔案的部署規劃中，特定於 Application Server 的部署描述元位於根層級目錄下。如果獨立 EJB 模組包括 CMP Bean，則部署規劃包括位於根層級目錄的 sun-cmp-mappings.xml 和 .dbschema 檔案。在下面列示的內容中，部署規劃介紹了 CMP Bean。

```
$ jar r -tvf myotherplan.jar
3603 Thu Mar 13 15:24:20 PST 2003 sun-ejb-jar.xml
3432 Thu Mar 13 15:24:20 PST 2003 sun-cmp-mappings.xml
84805 Thu Mar 13 15:24:20 PST 2003 RosterSchema.dbschema
```

第 6 章

JDBC 資源

本章說明如何配置存取資料庫的應用程式所要求使用的 JDBC 資源。它包含以下小節：

- 關於 JDBC 資源
- 設定資料庫存取
- 關於 JDBC 連線池
- 關於 JDBC 資源
- 關於持續性管理程式資源

關於 JDBC 資源

- JDBC 資源
- JDBC 連線池
- JDBC 資源和連線池如何協同工作

JDBC 資源

爲了儲存、組織和擷取資料，大多數應用程式均使用關聯式資料庫。J2EE 應用程式透過 JDBC API 存取關聯式資料庫。

JDBC 資源 (資料源) 爲應用程式提供了連線至資料庫的方法。通常，管理員要爲部署在網域中的應用程式存取的每個資料庫建立 JDBC 資源。(但是，可以爲一個資料庫建立多個 JDBC 資源。)

若要建立 JDBC 資源，請指定識別資源的唯一 JNDI 名稱。(請參閱 「JNDI 名稱和資源」一節。) JDBC 資源的 JNDI 名稱應在 `java:comp/env/jdbc` 子環境中。例如，薪水帳單資料庫資源的 JNDI 名稱可以爲 `java:comp/env/jdbc/payrolldb`。由於所有資源 JNDI 名稱位於 `java:comp/env` 子環境中，因此在管理主控台中指定 JDBC 資源的 JNDI 名稱時僅需輸入 `jdbc/`*name*。例如，對於薪水帳單資料庫，可以指定 `jdbc/payrolldb`。

JDBC 連線池

若要建立 JDBC 資源，請指定與其關聯的連線池。多個 JDBC 資源可指定單一連線池。

JDBC 連線池是針對特定資料庫的一組可重複使用的連線。由於每建立一個新的實體連線都會耗費時間，因此伺服器維護了可用連線池以提高效能。當應用程式請求連線時，它可以從池中取得一個連線。應用程式關閉連線時，連線將傳回池中。

連線池的特性可能會隨資料庫供應商的不同而有所不同。有一些特性是通用的，例如資料庫名稱 (URL)、使用者名稱和密碼。

JDBC 資源和連線池如何協同工作

爲了儲存、組織和擷取資料，大多數應用程式均使用關聯式資料庫。J2EE 應用程式透過 JDBC API 存取關聯式資料庫。應用程式存取資料庫之前，必須先取得連線。

以下是在執行階段應用程式連線至資料庫時所發生的情況：

1. 應用程式透過 JNDI API 進行呼叫以獲取與資料庫關聯的 JDBC 資源 (資料源)。

   如果給定了資源的 JNDI 名稱，命名和目錄服務將查找 JDBC 資源。每個 JDBC 資源指定一個連線池。

2. 通過 JDBC 資源，應用程式獲得一個資料庫連線。

   應用程式伺服器秘密地從與該資料庫相對應的連線池中擷取實體連線。池定義資料庫名稱 (URL)、使用者名稱和密碼等連線屬性。

3. 由於已將應用程式連線至資料庫，因此該應用程式可以讀取和修改資料庫中的資料以及將資料新增到資料庫中。

   應用程式透過對 JDBC API 進行呼叫來存取資料庫。JDBC 驅動程式將應用程式的 JDBC 呼叫翻譯爲資料庫伺服器的協定。

4. 存取資料庫完成之後，應用程式將關閉該連線。

   應用程式伺服器將連線傳回連線池。連線傳回連線池之後，下一個應用程式就可以使用該連線。

設定資料庫存取

- 設定資料庫存取的一般步驟
- 整合 JDBC 驅動程式

設定資料庫存取的一般步驟

1. 安裝支援的資料庫產品。如需有關 Application Server 支援的資料庫產品的清單，請參閱「更多資訊」一節中指向「**版本說明**」的連結。
2. 安裝適用於該資料庫產品的 JDBC 驅動程式。
3. 使網域的伺服器實例可以存取此驅動程式的 JAR 檔案。請參閱整合 JDBC 驅動程式。
4. 建立資料庫。通常，應用程式供應程式提供了用於建立和填入資料庫的程序檔。
5. 爲資料庫建立連線池。請參閱「建立 JDBC 連線池」。
6. 建立指向連線池的 JDBC 資源。請參閱「建立 JDBC 資源」。

整合 JDBC 驅動程式

JDBC 驅動程式將應用程式的 JDBC 呼叫翻譯爲資料庫伺服器的協定。若要將 JDBC 驅動程式整合到管理網域中，請執行以下操作之一：

- 使通用類別載入器可以存取該驅動程式。
  - 將驅動程式的 JAR 檔案和 ZIP 檔案複製到 *install_dir*`/domains/`*domain_dir*`/lib` 目錄或將驅動程式的類別檔案複製到 *install_dir*`/domains/`*domain_dir*`/lib/ext` 目錄中。
  - 重新啓動該網域。
- 使系統類別載入器可以存取該驅動程式。
  - 在管理主控台的樹檢視中，選取 [配置]。

- 選取所需的配置 (例如 default-config)。
- 選取 [JVM 設定]。
- 在 [JVM 設定] 頁面中，按一下 [路徑設定] 標籤。
- 在 [類別路徑後綴] 欄位中，輸入驅動程式 JAR 檔案的完全合格路徑名稱。
- 按一下 [儲存]。
- 重新啓動伺服器。

關於 JDBC 連線池

- 建立 JDBC 連線池
- 編輯 JDBC 連線池
- 刪除 JDBC 連線池

建立 JDBC 連線池

JDBC 連線池是針對特定資料庫的一組可重複使用的連線。使用管理主控台建立池時，管理員實際上是在定義與特定資料庫的連線的各個方面。

建立池之前，您必須首先安裝並整合 JDBC 驅動程式。

建置 [建立連線池] 頁面時，必須輸入特定於 JDBC 驅動程式和資料庫供應商的特定資料。繼續操作之前，請先收集以下資訊：

- 資料庫供應商名稱
- 資源類型，例如 `javax.sql.DataSource` (僅限於本機作業事件) 和 `javax.sql.XADataSource` (全域作業事件)
- 資料源類別名稱
- 必需的特性，例如資料庫名稱 (URL)、使用者名稱和密碼

若要建立 JDBC 連線池，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點。
2. 在 [資源] 節點下，展開 [JDBC] 節點。
3. 在 [JDBC] 節點下，選取 [連線池] 節點。

4. 在 [連線池] 頁面中，按一下 [新建]。
5. 在 [建立連線池] 的第一個頁面中，指定以下一般設定：
   a. 在 [名稱] 欄位中，輸入池的邏輯名稱。

      建立 JDBC 資源時將指定此名稱。
   b. 從 [資源類型] 組合方塊中選取一個項目。
   c. 從 [資料庫供應商] 組合方塊中選取一個項目。
6. 按一下 [下一步]。
7. 在 [建立連線池] 的第二個頁面中，爲 [DataSource 類別名稱] 欄位指定值。

   如果 JDBC 驅動程式具有與在上一頁中指定的資源類型和資料庫供應商相應的 DataSource 類別，則系統會提供 [DataSource 類別名稱] 欄位的值。
8. 按一下 [下一步]。
9. 在 [建立連線池] 的第三個也就是最後一個頁面中，執行以下作業：
   a. 在 [一般設定] 區段中，檢驗各個值是否正確。
   b. 對於 [池設定]、[連線驗證] 和 [作業事件隔絕] 區段中的欄位，保留預設值。

      您可以在以後非常方便地變更這些設定。請參閱 「編輯 JDBC 連線池」。
   c. 在 [附加特性] 表中，新增必需的特性，例如資料庫名稱 (URL)、使用者名稱和密碼。
10. 按一下 [完成]。

等效的 `asadmin` 指令爲：`create-jdbc-connection-pool`

編輯 JDBC 連線池

[編輯 JDBC 連線池] 頁面爲您提供了變更現有池的所有設定 (池的名稱除外) 的方法。

若要存取 [編輯 JDBC 連線池] 頁面，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點。
2. 在 [資源] 節點下，展開 [JDBC] 節點。
3. 在 [JDBC] 節點下，展開 [連線池] 節點。

4. 選擇要編輯的池的節點。
5. 在 [編輯 JDBC 連線池] 頁面中進行必要的變更。

   請參閱以下各節，以取得有關可以變更的設定的說明。
6. 按一下 [儲存]。

一般設定

一般設定的值取決於安裝的特定 JDBC 驅動程式。這些設定是 Java 程式設計語言中的類別名稱或介面名稱。

表 6-1 JDBC 連線池的一般設定

參數	描述
DataSource 類別名稱	實作 DataSource/ConnectionPoolDataSource/XADataSource API 的供應商特定的類別名稱。該類別位於 JDBC 驅動程式中。
資源類型	選項包括 javax.sql.DataSource (僅限於本機作業事件)、javax.sql.XADataSource (全域作業事件) 和 java.sql.ConnectionPoolDataSource (本機作業事件，效能可能會改善)。

池設定

一組實體資料庫連線保存在池中。應用程式請求連線時，將從池中移除該連線；而應用程式釋放該連線之後，連線將傳回到池中。

表 6-2 JDBC 連線池的池設定

參數	描述
池的初始大小和最小大小	池中連線的最小數目。該值還確定了首次建立池或應用程式伺服器啟動時置於池中的連線的數目。
池的最大大小	池中連線的最大數目。
池設定大小數量	當池向最小池大小收縮時，將成批調整大小。此值確定批次中的連線數目。將該值設置過大會延遲連線循環；而將該值設置過小則會導致效率降低。
閒置逾時	連線在池中保持閒置狀態的最長時間 (以秒為單位)。一旦超過此時間，即從池中移除該連線。
最長等待時間	請求連線的應用程式在達到連線逾時之前等待的時間。由於預設等待時間過長，應用程式可能會出現無限期當機的情況。

連線驗證

選擇性地，應用程式伺服器可以在將連線傳送給應用程式之前驗證連線。如果由於網路出現故障或資料庫伺服器當機造成資料庫不可用，此驗證將允許應用程式伺服器自動重新建立資料庫連線。連線驗證會耗用額外的時間，並會略微降低效能。

表 6-3 JDBC 連線池的連線驗證設定

參數	描述
連線驗證	選取 [需要] 核取方塊以啓用連線驗證。
驗證方法	應用程式伺服器可以使用三種方法來驗證資料庫連線：auto-commit、metadata 和 table。 • auto-commit 和 metadata — 應用程式伺服器透過呼叫 con.getAutoCommit () 方法和 con.getMetaData () 方法來驗證連線。但是，由於許多 JDBC 驅動程式快取了這些呼叫的結果，因此這兩種方法無法始終提供可靠的驗證。請與驅動程式供應商進行核實，以確定這些呼叫是否被快取。 • table — 應用程式將查詢指定的資料庫表格。此表必須存在並且可以存取，但不要求表的列數。請勿使用包含許多列的現有表或經常存取的表。
表格名稱	如果從 [驗證方法] 組合方塊中選取了表，請在此處指定資料庫表格的名稱。
一旦失敗	選取標記為 [關閉所有連線] 的核取方塊之後，如果單一連線失敗，應用程式伺服器將關閉池中的所有連線，然後重新建立這些連線。如果未選取此核取方塊，則只有要使用各個連線時才會重新建立這些連線。

作業事件隔絕

由於許多使用者通常可以同步運作地存取一個資料庫，因此可能出現一個作業事件在更新資料而另一個作業事件嘗試讀取同一資料的情況。作業事件的隔絕層級定義了正在更新的資料對於其他作業事件的可視程度。如需有關隔絕層級的詳細資訊，請參閱資料庫供應商的文件。

表 6-4 JDBC 連線池的作業事件隔絕設定

參數	描述
作業事件隔絕	使您可以為該池的連線選取作業事件隔絕層級。如果不指定此參數，連線將使用 JDBC 驅動程式提供的預設隔絕層級進行作業。

表 6-4 JDBC 連線池的作業事件隔絕設定

參數	描述
受保證隔絕層級	該項僅在指定了隔絕層級的情況下才適用。如果選取 [受保證] 核取方塊，則從池中獲取的所有連線都具有相同的隔絕層級。例如，如果上次使用連線時程式化 (使用 `con.setTransactionIsolation`) 變更了連線的隔絕層級，這種機制會將狀態變更回指定的隔絕層級。

特性

在 [附加特性] 表中，可以指定資料庫名稱 (URL)、使用者名稱和密碼等特性。由於隨資料庫供應商的不同，特性也會有所不同，因此請查閱供應商文件以取得詳細資訊。

驗證連線池設定

若要驗證連線池設定，請執行以下步驟：

1. 啓動資料庫伺服器。
2. 按一下 [Ping]。

   管理主控台將嘗試連線到資料庫。如果顯示錯誤訊息，請檢查資料庫伺服器是否已重新啓動。

刪除 JDBC 連線池

1. 在樹形元件中，展開 [資源] 節點。
2. 在 [資源] 節點下，展開 [JDBC] 節點。
3. 在 [JDBC] 節點下，選取 [連線池] 節點。
4. 在 [連線池] 頁面中，選取要刪除的池的核取方塊。
5. 按一下 [刪除]。

等效的 `asadmin` 指令爲：`delete-jdbc-connection-pool`

關於 JDBC 資源

- 建立 JDBC 資源
- 編輯 JDBC 資源
- 刪除 JDBC 資源

建立 JDBC 資源

JDBC 資源 (資料源) 爲應用程式提供連線至資料庫的方法。建立 JDBC 資源之前，先建立 JDBC 連線池。

若要建立 JDBC 資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點。
2. 在 [資源] 節點下，展開 [JDBC] 節點。
3. 在 [JDBC] 節點下，選取 [JDBC 資源] 節點。
4. 在 [JDBC 資源] 頁面中，按一下 [新建]。
5. 在 [建立 JDBC 資源] 頁面中，指定資源的設定：
   a. 在 [JNDI 名稱] 欄位中，鍵入唯一的名稱。按照慣例，該名稱應以 `jdbc/` 字串開頭。例如： `jdbc/payrolldb`。請不要忘記正斜線。
   b. 從 [池名稱] 組合方塊中選擇要與新的 JDBC 資源關聯的連線池。
   c. 依預設，建立資源之後立即可以使用資源 (已啓用)。如果要使資源不可用，請取消選取 [啓用] 核取方塊。
   d. 在 [描述] 欄位中，鍵入資源的簡短描述。
   e. 在 [目標] 區段，指定該資源在其中可用的目標 (叢集和獨立伺服器實例)。在左側選取所需的目標，然後按一下 [增加] 以將其增加到選取目標的清單中。
6. 按一下 [確定]。

等效的 `asadmin` 指令爲：`create-jdbc-resource`

編輯 JDBC 資源

1. 在樹形元件中，展開 [資源] 節點。
2. 在 [資源] 節點下，展開 [JDBC] 節點。
3. 在 [JDBC] 節點下，展開 [JDBC 資源] 節點。
4. 選取要編輯的 JDBC 資源的節點。
5. 在 [編輯 JDBC 資源] 頁面中，可以執行以下作業：
   a. 從 [池名稱] 組合方塊中選取其他連線池。
   b. 在 [描述] 欄位中，變更資源的簡短描述。
   c. 選取或取消選取核取方塊以啓用或停用資源。
   d. 選取 [目標] 標籤，以變更具有可用資源的目標 (叢集和獨立伺服器實例)。

      選取清單中的現有目標的核取方塊，然後按一下 [啓用] 以啓用該目標的資源，或者按一下 [停用] 以停用該目標的資源。

      按一下 [管理目標] 可以在清單中增加或移除目標。在 [管理目標] 頁面中，從左側的 [可用] 清單中選取所需的目標，然後按一下 [增加] 以將其增加到選取目標的清單中。按一下 [移除] 可以從 [已選取] 清單中移除目標。

      按一下 [確定] 以儲存對可用目標的變更。
6. 按一下 [儲存] 以套用編輯。

刪除 JDBC 資源

1. 在樹形元件中，展開 [資源] 節點。
2. 在 [資源] 節點下，展開 [JDBC] 節點。
3. 在 [JDBC] 節點下，選取 [JDBC 資源] 節點。
4. 在 [JDBC 資源] 頁面中，選取要刪除的資源的核取方塊。
5. 按一下 [刪除]。

- 啓用和停用 JDBC 資源

啟用和停用 JDBC 資源

1. 在樹形元件中，展開 [JDBC 資源] 節點或展開 [獨立實例] 以選取 [伺服器實例節點資源] 標籤。
2. 在 [資源] 頁面中，選取要啓用或停用的資源的核取方塊。
3. 按一下 [啓用] 或 [停用]。

關於持續性管理程式資源

- 建立持續性管理程式資源

建立持續性管理程式資源

向下相容性需要此功能。若要在版本 7 的 Application Server 上執行，使用容器管理的持續性 Bean (一種 EJB 元件) 的應用程式需要持續性管理程式資源。

若要建立持續性管理程式資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點。
2. 在 [資源] 節點下，選取 [持續性管理程式] 節點。
3. 在 [持續性管理程式] 頁面中，按一下 [新建]。
4. 在 [建立持續性管理程式] 頁面中，指定以下設定：
   a. 在 [JNDI 名稱] 欄位中，鍵入唯一的名稱，例如：`jdo/mypm`。請不要忘記正斜線。
   b. 在 [工廠類別] 欄位中，保留此版本提供的預設類別，或鍵入其他實作類別。
   c. 從 [連線池] 組合方塊中選擇新的持續性管理程式資源所屬的連線池。
   d. 依預設，將啓用新的持續性管理程式資源。若要將其停用，請取消選取 [啓用] 核取方塊。
   e. 在 [目標] 區段，指定具有可用資源的目標 (叢集和獨立伺服器實例)。在左側選取所需的目標，然後按一下 [增加] 以將其增加到選取目標的清單中。
5. 按一下 [確定]。

等效的 `asadmin` 指令爲：`create-persistence-resource`

編輯持續性管理程式資源

若要編輯現有的持續性管理程式資源特性，請執行以下步驟：

1. 在 [編輯持續性管理程式特性] 標籤中，選取 [增加特性] 按鈕。

   [附加特性] 表中將增加一個新列。

2. 增加所需的特性和值。

管理資源目標

若要管理資源目標，請執行以下步驟：

1. 選取 [目標] 標籤，以變更資源所在的目標 (叢集和獨立伺服器實例)。
2. 選取清單中的現有目標的核取方塊，然後按一下 [啓用] 以啓用該目標的資源，或者按一下 [停用] 以停用該目標的資源。
3. 按一下 [管理目標] 可以在清單中增加或移除目標。在 [管理目標] 頁面中，從左側的 [可用] 清單中選取所需的目標，然後按一下 [增加] 以將其增加到選取目標的清單中。按一下 [移除] 可以從 [已選取] 清單中移除目標。
4. 按一下 [確定] 以儲存對可用目標的變更。
5. 按一下 [儲存]。

刪除持續性管理程式資源

1. 在樹形元件中，展開 [持續性管理程式] 節點。
2. 選取 [持續性管理程式] 節點。
3. 在 [持續性管理程式] 頁面中，選取要刪除的持續性管理程式的核取方塊。
4. 按一下 [刪除]。

等效的 `asadmin` 指令爲：`delete-persistence-resource`

啟用和停用持續性管理程式資源

1. 在樹形元件中，展開 [持續性管理程式] 節點。
2. 選取要啓用或停用的資源的核取方塊。
3. 按一下 [啓用] 或 [停用]。

第 7 章

配置可用性和階段作業持續性

本章描述如何在 Sun Java™ System Application Server Enterprise Edition 環境中配置階段作業持續性和可用性。它包含以下小節：

- 關於可用性和階段作業持續性
- 用於配置可用性的管理主控台作業

關於可用性和階段作業持續性

- 需要階段作業持續性的原因
- 階段作業持續性配置概況
- 可用性層級
- 處於 HTTP 階段作業狀態的單次登入之可用性
- 應用程式範例

需要階段作業持續性的原因

在應用程式階段作業進行時，經常會有屬於階段作業的資料未儲存在傳統的資料庫中。例如，購物車的內容就是這樣的資料。Sun Java System Application Server 提供在儲存庫中儲存或保留此階段作業資料的功能，這樣，如果應用程式伺服器實例出現故障，階段作業狀態可以恢復，並且階段作業可以繼續進行而不會遺失資訊。

在 J2EE 應用程式中，階段作業資料通常儲存在 HTTP 階段作業或有狀態階段作業 Bean (SFSB) 階段作業中。Sun Java System Application Server 同時支援 HTTP 階段作業和 SFSB 階段作業的狀態持續性。還支援同時儲存在 HTTP 階段作業和 SFSB 階段作業中的某些 J2EE 物件參照之防故障備用，請參閱「Developer's Guide」。

隨附於 Sun Java System Application Server 的高可用性資料庫 (HADB) 可用做持續性儲存庫，以便爲階段作業資料提供高可用性。

階段作業持續性配置概況

爲了成功完成階段作業持續性配置，請確定按其所列順序執行以下步驟，因爲前面一個或多個步驟將做爲執行後面一些步驟的先決條件。

1. 爲叢集建立 HADB 資料庫。請參閱 「Reference Manual」中 `configure-ha-cluster` 指令的描述。
2. 爲叢集設定 HTTP 負載平衡。請參閱第 3 章，「配置負載平衡和防故障備用」。
3. 爲應用程式伺服器實例和 Web 或 EJB 容器 (應支援階段作業持續性) 啓用可用性，並配置階段作業持續性設定。選擇以下方法之一：
   - 請參閱第 136 頁的 「用於配置可用性的管理主控台作業」。
   - 請參閱 「Reference Manual」中 `configure-ha-persistence` 指令的描述。
4. 如果您*未*啓用可用性，則可以爲 SFSB 變更檔案系統階段作業儲存 (如果需要)。請參閱第 136 頁的 「停用可用性時配置 SFSB 階段作業儲存」。
5. 重新啓動叢集中的每個伺服器實例。
6. 爲需要可用性的任何特定 SFSB 啓用可用性，然後選取需要爲其進行階段作業狀態檢查點操作的方法。請參閱 「Developer's Guide」。
7. 使每個應具有高可用性的 Web 模組可分散。請參閱 「Developer's Guide」。
8. 在部署期間，爲 J2EE 應用程式、Web 模組或 EJB 模組啓用可用性。在管理主控台中，核取 [啓用可用性] 方塊，或結合使用 `deploy` 指令和設定爲 `true` 的 `--availabilityenabled` 選項。

備註 階段作業持續性與動態部署、動態重新載入和自動部署不相容。這些部署功能適用於開發環境而非生產環境。如需有關如何停用這些功能的資訊，請參閱第 5 章，「部署應用程式」。

備註 如果實例目前正在處理請求，請在重新啓動該實例前將其靜止，這樣，它就有足夠的時間來處理正在處理的請求。如需更多資訊，請參閱 「停用 (靜止) 伺服器實例或叢集」。

可用性層級

可用性可在五個不同層級上啓用：

1. 伺服器實例，依預設已啓用
2. Web 或 EJB 容器，依預設已啓用
3. 應用程式，依預設已停用
4. 獨立的 Web 或 EJB 模組，依預設已停用
5. SFSB，依預設已停用

對於要啓用的給定層級之可用性，還必須在所有更高層級中啓用。例如，若要啓用應用程式層級的可用性，必須在伺服器實例層級和容器層級啓用可用性。

給定層級的預設值是其上一層級的設定值。例如，如果已啓用容器層級的可用性，則依預設啓用應用程式層級的可用性。

如果已停用伺服器實例層級的可用性，則啓用其他任何層級的可用性都不生效。如果已啓用伺服器實例層級的可用性，除非已明確停用，否則將啓用所有層級的可用性。

處於 HTTP 階段作業狀態的單次登入之可用性

在單一應用程式伺服器實例中，如果使用者已由一個應用程式認證，則在同一實例上執行的其他應用程式將不會對該使用者進行重新認證。這稱爲**單次登入**。如需有關單次登入的更多資訊，請參閱第 204 頁的「驗證單次登入」。

若要使該功能在 HTTP 階段作業防故障備用到叢集中另一個實例後仍然可用，必須將單次登入資訊保留在 HADB 中。首先啓用伺服器實例和 Web 容器的可用性，然後啓用單次登入狀態持續性。請參閱第 137 頁的「配置伺服器實例層級的可用性」。

可以透過單一名稱和密碼的組合進行存取的應用程式組成了**單次登入群組**。

對於與應用程式 (是單次登入群組的一部分) 相對應的 HTTP 階段作業，如果其中一個階段作業逾時，其他階段作業並不會失效，並且仍然可用。這是因爲一個階段作業的逾時不應影響其他階段作業的可用性。

因此，如果一個階段作業逾時並且您嘗試從執行該階段作業的同一瀏覽器視窗存取相應的應用程式，則無需再次進行認證。但是將建立一個新的階段作業。

以屬於含有其他兩個應用程式的單次登入群組的購物車應用程式爲例。假設其他兩個應用程式的階段作業逾時值大於購物車應用程式的階段作業逾時值。如果購物車應用程式的階段作業逾時，並且您嘗試從執行該階段作業的同一瀏覽器視窗執行購物車應用程式，則無需再次進行認證。但是，前一輛購物車將遺失，並且必須建立一輛新的購物車。即使執行購物車應用程式的階段作業已逾時，其他兩個應用程式也會繼續照常執行。

類似地，可假定與其他兩個應用程式中的任何一個應用程式對應的階段作業逾時。當從執行該階段作業的同一瀏覽器視窗連線應用程式時，您無需再次進行認證。

備註 此運作方式僅適用於階段作業逾時情況。如果啓用了單次登入並且您使用 `HttpSession.invalidate()` 令其中一個階段作業失效，則屬於單次登入群組的所有應用程式之階段作業都將失效。如果您嘗試存取屬於單次登入群組的任一應用程式，則需要再次進行認證，系統將爲存取該應用程式的用戶端建立一個新的階段作業。

應用程式範例

以下目錄包含用於說明 HTTP 和 SFSB 階段作業持續性的範例應用程式：

install_dir`/samples/ee-samples/highavailability`
install_dir`/samples/ee-samples/failover`

用於配置可用性的管理主控台作業

- 停用可用性時配置 SFSB 階段作業儲存
- 配置伺服器實例層級的可用性
- 配置 Web 容器層級的可用性
- 配置 EJB 容器層級的可用性

停用可用性時配置 SFSB 階段作業儲存

如果停用了可用性，本機檔案系統將用於 SFSB 狀態鈍化，但不是持續性。若要變更 SFSB 狀態的儲存位置，請變更 EJB 容器中的 [階段作業儲存位置] 設定。請參閱第 191 頁的「配置一般 EJB 設定」。

配置伺服器實例層級的可用性

若要使用管理主控台啓用或停用伺服器實例層級的可用性，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 展開要編輯的配置之節點。
3. 選取 [可用性服務] 節點。
4. 移至 [可用性服務] 頁面。
5. 透過核取 [可用性服務] 方塊來啓用實例層級可用性。若要停用實例層級可用性，請取消核取該方塊。

   如果變更了用於連線 HADB 的 JDBC 資源以獲得階段作業持續性，您可以變更儲存池名稱。如需有關詳細資訊，請參閱 「Reference Manual」中 `configure-ha-cluster` 指令的描述。
6. 按一下 [儲存] 按鈕。
7. 展開 [實例] 節點。
8. 選取伺服器實例。
9. 移至伺服器實例頁面。
10. 重新啓動伺服器。

配置 Web 容器層級的可用性

若要啓用可用性或置換單一 Web 應用程式的可用性設定，請使用 `sun-web.xml` 檔案中的設定。如需有關詳細資訊，請參閱 「Developer's Guide」。

若要使用管理主控台啓用或停用 Web 容器的可用性，請執行以下步驟：

1. 選取 [Web 容器可用性] 標籤，然後核取 [可用性服務] 方塊。若要停用可用性，請取消核取該方塊。您還可以變更以下可選設定：

   - 持續性類型：爲已啓用可用性的 Web 應用程式指定階段作業持續性機制。允許的值包括 `memory` (無持續性)、`file` (檔案系統) 和 `ha` (HADB)。如果啓用了可用性，則預設爲 `ha`。如果停用了可用性，則預設爲 `memory`。對於需要階段作業持續性的生產環境，請使用 `ha`。

     如果將持續性類型設定爲 `memory`，您可以使用 `sessionFilename` 特性來指定正常關閉伺服器實例時儲存 HTTP 階段作業狀態的檔案系統位置。這對於內部測試很有用，但不受生產環境支援。

如果將持續性類型設定爲 file，您可以使用目錄特性來指定儲存 HTTP 階段作業狀態的檔案系統位置。保留檔案系統對於內部測試很有用，但不受生產環境支援。

- 持續性頻率：指定儲存階段作業狀態的頻率。僅當持續性類型爲 ha 時適用。允許的值包括：
  - web-method — 將回應傳送回用戶端之前，在每個 Web 請求結束時儲存階段作業狀態。此模式爲發生故障時完全更新階段作業狀態提供了最好的保證。此爲預設值。
  - time-based — 在後台中根據 reapIntervalSeconds 儲存特性設定的頻率儲存階段作業狀態。此模式不能保證完全更新階段作業狀態。但是，它可以提供很大的效能改善，因爲在每個請求之後都不儲存狀態。若要設定此特性，請參閱第 190 頁的「配置儲存特性」。
- 持續性範圍：指定儲存階段作業狀態的量。僅當持續性類型爲 ha 時適用。允許的值包括：
  - session — 每次儲存整個階段作業狀態。此模式爲正確儲存任何可分散 Web 應用程式的階段作業資料提供了最好的保證。此爲預設值。
  - modified-session — 如果階段作業已進行修改，則儲存整個階段作業狀態。如果呼叫了 HttpSession.setAttribute （）或 HttpSession.removeAttribute （），則系統將認爲階段作業狀態已進行修改。您必須保證每次變更屬性時都呼叫 setAttribute()。這不是 J2EE 規格的需求，但是此模式需要這樣做才能正常工作。
  - modified-attribute — 僅儲存修改的階段作業屬性。若要使此模式正常工作，您必須遵循一些指導原則。

    每次修改階段作業狀態時都呼叫 setAttribute()。

    確定各屬性之間沒有交叉參照。系統將對每個不同屬性關鍵字下的物件圖形單獨進行串列化並單獨儲存。如果每個單獨的關鍵字下的物件之間存在物件交叉參照，則它們將不會進行正確串列化和反串列化。

    在多個屬性之間分布階段作業狀態，或者至少在唯讀屬性和可修改屬性之間分布階段作業狀態。
- 單次登入狀態：核取此方塊以啓用單次登入狀態的持續性。若要停用持續性，請取消核取該方塊。
- HTTP 階段作業儲存：如果變更了用於連線 HADB 的 JDBC 資源以獲得階段作業持續性，您可以變更 HTTP 階段作業儲存。如需有關詳細資訊，請參閱「Reference Manual」中 configure-ha-cluster 指令的描述。

2. 按一下 [儲存] 按鈕。

3. 若要變更影響階段作業持續性的其他可選設定，請參閱第 188 頁的「配置 Web 容器階段作業」。
4. 展開 [實例] 節點。
5. 選取伺服器實例。
6. 移至伺服器實例頁面。
7. 重新啓動伺服器。

配置 EJB 容器層級的可用性

若要啓用可用性並爲單一有狀態階段作業 Bean (SFSB) 選取要執行檢查點作業的方法，請使用 `sun-ejb-jar.xml` 檔案中的設定。如需有關詳細資訊，請參閱「Developer's Guide」。

若要使用管理主控台來啓用或停用 EJB 容器的可用性，請執行以下步驟：

1. 選取 [EJB 容器可用性] 標籤，然後核取 [可用性服務] 方塊。若要停用可用性，請取消核取該方塊。您還可以變更以下可選設定：
   - HA 持續性類型：爲已啓用可用性的 SFSB 指定階段作業持續性和鈍化機制。允許的值包括 `file` (檔案系統) 和 `ha` (HADB)。對於需要階段作業持續性的生產環境，請使用預設值 `ha`。
   - SFSB 持續性類型：爲**尚未**啓用可用性的 SFSB 指定鈍化機制。允許的值包括 `file` (預設值) 和 `ha`。

     如果將任一持續性類型設定爲 `file`，EJB 容器都將指定用於儲存已鈍化的階段作業 Bean 狀態的檔案系統位置。請參閱 第 191 頁的「配置一般 EJB 設定」。檔案系統的檢查點操作對於內部測試很有用，但不受生產環境支援。
   - SFSB 儲存池名稱：如果您已變更用於連線 HADB 的 JDBC 資源，則可以變更 SFSB 儲存池名稱以獲得階段作業持續性。如需有關詳細資訊，請參閱「Reference Manual」中 `configure-ha-cluster` 指令的描述。
2. 按一下 [儲存] 按鈕。
3. 展開 [實例] 節點。
4. 選取伺服器實例。
5. 移至伺服器實例頁面。
6. 重新啓動伺服器。

第 8 章

配置 Java 訊息服務資源

本章描述如何爲使用 Java 訊息服務 (JMS) API 的應用程式配置資源。它包含以下小節：

- 關於 JMS 資源
- 用於 JMS 連線工廠的管理主控台作業
- 用於 JMS 目標資源的管理主控台作業
- 用於 JMS 實體目標的管理主控台作業
- 用於 JMS 提供者的管理主控台作業

關於 JMS 資源

- Application Server 中的 JMS 提供者
- JMS 資源
- JMS 資源與連接器資源之間的關係

Application Server 中的 JMS 提供者

Application Server 透過將 Sun Java System Message Queue (原來稱爲 Sun ONE Message Queue) 整合到 Application Server 中，實作了 Java 訊息服務 (JMS) API。對於基本的 JMS API 管理作業，請使用 Application Server 管理主控台。對於進階作業 (包括管理 Message Queue 叢集)，請使用 *install_dir*`/imq/bin` 目錄中提供的工具。

如需有關管理 Message Queue 的詳細資訊，請參閱 「Sun Java System Message Queue Administration Guide」。

JMS 資源

Java 訊息服務 (JMS) API 使用兩種管理物件：

- 允許應用程式以程式化方式建立其他 JMS 物件的連線工廠物件。
- 用作訊息儲存庫的目標。

這些物件是以管理方式建立的，而建立物件的方式則特定於每個 JMS 實作。在 Application Server 中，請執行以下作業：

- 透過建立連線工廠資源來建立連線工廠
- 透過建立兩個物件來建立目標：
  - 實體目標
  - 參考實體目標的目標資源

JMS 應用程式使用 JNDI API 來存取連線工廠和目標資源。通常，JMS 應用程式至少使用一個連線工廠和一個目標。若要瞭解應建立的資源，請研究應用程式或向應用程式開發者洽詢。

連線工廠分爲三種類型：

- `QueueConnectionFactory` 物件，用於點對點通訊
- `TopicConnectionFactory` 物件，用於出版訂閱通訊
- `ConnectionFactory` 物件，可用於點對點通訊和出版訂閱通訊；建議將這些物件用於新的應用程式

有兩種類型的目標：

- `Queue` 物件，用於點對點通訊
- `Topic` 物件，用於出版訂閱通訊

「J2EE 1.4 Tutorial」中有關 JMS 的章節提供了有關這兩類通訊和 JMS 其他方面的詳細資訊 (請參閱 `http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html`)。

建立資源的次序並不重要。

對於 J2EE 應用程式，請在 Application Server 部署描述元中指定連線工廠和目標資源，如下所示：

- 在 `resource-ref` 或 `mdb-connection-factory` 元素中指定連線工廠 JNDI 名稱。

- 在訊息導引 Bean 的 `ejb` 元素和 `message-destination` 元素中指定目標資源 JNDI 名稱。
- 在 `message-destination-link` 元素中指定實體目標名稱，該元素在企業 Bean 部署描述元的 `message-driven` 元素或 `message-destination-ref` 元素內。此外，還應在 `message-destination` 元素中指定該實體目標名稱。(`message-destination-ref` 元素替代了在新的應用程式中已被拒絕的 `resource-env-ref` 元素。) 在 Application Server 部署描述元的 `message-destination` 元素中，將實體目標名稱與目標資源名稱連結起來。

JMS 資源與連接器資源之間的關係

Application Server 透過使用名爲 `jmsra` 的系統資源介面實作 JMS。使用者建立 JMS 資源時，Application Server 會自動建立連接器資源，這些連接器資源將顯示在管理主控台樹檢視的 [連接器] 節點下。

對於使用者建立的每個 JMS 連線工廠，Application Server 都將建立連接器連線池和連接器資源。對於使用者建立的每個 JMS 目標，Application Server 都會建立管理物件資源。使用者刪除 JMS 資源時，Application Server 將自動刪除連接器資源。

可以透過使用管理主控台的 [連接器] 節點 (而不是使用 [JMS 資源] 節點) 來爲 JMS 系統資源介面建立連接器資源。請參閱第 11 章，「連接器資源」，以取得詳細資訊。

用於 JMS 連線工廠的管理主控台作業

- 建立 JMS 連線工廠資源
- 編輯 JMS 連線工廠資源
- 刪除 JMS 連線工廠資源

建立 JMS 連線工廠資源

若要建立 JMS 連線工廠資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [JMS 資源] 節點。
2. 選取 [連線工廠] 節點。

3. 在 [JMS 連線工廠] 頁面中，按一下 [新建]。將顯示 [建立 JMS 連線工廠] 頁面。
4. 在 [JNDI 名稱] 欄位中，鍵入連線工廠的名稱。例如：

   `jms/ConnectionFactory1`

   建議的做法是使用 JMS 資源的命名子環境前綴 `jms/`。
5. 從 [類型] 下拉式清單中，選擇 `javax.jms.ConnectionFactory`、`javax.jms.QueueConnectionFactory` 或 `javax.jms.TopicConnectionFactory`。
6. 選取 [啓用] 核取方塊以在執行階段啓用該資源。
7. 在 [進階] 區域中，變更連線工廠屬性所需的值。如需有關這些屬性的詳細資訊，請參閱第 173 頁的「編輯連接器連線池」中標題爲「連接器連線池的池設定」的表。Application Server 會將這些屬性套用到爲連線工廠建立的連接器連線池。

   對於 JMS 連線工廠資源，請指定作業事件支援值，如下所示：

   - 爲可用於作業事件 (此作業事件包含在作業事件範圍內使用的多個資源，例如，此資源加上 JDBC 資源、連接器資源或其他 JMS 連線工廠資源) 的資源指定 `XATransaction` (預設值)。此值提供了最大的靈活性。配置爲 `XATransaction` 的資源將參與分爲兩個階段的確定作業。
   - 爲可用於作業事件的資源 (此作業事件僅包含作業事件範圍內的唯一一種資源)，或用作分散式作業事件中的最後一個代理程式 (此分散式作業事件包含多個 XA 資源) 的資源指定 `LocalTransaction`。使用此值可以獲得更好的效能。配置爲 `LocalTransaction` 的資源將不會用於分爲兩個階段的確定作業。
   - 爲永遠無法參與作業事件的資源指定 `NoTransaction`；此設定在 JMS 應用程式中的使用受限。

8. 在 [附加特性] 區域中，爲應用程式所需的特性提供值。下表列示了可用的特性。

表 8-1 JMS 連線工廠的附加特性

特性名稱	描述
ClientId	為長期訂戶將要使用的連線工廠指定用戶端 ID。
AddressList	指定應用程式將進行通訊的訊息代理程式實例的名稱 (和連接埠號 [選擇性地])。清單中的每個位址都指定了要連線的主機名稱 (和連接埠號與連線服務 [選擇性地])。例如，該值可能為 earth 或 earth:7677。當訊息代理程式在預設連接埠 (7676) 以外的連接埠上執行時，請指定連接埠號。如果特性設定在叢集環境中指定了多個主機和連接埠，則將使用該清單中的第一個可用主機 (除非將 AddressListBehavior 特性設定為 RANDOM)。 如需有關詳細資訊，請參閱 「Sun Java System Message Queue Developer's Guide for Java Clien's」。 預設：本地主機和預設連接埠號 (7676)。用戶端將嘗試連線至位於本地主機連接埠 7676 上的代理程式。
MessageServiceAddressList	與 AddressList 相同。此特性名稱已被拒絕。請使用 AddressList 代替。
使用者名稱	進入連線工廠的使用者名稱。 預設：guest
密碼	進入連線工廠的密碼。 預設：guest
ReconnectEnabled	如果啟用該屬性 (值 = true)，則指定用戶端執行階段在遺失連線時嘗試重新連線至訊息伺服器 (或 AddressList 中的位址清單)。 預設：TRUE
ReconnectAttempts	指定用戶端執行階段嘗試連線 (或重新連線) AddressList 清單中每個位址的次數。到達這個值後，用戶端執行階段將嘗試連線清單中的下一個位址。值 -1 表示重新連線嘗試次數沒有限制 (用戶端執行階段將嘗試連線至第一個位址，直到連線成功)。 預設：3
ReconnectInterval	指定兩次連線嘗試之間的間隔 (以毫秒為單位)。此屬性適用於對 AddressList 中每個位址的嘗試，及對該清單中連續位址的嘗試。如果該間隔太短，則代理程式將沒有時間恢復。如果該間隔太長，則重新連線會變得遲緩，以至於讓人無法接受。 預設：30000

表 8-1 JMS 連線工廠的附加特性 (續)

特性名稱	描述
AddressListBehavior	指定按 AddressList 屬性中的位址的次序 (PRIORITY) 還是按隨機次序 (RANDOM) 嘗試連線。 RANDOM 表示重新連線將從 AddressList 中隨機選擇位址。如果許多用戶端嘗試使用同一個連線工廠來進行連線，則使用該值可以阻止它們全部連線至同一個位址。 PRIORITY 表示重新連線時始終嘗試連線 AddressList 中的第一個伺服器位址，而僅在第一個代理程式不可用時才使用其他位址。 預設：RANDOM
AddressListIterations	指定用戶端執行階段建立 (或重新建立) 連線時，在 AddressList 中循環的次數。值 -1 表示嘗試次數沒有限制。 預設：3

9. 在 [目標] 區域中，執行以下操作：
   a. 從 [可用] 欄中選取一個或多個目標，在這些目標上將部署使用了資源的應用程式。可用目標包括可用叢集和可用伺服器實例，以及預設的伺服器實例 server。
   b. 按一下 [新增] 以將目標移至 [選取] 欄中。
10. 按一下 [確定] 以儲存連線工廠。

等效的 asadmin 指令爲：create-jms-resource

編輯 JMS 連線工廠資源

若要編輯 JMS 連線工廠資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [JMS 資源] 節點。
2. 展開 [連線工廠] 節點。
3. 選取要編輯的連線工廠。
4. 在 [編輯 JMS 連線工廠] 頁面中，可以執行以下作業：
   - 修改 [描述] 欄位中的文字。
   - 選取或取消選取 [啓用] 核取方塊以啓用或停用資源。
   - 變更 [進階] 區域中的屬性值。
   - 新增、移除或修改特性。

5. 此外，按一下 [目標] 標籤以移至 [JMS 連線工廠資源目標] 頁面。在此頁面中，執行以下操作：

   a. 按一下 [管理目標] 以開啓 [管理資源目標] 頁面。

      在此頁面中，在 [可用] 欄與 [選取] 欄之間移動目標。確定將目標放入 [選取] 欄中。使用了資源的應用程序將部署在這樣的目標上。可用目標包括可用叢集和可用伺服器實例，以及預設的伺服器實例 `server`。按一下 [確定] 以儲存變更。

   b. 選取目標的核取方塊，然後按一下 [啓用] 或 [停用]，以啓用或停用目標的資源。

6. 按一下 [儲存] 以儲存變更。

刪除 JMS 連線工廠資源

若要刪除 JMS 連線工廠資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [JMS 資源] 節點。
2. 選取 [連線工廠] 節點。
3. 在 [JMS 連線工廠] 頁面中，選取要刪除的連線工廠的名稱旁邊的核取方塊。
4. 按一下 [刪除]。

等效的 `asadmin` 指令爲：`delete-jms-resource`

用於 JMS 目標資源的管理主控台作業

- 建立 JMS 目標資源
- 編輯 JMS 目標資源
- 刪除 JMS 目標資源

建立 JMS 目標資源

若要建立 JMS 目標資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [JMS 資源] 節點。
2. 選取 [目標資源] 節點。

3. 在 [JMS 目標資源] 頁面中，按一下 [新建]。將顯示 [建立 JMS 目標資源] 頁面。
4. 在 [JNDI 名稱] 欄位中，鍵入資源的名稱。例如：

   `jms/Queue`

   建議的做法是使用 JMS 資源的命名子環境前綴 `jms/`。
5. 從 [類型] 下拉式清單中，選擇 `javax.jms.Topic` 或 `javax.jms.Queue`。
6. 選取 [啓用] 核取方塊以在執行階段啓用該資源。
7. 在 [附加特性] 區域中，爲特性提供值。下表列示了可用的特性。

表 8-2 JMS 目標資源的附加特性

特性名稱	描述
`Name`	(需要) 資源參考的實體目標的名稱。
`Description`	實體目標的描述。

8. 在 [目標] 區域中，執行以下操作：
   a. 從 [可用] 欄中選取一個或多個目標，在這些目標上將部署使用了資源的應用程式。可用目標包括可用叢集和可用伺服器實例，以及預設的伺服器實例 `server`。
   b. 按一下 [新增] 以將目標移至 [選取] 欄中。
9. 按一下 [確定]。

等效的 `asadmin` 指令爲：`create-jms-resource`

編輯 JMS 目標資源

若要編輯 JMS 目標資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [JMS 資源] 節點。
2. 展開 [目標資源] 節點。
3. 選取要編輯的目標資源。
4. 在 [編輯 JMS 目標資源] 頁面中，可以執行以下作業：
   - 變更資源的類型。

- 修改 [描述] 欄位中的文字。
- 選取或取消選取 [啓用] 核取方塊以啓用或停用資源。
- 新增、移除或修改 Name 或 Description 特性。

5. 按一下 [儲存] 以儲存變更。
6. 此外，按一下 [目標] 標籤以移至 [JMS 目標資源目標] 頁面。在此頁面中，執行以下操作：

   a. 按一下 [管理目標] 以開啓 [管理資源目標] 頁面。

      在此頁面中，在 [可用] 欄與 [選取] 欄之間移動目標。確定將目標放入 [選取] 欄中。使用了資源的應用程序將部署在這樣的目標上。可用目標包括可用叢集和可用伺服器實例，以及預設的伺服器實例 `server`。按一下 [確定] 以儲存變更。

   b. 選取目標的核取方塊，然後按一下 [啓用] 或 [停用]，以啓用或停用目標的資源。

刪除 JMS 目標資源

若要刪除 JMS 目標資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [JMS 資源] 節點。
2. 選取 [目標資源] 節點。
3. 在 [JMS 目標資源] 頁面中，選取要刪除的目標資源的名稱旁邊的核取方塊。
4. 按一下 [刪除]。

等效的 `asadmin` 指令爲：`delete-jms-resource`

用於 JMS 實體目標的管理主控台作業

- 建立 JMS 實體目標
- 刪除 JMS 實體目標

建立 JMS 實體目標

若要進行生產，務必建立實體目標。但是，在開發和測試期間，不需要執行此步驟。應用程式首次存取目標資源時，Message Queue 會自動建立目標資源的 Name 特性指定的實體目標。該實體目標是暫時的，並且將在 Message Queue 配置特性指定的時間段後過期。

若要建立 JMS 實體目標，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點，然後展開 [Java 訊息服務] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 選取 [實體目標] 節點。
4. 在 [實體目標] 頁面中，按一下 [新建]。將顯示 [建立實體目標] 頁面。
5. 在 [實體目標名稱] 欄位中，鍵入目標的名稱 (例如，`PhysicalQueue`)。
6. 從 [類型] 下拉式清單中，選擇 `topic` 或 `queue`。

7. 在 [附加特性] 區域中，按一下 [增加特性] 以增加特性。下表列示了目前可用的一個特性。

表 8-3 JMS 實體目標的附加特性

特性名稱	描述
maxNumActiveConsumers	在佇列目標的負載平衡傳送中處於使用中狀態的使用者的最大數目。如果值為 -1，則表示沒有數目限制。如果目標是為獨立伺服器實例建立的，則預設值為 1；如果目標是為叢集建立的，則預設值為 -1。

若要修改此特性的值或要指定其他實體目標特性，請使用 *install_dir*/imq/bin/imqcmd 指令。請參閱 「Sun Java System Message Queue Administration Guide」，以取得更多資訊。

8. 按一下 [確定]。

[實體目標] 頁面顯示系統目標，即過期和無法傳送的訊息重新導向至的佇列 mq.sys.dmq。可以爲此目標建立目標資源、使用者和瀏覽器，但不能刪除它或向其傳送訊息。

等效的 asadmin 指令爲：create-jmsdest

刪除 JMS 實體目標

若要刪除 JMS 實體目標，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點，然後展開 [Java 訊息服務] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 server，請選取 server-config 節點。
   b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 default-config 節點。
3. 選取 [實體目標] 節點。
4. 在 [實體目標] 頁面中，選取要刪除的目標的名稱旁邊的核取方塊。
5. 按一下 [刪除]。

如果嘗試刪除系統目標 mq.sys.dmq，將顯示錯誤訊息。

等效的 asadmin 指令爲：delete-jmsdest

用於 JMS 提供者的管理主控台作業

- 配置 JMS 提供者的一般特性
- 建立 JMS 主機
- 編輯 JMS 主機
- 刪除 JMS 主機

配置 JMS 提供者的一般特性

使用 [JMS 服務] 頁面配置所有 JMS 連線都使用的特性。請依照下列步驟執行：

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 選取 [Java 訊息服務] 節點以開啓 [JMS 服務] 頁面。
4. 請編輯 [啓動逾時] 欄位中的值，以變更 Application Server 在中斷啓動之前等待 JMS 服務啓動的時間。在運行緩慢或超載的系統中，在預設逾時值 (60) 的基礎上增大該值。
5. 從 [類型] 下拉式清單中：
   - 選擇 `LOCAL` (`server-config` 配置的預設設置) 以存取本地主機上的 JMS 服務。該 JMS 服務由 Application Server 啓動和管理。
   - 選擇 `REMOTE` (`default-config` 配置的預設設置) 以存取其他系統或叢集上的 JMS 服務。如果選擇了 `REMOTE`，則下一次啓動伺服器時 JMS 服務不會由 Application Server 啓動，而是通過 Message Queue 來啓動和管理 JMS 服務，因而必須單獨啓動 Message Queue 代理程式。如需有關啓動代理程式的資訊，請參閱 「Sun Java System Message Queue Administration Guide」。如果選擇此值並使用了遠端主機，請按照第 156 頁的「編輯 JMS 主機」中的說明來指定遠端主機的名稱。
6. 在 [啓動引數] 欄位中，鍵入引數以自訂 JMS 服務啓動。在 *install_dir*`/imq/bin/imqbrokerd` 指令中使用任何可用的引數。

7. 連線遺失時，使用 [重新連線] 核取方塊指定 JMS 服務是否嘗試重新連線至訊息伺服器 (或 AddressList 中的位址清單)。

   依預設，啓用重新連線。

8. 在 [重新連線間隔] 欄位中，鍵入重新連線嘗試之間的秒數。此屬性適用於對 AddressList 中每個位址的嘗試，及對該清單中連續位址的嘗試。如果該間隔太短，則代理程式將沒有時間恢復。如果該間隔太長，則重新連線會變得遲緩，以至於讓人無法接受。

   其預設值爲 60 秒。

9. 在 [重新連線嘗試] 欄位中，鍵入用戶端執行階段嘗試連線 (或重新連線) AddressList 清單中每個位址的次數。到達這個值後，用戶端執行階段將嘗試連線清單中的下一個位址。值 -1 表示重新連線嘗試次數沒有限制 (用戶端執行階段將嘗試連線至第一個位址，直到連線成功)。

   預設值爲 3。

10. 從 [預設 JMS 主機] 下拉式清單中選擇一個主機。預設爲 `default_JMS_host`。

11. 在 [位址清單運作方式] 下拉式清單中，選擇是按 AddressList 中的位址次序 (`priority`) 還是按隨機次序 (`random`) 嘗試連線。

    `priority` 表示重新連線始終嘗試連線 AddressList 中的第一個伺服器位址，而僅在第一個代理程式不可用時才使用其他位址。

    如果許多用戶端嘗試使用同一個連線工廠來進行連線，則應指定 `random` 以防止它們全部連線至同一個位址。

    預設爲 `random`。

12. 在 [位址清單循環] 欄位中，鍵入 JMS 服務建立 (或重新建立) 連線時，在 AddressList 中循環的次數。值 -1 表示嘗試次數沒有限制。

    預設值爲 3。

13. 在 [MQ 方案] 和 [MQ 服務] 欄位中，鍵入 Message Queue 位址方案名稱和 MQ 連線服務名稱 (如果要使用非預設方案或服務)。訊息服務位址的完整語法爲

scheme://*address_syntax*

其中，*scheme* 和 *address_syntax* 將在下表中描述。

MQ 方案和 MQ 服務是下表的前兩欄中顯示的值。

表 8-4 訊息伺服器位址方案和語法

方案名稱	連線服務	描述	位址語法
mq	jms 和 ssljms	MQ 用戶端執行階段將連線位於指定主機和連接埠的 MQ 連接埠對映器。該連接埠對映器傳回動態建立的連線服務連接埠的清單，然後 MQ 用戶端執行階段將連線託管指定連線服務的連接埠。	[*hostName*][:*port*][/*serviceName*] 預設： *hostName* = localhost *port* = 7676 *serviceName* = jms 預設僅適用於 jms 連線服務。對於 ssljms 連線服務，需要指定所有的變數 範例： mq:MyHost:7677/ssljms
mqtcp	jms	MQ 用戶端執行階段將透過與指定的主機和連接埠建立 TCP 連線 (繞過 MQ 連接埠對映器) 來建立連線。	*hostName*:*port*/jms 範例： mqtcp:localhost:7676/jms
mqssl	ssljms	MQ 用戶端執行階段將透過與指定的主機和連接埠建立安全 SSL 連線 (繞過 MQ 連接埠對映器) 來建立連線。	*hostName*:*port*/ssljms 範例： mqssl:localhost:7676/ssljms
http	httpjms	MQ 用戶端執行階段將與指定 URL 處的 MQ 隧道 Servlet 建立 HTTP 連線。(必須配置代理程式，以存取 MQ 的 Administrator's Guide 中所述的 HTTP 隧道 Servlet。)	*hostName*:*port*/*contextRoot*/tunnel 如果多個代理程式實例使用同一個隧道 Servlet，則連線特定代理程式實例 (而不是隨機選取的實例) 的語法為： http://*hostName*:*port*/*contextRoot*/tunnel?*serverName=hostName*:*instanceName*

表 8-4 訊息伺服器位址方案和語法 (續)

方案名稱	連線服務	描述	位址語法
`https`	`httpsjms`	MQ 用戶端執行階段將與指定的 MQ 隧道 Servlet URL 建立安全 HTTPS 連線。(必須配置代理程式，以存取 MQ 的 Administrator's Guide 中所述的 HTTPS 隧道 Servlet。)	*hostName*:*port*/*contextRoot*/`tunnel` 如果多個代理程式實例使用同一個隧道 Servlet，則連線特定代理程式實例 (而不是隨機選取的實例) 的語法為： `http://`*hostName*:*port*/*contextRoot*/`tunnel?`*serverName=hostName*:*instanceName*

14. 在 [附加特性] 區域中，按一下 [增加特性] 以增加特性。下表列示了可用的 Message Queue 代理程式配置特性。

表 8-5 JMS 提供者的附加特性

特性名稱	描述
`instance-name`	指定完整的 Sun Java System Message Queue 代理程式實例名稱。預設值為 `imqbroker`。
`instance-name-suffix`	指定要增加到完整的 Sun Java System Message Queue 代理程式實例名稱中的後綴。該後綴與實例名稱之間以底線字元 (_) 分隔。例如，如果實例名稱為 `imqbroker`，則在附加後綴 `xyz` 之後，實例名稱將變更為 `imqbroker_xyz`。
`append-version`	如果為 `true`，則應在完整的 Sun Java System Message Queue 代理程式實例名稱後附加主要和次要版本號碼，這兩個版本號碼之前都有一個底線字元 (_)。例如，如果實例名稱為 `imqbroker`，則在附加版本號碼之後，實例名稱將變更為 `imqbroker_8_0`。預設值為 `false`。

15. 按一下 [儲存] 以儲存變更，或按一下 [載入預設值] 以復原服務的預設值。

按一下 [Ping] 以查看 JMS 服務是否已啓動並正在執行。如果 JMS 服務已啓動並正在運行，則將顯示「Ping 成功：JMS 服務正在執行」的訊息。

將提供者和主機變更爲遠端系統會使所有 JMS 應用程式都在遠端伺服器上執行。若要在使用本機伺服器的同時使用一個或多個遠端伺服器，請使用 `AddressList` 特性建立連線工廠資源從而建立存取遠端伺服器的連線。

如需有關配置 JMS 服務的更多資訊，請參閱「Sun Java System Application Server Developer's Guide」。

等效的 `asadmin` 指令爲：`jms-ping`

建立 JMS 主機

若要建立 JMS 主機，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 展開 [Java 訊息服務] 節點。
4. 選取 [JMS 主機] 節點。
5. 在 [JMS 主機] 頁面中，按一下 [新建]。將顯示 [建立 JMS 主機] 頁面。
6. 在 [名稱] 欄位中，鍵入主機的名稱。例如：

   `NewJmsHost`
7. 在 [主機] 欄位中，鍵入執行 JMS 主機的系統的名稱或網際網路通訊協定 (IP) 位址 (`localhost` 或者本機或遠端系統的名稱)。
8. 在 [連接埠] 欄位中，鍵入 JMS 服務的連接埠號。僅當要使用的 JMS 服務在非預設連接埠上執行時，才需要變更此欄位。(預設連接埠號爲 7676。)
9. 在 [管理使用者名稱] 和 [管理密碼] 欄位中，鍵入 MQ 代理程式使用者名稱和密碼。該使用者名稱和密碼與 Application Server 的使用者名稱和密碼不同。僅當已經使用 *install_dir*`/imq/bin/imqusermgr` 指令變更了 MQ 代理程式值時，才需要編輯這些欄位。預設值爲 `admin` 和 `admin`。
10. 按一下 [確定]。

等效的 `asadmin` 指令爲：`create-jms-host`

編輯 JMS 主機

若要編輯 JMS 主機，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。

b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 default-config 節點。

3. 展開 [Java 訊息服務] 節點。
4. 選取 [JMS 主機] 節點。
5. 在 [JMS 主機] 頁面中，選取要編輯的主機。
6. 在 [編輯 JMS 主機] 頁面中，可以執行以下作業：
   - 在 [主機] 欄位中，變更主機名稱或網際網路通訊協定 (IP) 位址。
   - 在 [連接埠] 欄位中，變更 JMS 服務的連接埠號。
   - 變更 [管理使用者名稱] 和 [管理密碼] 欄位中的值。
7. 按一下 [儲存] 以儲存變更，或按一下 [載入預設值] 以復原主機的預設值。

刪除 JMS 主機

若要刪除 JMS 主機，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 server，請選取 server-config 節點。
   b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 default-config 節點。
3. 展開 [Java 訊息服務] 節點。
4. 選取 [JMS 主機] 節點。
5. 在 [JMS 主機] 頁面中，選取要刪除的主機名稱旁邊的核取方塊。
6. 按一下 [刪除]。

等效的 asadmin 指令爲：delete-jms-host

第 9 章

配置 JavaMail 資源

本章介紹如何爲使用 JavaMail API 的應用程式配置資源。它包含以下各節：

- 關於 JavaMail
- 有關 JavaMail 的管理主控台作業

關於 JavaMail

- JavaMail API

JavaMail API

JavaMail API 是一組用於建立郵件系統模型的抽象 API。API 提供了一個不受限於平台且不受限於通訊協定的架構來建置郵件和訊息傳送應用程式。JavaMail API 提供了多個工具用於讀取和傳送電子郵件。服務提供者可實作特定協定。

JavaMail API 可作爲 Java 平台選擇性套裝軟體實作，還可作爲 J2EE 平台的一部分使用。

Application Server 包含 JavaMail API 以及 JavaMail 服務提供者，使應用程式元件可以通過網際網路傳送電子郵件通知，以及從 IMAP 和 POP3 郵件伺服器讀取電子郵件。

如需有關 JavaMail API 的更多資訊，請移至 JavaMail 網站 (`http://java.sun.com/products/javamail/`)。

有關 JavaMail 的管理主控台作業

- 建立 JavaMail 階段作業
- 編輯 JavaMail 階段作業
- 刪除 JavaMail 階段作業

建立 JavaMail 階段作業

若要建立 JavaMail 階段作業，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後選取 [JavaMail 階段作業] 節點。
2. 在 [JavaMail 階段作業] 頁面中，按一下 [新建]。將顯示 [建立 JavaMail 階段作業] 頁面。
3. 在 [JNDI 名稱] 欄位中，鍵入階段作業的名稱。例如：

   `mail/MySession`

   建議的做法是使用 JavaMail 資源的命名子環境前綴 `mail/`。
4. 在 [郵件主機] 欄位中，鍵入預設郵件伺服器的主機名稱。如果未提供協定特定的主機特性，Store 和 Transport 物件的連線方法將使用該值。名稱必須可以解析爲實際的主機名稱。
5. 在 [預設使用者] 欄位中，鍵入連線至郵件伺服器時要提供的使用者名稱。如果未提供協定特定的使用者名稱特性，Store 和 Transport 物件的連線方法使用該值。
6. 在 [預設傳回位址] 欄位中，鍵入預設使用者的電子郵件位址，格式爲：*`username@host.domain`*。
7. 如果您不希望此時啓用郵件階段作業，請取消選取 [啓用] 核取方塊。
8. 在 [進階] 區段，僅當已將 Application Server 的郵件提供者重新配置爲使用非預設儲存或傳輸協定時，才需要變更這些欄位值。依預設，儲存協定爲 `imap`；儲存協定類別爲 `com.sun.mail.imap.IMAPStore`；傳輸協定爲 `smtp`；傳輸協定類別爲 `com.sun.mail.smtp.SMTPTransport`。

   選取 [除錯] 核取方塊以啓用郵件階段作業的附加除錯輸出 (包括通訊協定追蹤)。如果 JavaMail 記錄層級設定爲 [`FINE`] 或更精細，則系統會產生除錯輸出，並且該輸出包含在系統記錄檔中。請參閱第 305 頁的 「配置記錄層級」以取得有關設定記錄層級的資訊。

9. 在 [附加特性] 區域中，按一下 [增加特性] 以新增應用程式所需的特性 (如協定特定的主機或使用者名稱特性)。JavaMail API 文件列示了可用特性 (`http://java.sun.com/products/javamail/javadocs/index.html`)。
10. 在 [目標] 區域中，執行以下操作：
    a. 從 [可用] 欄中選取將要部署使用了資源的應用程式的一個或多個目標。可用目標包括可用叢集和可用伺服器實例，以及預設的伺服器實例 `server`。
    b. 按一下 [新增] 以將目標移至 [選取] 欄中。
11. 按一下 [確定] 以儲存階段作業。

等效的 `asadmin` 指令爲：`create-javamail-resource`

編輯 JavaMail 階段作業

若要編輯 JavaMail 階段作業，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後選取 [JavaMail 階段作業] 節點。
2. 在 [JavaMail 階段作業] 頁面中，選取要編輯的階段作業。
3. 在 [編輯 JavaMail 階段作業] 頁面中，您可以執行以下作業：
   - 修改 [郵件主機]、[預設使用者]、[預設傳回位址] 和 [描述] 欄位中的值。
   - 選取或取消選取 [啓用] 核取方塊以啓用或停用資源。
   - 修改 [進階] 欄位中的值。
   - 新增、移除或修改特性。
4. 按一下 [目標] 標籤以移至 [JavaMail 階段作業目標] 頁面。在此頁面中，執行以下操作：
   a. 按一下 [管理目標] 以開啓 [管理資源目標] 頁面。

      在此頁面中，在 [可用] 欄與 [選取] 欄之間移動目標。確定將要部署使用了資源的應用程式的目標放入 [選取] 欄中。可用目標包括可用叢集和可用伺服器實例，以及預設的伺服器實例 `server`。按一下 [確定] 以儲存變更。
   b. 選取目標的核取方塊，然後按一下 [啓用] 或 [停用]，以啓用或停用目標的資源。
5. 按一下 [儲存] 以儲存變更，或按一下 [載入預設值] 以復原郵件階段作業的預設值。

刪除 JavaMail 階段作業

若要刪除 JavaMail 階段作業，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後選取 [JavaMail 階段作業] 節點。
2. 在 [JavaMail 階段作業] 頁面中，選取要刪除的階段作業的名稱旁邊的核取方塊。
3. 按一下 [刪除]。

等效的 asadmin 指令爲：`delete-javamail-resource`

第 10 章

JNDI 資源

本章描述如何使用管理主控台來配置 JNDI 資源。它包含以下小節：

- 關於 Java 命名與目錄介面 (JNDI)
- 關於自訂資源
- 關於外部 JNDI 儲存庫和資源

關於 Java 命名與目錄介面 (JNDI)

本小節將討論 Java 命名和目錄介面 (JNDI)。JNDI 是用於存取不同種類的命名和目錄服務的應用程式設計介面 (API)。J2EE 元件透過呼叫 JNDI 查找方法來尋找物件。

本小節包含以下主題：

- JNDI 名稱和資源
- J2EE 命名服務
- 命名參考與連結資訊

JNDI 名稱和資源

JNDI 是 Java 命名和目錄介面 API 的首字母縮略。透過對此 API 進行呼叫，應用程式可以尋找資源和其他程式物件。資源是提供連線到系統的 (如資料庫伺服器和訊息傳送系統) 程式物件。(JDBC 資源有時被稱爲資料源。) 每個資源物件都是由唯一的易懂的名稱識別，稱爲 JNDI 名稱。Application Server 包含的命名和目錄服務將資源物件及其 JNDI 名稱連結在一起。若要建立新資源，需要將新的名稱 — 物件連結輸入到 JNDI 中。

J2EE 命名服務

JNDI 名稱是易懂的的物件名稱。這些名稱透過 J2EE 伺服器提供的命名和目錄服務連結到其物件。由於 J2EE 元件透過 JNDI API 存取此服務，因此物件通常使用其 JNDI 名稱。例如，Pointbase 資料庫的 JNDI 名稱爲 `jdbc/Pointbase`。當 Sun Java System Application Server 啓動時，將從配置檔案中讀取資訊，並自動將 JNDI 資料庫名稱增加到名稱空間。

需要 J2EE 應用程式用戶端、企業 Bean 與 Web 元件來存取 JNDI 命名環境。

應用程式元件的命名環境是一種機制，使用它可以在部署或組譯期間自訂應用程式元件的企業邏輯。使用應用程式元件的環境即可對應用程式元件進行自訂，而無需存取或變更應用程式元件的源代碼。

J2EE 容器實作應用程式元件的環境，並將該環境作爲 JNDI 命名環境提供給應用程式元件實例。應用程式元件的環境的使用方式如下：

- 應用程式元件的商業方法使用 JNDI 介面存取該環境。應用程式元件提供者在部署描述元中宣告應用程式元件需要其執行階段環境提供的所有環境項目。
- 容器提供儲存應用程式元件環境的 JNDI 命名環境的實作。容器還提供了部署程式可以用於建立和管理每個應用程式元件的環境的工具。
- 部署程式使用容器提供的工具，可以初始化應用程式元件的部署描述元中宣告的環境項目。部署程式可以設定和修改環境項目的值。
- 容器使環境命名環境在執行階段可用於應用程式元件實例。應用程式元件的實例使用 JNDI 介面獲取環境項目的值。

每個應用程式元件定義了其本身的環境項目集。一個應用程式元件在同一容器內的所有實例共用相同的環境項目。不允許應用程式元件實例在執行階段修改環境。

命名參考與連結資訊

資源參考是部署描述元中的元素，可以爲資源識別元件的編碼名稱。更具體地說，編碼名稱參考資源的連線工廠。在下一小節給出的範例中，資源參考名稱是 `jdbc/SavingsAccountDB`。

資源的 JNDI 名稱與資源參考的名稱是不同的。使用此命名方法，您需要在進行部署之前先對映這兩個名稱，但此方法也用於分離元件與資源。由於具有此分離功能，因此如果元件在以後需要存取其他資源，則無需變更名稱。這一靈活性使您可以更輕鬆地從預先存在的元件編譯 J2EE 應用程式。

表 10-1 列示了用於 Sun Java System Application Server 所使用的 J2EE 資源的 JNDI 查找及其關聯的參考。

表 10-1 JNDI 查找及其關聯的參考

JNDI 查找名稱	關聯的參考
java:comp/env	應用程式環境項目
java:comp/env/jdbc	JDBC DataSource 資源管理程式連線 Factory
java:comp/env/ejb	EJB 參考
java:comp/UserTransaction	UserTransaction 參考
java:comp/env/mail	JavaMail 階段作業連線 Factory
java:comp/env/url	URL 連線 Factory
java:comp/env/jms	JMS 連線 Factory 與目標
java:comp/ORB	應用程式元件共用的 ORB 實例

關於自訂資源

- 使用自訂資源
- 建立自訂資源
- 編輯自訂資源
- 刪除自訂資源
- 列示自訂資源

使用自訂資源

自訂資源存取本機 JNDI 儲存庫，外部資源存取外部 JNDI 儲存庫。這兩種類型的資源都需要使用者指定的工廠類別元素、JNDI 名稱屬性等。在本小節中，我們將討論如何爲 J2EE 資源配置 JNDI 連線工廠資源，以及如何存取這些資源。

在 Application Server 中，您可以建立、刪除和列示資源以及 list-jndi-entities。

建立自訂資源

若要建立自訂資源，請執行以下步驟：

1. 在管理主控台的左側窗格中，爲要修改的 JNDI 配置開啓 Sun Java System Application Server 實例。
2. 開啓 [JNDI] 標籤，並按一下 [自訂資源]。任何已經建立的自訂資源會在右側窗格中列示。若要建立新的自訂資源，按一下 [新建]。開啓 [JNDI] 標籤，並按一下 [新建]。將顯示一個頁面，用於增加新的自訂資源。
3. 在 [JNDI 名稱] 欄位中，輸入用於存取資源的名稱。此名稱將註冊到 JNDI 命名服務中。
4. 在 [資源類型] 欄位中，輸入完全合格的類型定義，如上面範例中所示。[資源類型] 定義遵循的格式爲 `xxx.xxx`。
5. 在 [工廠類別] 欄位中，爲要建立的自訂資源輸入工廠類別名稱。該欄位中的值是使用者指定的工廠類別的名稱。此類別實作 `javax.naming.spi.ObjectFactory` 介面。
6. 在 [描述] 欄位中，爲要建立的資源輸入描述。此描述是字串值，最多可以包含 250 個字元。
7. 在 [附加特性] 區段，增加特性名稱和值。
8. 標示 [啓用自訂資源] 核取方塊，以啓用自訂資源。
9. 按一下 [確定]，以儲存自訂資源。

如果已在叢集或獨立實例上部署自訂資源，則可以使用 [目標] 標籤管理目標。此標籤將在建立自訂資源之後顯示。透過輸入目標名稱並按一下 [確定] 來設定目標。

等效的 asadmin 指令爲：create-custom-resource。

編輯自訂資源

若要編輯自訂資源，請執行以下步驟：

1. 在管理主控台的左側窗格中，爲要修改的 JNDI 配置開啓 Sun Java System Application Server 實例。
2. 開啓 [JNDI] 並選取 [自訂資源]。任何已經建立的自訂資源會在右側窗格中列示。若要編輯自訂資源，請在右側窗格中按一下檔案名稱。
3. 編輯 [資源類型] 欄位、[工廠類別] 欄位或 [描述] 欄位。
4. 標示 [啓用自訂資源] 核取方塊，以啓用自訂資源。

5. 按一下 [儲存] 以儲存對自訂資源的變更。

刪除自訂資源

若要刪除自訂資源，請執行以下步驟：

1. 在管理主控台的左側窗格中，開啓 [JNDI] 標籤。
2. 按一下 [自訂資源]。任何已經建立的自訂資源會在右側窗格中列示。若要刪除自訂資源，請按一下要刪除的資源名稱旁邊的方塊。
3. 按一下 [刪除]。此自訂資源即被刪除。

等效的 asadmin 指令爲：delete-custom-resource。

列示自訂資源

若要列示自訂資源，請鍵入 `asadmin ist-custom-resources` 指令。例如，若要列示主機 plum 上的自訂資源，請鍵入以下指令：

```
$asadmin list-custom-resource --host plum target6
```

如需檢視完整的環境，請鍵入 `asadmin help list-custom-resources`。

關於外部 JNDI 儲存庫和資源

- 使用外部 JNDI 儲存庫和資源
- 建立外部資源
- 編輯外部資源
- 刪除外部資源
- 列示外部資源

使用外部 JNDI 儲存庫和資源

通常，Sun Java System Application Server 上執行的應用程式需要存取儲存在外部 JNDI 儲存庫中的資源。例如，一般的 Java 物件可能會以 Java 模式儲存在 LDAP 伺服器中。外部 JNDI 資源元素允許使用者配置此類外部資源儲存庫。外部 JNDI 工廠必須實作 `javax.naming.spi.InitialContextFactory` 介面。

使用外部 JNDI 資源的範例爲：

```
<resources>
<!-- external-jndi-resource 元素指定存取儲存在外部 JNDI 儲存庫中的
-- J2EE 資源的方式。下面的範例
-- 闡明存取儲存在 LDAP 中的 java 物件的方式。
-- factory-class 元素指定用來存取資源 Factory 所需的
-- JNDI InitialContext Factory。特性元素
-- 對應於外部 JNDI 環境可用的環境
-- 並且 jndi-lookup-name 指的是爲了擷取指定的 （在此例中爲
-- java）物件所要查找的 JNDI 名稱。
-->
<external-jndi-resource jndi-name="test/myBean"
jndi-lookup-name="cn=myBean"
res-type="test.myBean"
factory-class="com.sun.jndi.ldap.LdapCtxFactory">

<property name="PROVIDER-URL"
value="ldap://ldapserver:389/o=myObjects" />
<property name="SECURITY_AUTHENTICATION" value="simple" />
<property name="SECURITY_PRINCIPAL", value="cn=joeSmith,
o=Engineering" />
<property name="SECURITY_CREDENTIALS" value="changeit" />
</external-jndi-resource>
</resources>
```

建立外部資源

若要建立外部資源，請執行以下步驟：

1. 在管理主控台的左側窗格中，爲要修改的 JNDI 配置開啓 Sun Java System Application Server 實例。
2. 開啓 [JNDI] 並選取 [外部資源]。任何已經建立的外部資源會在右側窗格中列示。若要建立新的外部資源，按一下 [新建]。
3. 在 [JNDI 名稱] 欄位中，輸入用於存取資源的名稱。此名稱將註冊到 JNDI 命名服務中。

4. 在 [資源類型] 欄位中，輸入完全合格的類型定義，如上面範例中所示。[資源類型] 定義遵循的格式爲 `xxx.xxx`。
5. 在 [JNDI 查找] 欄位中，輸入要在外部儲存庫中查找的 JNDI 值。例如，在建立一個與外部儲存庫連線的外部資源時，爲了測試某個 Bean 類別，[JNDI 查找] 可能會如下所示：`cn=`*testmybean*。
6. 在 [工廠類別] 欄位中，輸入 JNDI 工廠類別外部儲存庫 (例如，`com.sun.jndi.ldap`)。此類別實作 javax.naming.spi. `ObjectFactory` 介面。
7. 在 [描述] 欄位中，爲要建立的資源輸入描述。此描述是字串值，最多可以包含 250 個字元。
8. 在 [附加特性] 區段，增加特性名稱和值。
9. 標示 [啓用外部資源] 核取方塊，以啓用外部資源。
10. 按一下 [確定] 以儲存外部資源。

如果已在叢集或獨立實例上部署外部資源，則可以使用 [目標] 標籤管理目標。此標籤將在建立外部資源之後顯示。透過輸入目標名稱並按一下 [確定] 來設定目標。

等效的 asadmin 指令爲：`create-jndi-resource`。

編輯外部資源

若要編輯外部資源，請執行以下步驟：

1. 在管理主控台的左側窗格中，爲要修改的 JNDI 配置開啓 Sun Java System Application Server 實例。
2. 開啓 [JNDI] 並選取 [外部資源]。任何已經建立的外部資源會在右側窗格中列示。若要編輯外部資源，請在右側窗格中按一下檔案名稱。
3. 編輯 [資源類型] 欄位、[JNDI 查找] 欄位、[工廠類別] 欄位或 [描述] 欄位。
4. 標示 [External Resource Enable] 核取方塊，以啓用外部資源。
5. 按一下 [儲存] 以儲存對外部資源的變更。

刪除外部資源

若要刪除外部資源，請執行以下步驟：

1. 在管理主控台的左側窗格中，開啓 [JNDI] 標籤。
2. 按一下 [外部資源]。任何已經建立的外部資源會在右側窗格中列示。若要刪除外部資源，請按一下要刪除的資源名稱旁邊的方塊。
3. 按一下 [刪除]。此外部資源即被刪除。

等效的 asadmin 指令爲 delete-jndi-resource。

列示外部資源

若要列示外部資源，請鍵入 `asadmin list-jndi-resources` 指令並指定 JNDI 名稱。例如，鍵入以下指令可以列示外部資源：

```
$asadmin list-jndi-resources -- target plum jndi_name_test
```

如需檢視完整的環境，請鍵入 `asadmin help list-jndi-resources`。

第 11 章

連接器資源

本章描述如何配置用於存取企業資訊系統 (EIS) 的連接器。它包含以下小節：

- 關於連接器
- 連接器連線池作業
- 連接器資源作業
- 受管理物件資源作業

關於連接器

- 連接器模組、連線池和資源

連接器模組、連線池和資源

連接器模組也稱爲資源介面，是允許應用程式與企業資訊系統 (EIS) 進行互動式操作的 J2EE 元件。EIS 軟體包含各種類型的系統：包括企業資源計劃 (ERP)、主機作業事件處理和非關聯式資料庫。類似其他 J2EE 模組，安裝連接器模組即是部署該連接器模組。

連接器連線池是一組用於特定 EIS 的可重複使用的連線。若要建立連接器連線池，請指定與池關聯的連接器模組 (資源介面)。

連接器資源是爲應用程式提供到 EIS 的連線的程式物件。若要建立連接器資源，請指定其 JNDI 名稱及其關聯的連線池。多個連接器資源可以指定單一連線池。應用程式可透過查找資源的 JNDI 名稱定位資源。(如需有關 JNDI 的更多資訊，請參閱 [JNDI 名稱和資源] 部分)。EIS 的連接器資源的 JNDI 名稱通常位於 `java:comp/env/`*`eis-specific`* 子環境中。

Application Server 使用連接器模組 (資源介面) 實作 JMS。請參閱 「JMS 資源與連接器資源之間的關係」。

連接器連線池作業

- 設定 EIS 存取的一般步驟
- 建立連接器連線池
- 編輯連接器連線池
- 刪除連接器連線池

設定 EIS 存取的一般步驟

1. 部署 (安裝) 連接器。請參閱 「部署連接器模組」。
2. 爲連接器建立連線池。請參閱 「建立連接器連線池」。
3. 建立與連線池關聯的連接器資源。請參閱 「建立連接器資源」。

建立連接器連線池

建立池之前，先部署與該池關聯的連接器模組 (資源介面)。爲新池指定的值取決於部署的連接器模組。

若要建立連接器連線池，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 選取 [連接器連線池] 節點。
3. 在 [連接器連線池] 頁面中，按一下 [新建]。
4. 在 [建立連接器連線池] 的第一個頁面中，指定以下設定：
   a. 在 [名稱] 欄位中，輸入池的邏輯名稱。

      建立連接器資源時將指定此名稱。

   b. 從 [資源介面] 組合方塊中選取一個項目。

      該組合方塊顯示已部署的資源介面 (連接器模組) 的清單。
5. 按一下 [下一步]。

6. 在 [建立連接器連線池] 的第二個頁面中，從 [連線定義] 組合方塊中選取一個值。

   組合方塊中的選擇取決於資源介面。通常，指定一種 ConnectionFactory (工廠實例) 以建立與 EIS 的連線。

7. 按一下 [下一步]。
8. 在 [建立連接器連線池] 的第三個也即最後一個頁面中，執行以下作業：

   a. 在 [一般設定] 區段中檢驗各個值是否正確。

   b. 對於 [池設定] 區段中的欄位，可以保留預設值。

      可以在以後變更這些設定。請參閱 「編輯連接器連線池」。

   c. 在 [附加特性] 表中新增任何所需的特性。

      在 [建立連接器連線池] 的上一個頁面中，從 [連線定義] 組合方塊中選取了一個類別。如果該類別位於伺服器的類別路徑中，則 [附加特性] 表將顯示預設特性。

9. 按一下 [完成]。

等效的 asadmin 指令爲：create-connector-connection-pool

- 連接器模組、連線池和資源
- 部署連接器模組

編輯連接器連線池

[編輯連接器連線池] 頁面使您可以變更池設定和附加特性。

若要存取 [編輯連接器連線池] 頁面，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 展開 [連接器連線池] 節點。
3. 選取要編輯的池的節點。
4. 在 [編輯連接器連線池] 頁面中，您可以變更控制池中連線的數目的設定。請參閱表 11-1。

表 11-1 連接器連線池的池設定

參數	描述
池的初始大小和最小大小	池中連線的最小數目。該值還確定了首次建立池或應用程式伺服器啟動時被置於池中的連線的數目。
池的最大大小	池中連線的最大數目。
儲存區調整容量數量	當池向最小池大小方向收縮時，將成批調整大小。此值確定批次中的連線數目。將該值設置過大會延遲連線循環；而將該值設置過小則會導致效率太低。
閒置逾時	連線在池中保持閒置狀態的最長時間 (以秒為單位)。一旦超過此時間，即從池中移除該連線。
最長等待時間	已請求連線的應用程式在達到連線逾時之前等待的時間。由於預設等待時間過長，應用程式可能會出現無限期當機的情況。
一旦失敗	選取標記為 [關閉所有連線] 的核取方塊之後，如果單一連線失敗，應用程式伺服器將關閉池中的所有連線，然後重新建立這些連線。如果未選取此核取方塊，則只有要使用各個連線時才會重新建立這些連線。
作業事件支援	使用 [作業事件支援] 清單可以為連線池選取作業事件支援類型。選擇的作業事件支援將以向下相容方式置換與此連線池關聯的資源介面中的作業事件支援屬性。也就是說，它可以支援比資源介面中指定的作業事件層級低或與其相同的作業事件層級，但它不能指定更高的層級。 作業事件支援選項包括以下內容： [作業事件支援] 功能表中的 [無] 選項表示資源介面不支援本機資源管理員或 JTA 作業事件，也不實作 XAResource 或 LocalTransaction 介面。 [本機] 作業事件支援表示資源介面將透過實作 LocalTransaction 介面來支援本機作業事件。本機作業事件的管理在資源管理員內部進行，不涉及任何外部作業事件管理員。 [XA] 作業事件支援表示資源介面將透過實作 LocalTransaction 和 XAResource 介面來支援本機資源管理員和 JTA 作業事件。XA 作業事件由作業事件管理員在資源管理員外部進行控制和協調。本機作業事件的管理在資源管理員內部進行，不涉及任何外部作業事件管理員。

5. 在 [附加特性] 表中，指定名稱 - 值對。指定的特性取決於此池使用的資源介面。部署程式使用此表指定的名稱 - 值對可用於置換由資源介面供應商定義的特性的預設值。

6. 在 [安全性對映] 標籤窗格中，建立或修改用於連線池的安全性對映。請參閱「關於安全性對映」，以取得有關如何建立安全對映的資訊。

7. 按一下 [儲存]。

刪除連接器連線池

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 選取 [連接器連線池] 節點。
3. 在 [連接器連線池] 頁面中，選取要刪除的池的核取方塊。
4. 按一下 [刪除]。

等效的 asadmin 指令爲：delete-connector-connection-pool

連接器資源作業

- 建立連接器資源
- 編輯連接器資源
- 刪除連接器資源
- 配置連接器服務

建立連接器資源

連接器資源 (資料源) 爲應用程式提供 EIS 連線。建立連接器資源之前，請先建立連接器連線池。

若要建立連接器資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 展開 [連接器資源] 節點。
3. 在 [連接器資源] 頁面中，按一下 [新建]。
4. 在 [建立連接器資源] 頁面中，指定資源的設定：
   a. 在 [JNDI 名稱] 欄位中，鍵入唯一的名稱，例如：eis/myERP。請不要忘記正斜線。
   b. 從 [池名稱] 組合方塊中選擇新連接器資源所屬的連線池。

c. 依預設，建立資源之後立即可以使用資源 (已啓用)。若要將資源變更爲不可用，請選取 [在所有目標上停用] 單選按鈕。

d. 在此頁面的 [目標] 區段中，從 [可用] 欄位中選取連接器資源所在的網域、叢集或伺服器實例，然後按一下 [新增]。如果不想將連接器資源部署到 [已選取] 欄位中列出的某個網域、叢集或伺服器實例，請從欄位中選取該實例，然後按一下 [移除]。

5. 按一下 [確定]。

等效的 asadmin 指令爲：create-connector-resource

編輯連接器資源

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 展開 [連接器資源] 節點。
3. 選取要編輯的連接器資源的節點。
4. 在 [編輯連接器資源] 頁面中，您可以從 [池名稱] 功能表中選取不同的連線池。
5. 在 [目標] 標籤窗格中，可以透過按一下 [管理目標] 來編輯連接器資源部署到的目標。請參閱「建立連接器資源」，以取得有關目標的更多資訊。
6. 按一下 [儲存] 以套用編輯。

刪除連接器資源

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 選取 [連接器資源] 節點。
3. 在 [連接器資源] 頁面中，選取要刪除的資源的核取方塊。
4. 按一下 [刪除]。

等效的 asadmin 指令爲：delete-connector-resource

配置連接器服務

使用 [連接器服務] 螢幕可以爲部署到此叢集或伺服器實例的所有資源介面配置連接器容器。

若要配置連接器容器，請執行以下步驟：

1. 從樹中選擇 [配置]。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 server，請選擇 [server-config] 節點。
   b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 [default-config] 節點。
3. 選取 [連接器服務] 節點。
4. 在 [關閉逾時] 欄位中指定關閉逾時 (以秒爲單位)。請輸入一個整數，此整數表示應用程式伺服器可以等待連接器模組的實例的 `ResourceAdapter.stop` 方法完成的秒數。應用程式伺服器將忽略所需時間超過指定關閉逾時的的資源介面，並且關閉程序將繼續。預設的關閉逾時爲 30 秒。按一下 [載入預設值]，可以爲部署到此叢集或伺服器實例的資源介面選取預設關閉逾時。

受管理物件資源作業

- 建立受管理物件資源
- 編輯受管理物件資源
- 刪除受管理物件資源

建立受管理物件資源

封裝在資源介面 (連接器模組) 中的受管理物件爲應用程式提供專用功能。例如，受管理物件可以提供對特定於資源介面及其關聯的 EIS 的剖析器的存取。物件可以被管理，即物件可以由管理員配置。若要配置物件，請在 [建立受管理物件資源] 頁面或 [編輯受管理物件資源] 頁面中新增名稱 - 值特性對。建立受管理物件資源時，請將受管理物件與 JNDI 名稱相關聯。

Application Server 使用資源介面實作 JMS。對於建立的每個 JMS 目標，Application Server 都會自動建立一個受管理物件資源。

若要建立受管理物件資源，請執行以下步驟：

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 展開 [受管理物件資源] 節點。
3. 在 [受管理物件資源] 頁面中，按一下 [新建]。
4. 在 [受管理物件資源] 頁面中，指定以下設定：
   a. 在 [JNDI 名稱] 欄位中，鍵入識別資源的唯一名稱。
   b. 在 [資源類型] 欄位中，輸入資源的 Java 類型。
   c. 從 [資源介面] 組合方塊中，選取包含受管理物件的資源介面。
   d. 選取或取消選取核取方塊以啓用或停用資源。
   e. 按一下 [下一步]。
5. 在 [建立受管理物件資源] 的第二個頁面中，可以執行以下作業。
   a. 若要使用名稱 - 值特性對配置受管理物件，請按一下 [新增特性]。
   b. 在此頁面的 [目標] 區段中，從 [可用] 欄位中選取受管理物件所在的網域、叢集或伺服器實例，然後按一下 [新增]。如果要取消部署受管理物件到 [已選取] 欄位中列出的某個網域、叢集或伺服器實例，請從欄位中選擇該實例，然後按一下 [移除]。
6. 按一下 [完成]。

等效的 asadmin 指令爲：`create-admin-object`

編輯受管理物件資源

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 展開 [受管理物件資源] 節點。
3. 選取要編輯的受管理物件資源的節點。
4. 在 [編輯受管理物件資源] 頁面中，修改在 [建立受管理物件資源] 中指定的值。
5. 在 [目標] 標籤窗格中，可以透過按一下 [管理目標] 來編輯受管理物件部署到的目標。請參閱「建立受管理物件資源」，以取得有關目標的更多資訊。
6. 按一下 [儲存] 以套用編輯。

刪除受管理物件資源

1. 在樹形元件中，展開 [資源] 節點，然後展開 [連接器] 節點。
2. 選取 [受管理物件資源] 節點。
3. 在 [受管理物件資源] 頁面中，選取要刪除的資源的核取方塊。
4. 按一下 [刪除]。

等效的 asadmin 指令爲：delete-admin-object

第 12 章

管理已命名的配置

本章描述在 Application Server 中新增、變更以及使用已命名的伺服器配置。它包含以下小節：

- 關於已命名的配置
- 用於已命名的配置的管理主控台作業

關於已命名的配置

- 已命名的配置
- default-config 配置
- 建立實例或叢集時建立的配置
- 唯一連接埠號和配置

已命名的配置

已命名的配置是一組伺服器配置資訊。此資訊包括針對以下內容的配置設定：HTTP 偵聽程式、orb/iiop 偵聽程式、JMS 代理程式、EJB 容器、安全性、記錄和監視功能。已命名的配置中未定義應用程式和資源。

配置是在管理網域中建立的。該網域中的多個伺服器實例或叢集可以參考相同的配置，也可以有各自獨立的配置。

對於叢集，叢集中的所有伺服器實例都繼承叢集的配置，從而確定叢集實例具有同質環境。

由於已命名的配置包含如此多的必需配置設定，因此請透過複製現有已命名的配置來建立新配置。變更新建配置的配置設定之前，該配置與被複製的配置完全相同。

根據叢集或實例對配置使用方式的不同，可以將叢集和實例分爲三類：

- **獨立**。獨立伺服器實例或叢集不與其他伺服器實例共用其配置；也就是說，其他伺服器實例或叢集不參考獨立伺服器實例或叢集的已命名的配置。
- **共用**。共用伺服器實例或叢集與其他伺服器實例或叢集共用配置；也就是說，多個實例或叢集參考相同的已命名的配置。
- **叢集**。叢集伺服器實例繼承叢集的配置。

default-config 配置

`default-config` 配置是一種特殊配置，用作建立獨立伺服器實例或獨立叢集配置的範本。非叢集伺服器實例或叢集不允許參考 `default-config` 配置；只能對其進行複製以建立新配置。編輯預設配置，以確定從預設配置複製而來的新配置具有正確的初始設定。

建立實例或叢集時建立的配置

建立新伺服器實例或新叢集時，可以執行以下操作之一：

- 參考現有配置。不新增新配置。
- 建立現有配置的副本。新增伺服器實例或叢集時，將新增新配置。

依預設，在建立新叢集或實例時，其配置是從 `default-config` 配置中複製的。若要從其他配置進行複製，請在建立新實例或叢集時指定要複製的配置。

對於伺服器實例，新配置的名稱爲 *instance_name*`-config`。對於叢集，新配置的名稱爲 *cluster-name*`-config`。

唯一連接埠號和配置

如果同一主機電腦上有多個實例參考相同的配置，則每個實例必須在唯一的連接埠號上進行偵聽。例如，如果兩個伺服器實例都參考某個已命名的配置，並且該配置包含一個位於連接埠 80 上的 HTTP 偵聽程式，則連接埠衝突將阻止其中一個伺服器實例啓動。變更用於定義連接埠號 (各個伺服器實例在這些連接埠號上進行偵聽) 的特性，從而確定各個實例使用唯一的連接埠。

以下原則適用於連接埠號設定：

- 各個伺服器實例的連接埠號最初是從配置繼承而來的。

- 建立伺服器實例時，如果該連接埠號已經被使用，則請在實例層級上置換繼承的預設值，以防止發生連接埠衝突。
- 假設實例正在共用配置。配置具有連接埠號 n。如果使用相同的配置在機器上建立新實例，則爲新實例指定的連接埠號爲 n+1 (如果此連接埠號可用)。如果此連接埠號不可用，將選擇 n+1 後下一個可用的連接埠。
- 如果您變更了配置的連接埠號，則繼承該連接埠號的伺服器實例將自動繼承變更後的連接埠號。
- 如果您變更了實例的連接埠號，然後又變更了配置的連接埠號，則實例的連接埠號將保持不變。

用於已命名的配置的管理主控台作業

- 建立已命名的配置
- 編輯已命名的配置的特性
- 編輯參考配置的實例的連接埠號
- 檢視已命名的配置的目標
- 刪除已命名的配置

建立已命名的配置

若要建立已命名的配置，請執行以下步驟：

1. 在樹形元件中，選擇 [配置] 節點。
2. 在 [配置] 頁面中，按一下 [新建]。
3. 在 [建立配置] 頁面中，爲配置輸入唯一名稱。
4. 選擇要複製的配置。

   `default-config` 配置是建立獨立伺服器實例或獨立叢集時所使用的預設配置。

等效的 `asadmin` 指令爲：`copy-config`。

編輯已命名的配置的特性

若要編輯已命名的配置的特性，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取已命名的配置的節點。
3. 在 [配置系統特性] 頁面中，選擇是否啓用動態重新配置。

   如果已啓用，則對配置所作的變更將套用至伺服器實例，而無需重新啓動伺服器。
4. 新增特性、變更特性的目前值或刪除特性。

   連接埠是已定義的特性。如果系統中存在多個伺服器實例，則它們各自的連接埠號必須是唯一的。

表 12-1 列出了預先定義的特性及其說明。

表 12-1 已命名的配置的特性

特性名稱	描述
HTTP_LISTENER_PORT	此特性指定 http-listener-1 的連接埠號。有效值為 1 到 65535。在 UNIX 中，建立在 1 到 1024 連接埠上進行偵聽的套接字要求具有超級使用者權限。
HTTP_SSL_LISTENER_PORT	此特性指定 http-listener-2 的連接埠號。有效值為 1 到 65535。在 UNIX 中，建立在 1 到 1024 連接埠上進行偵聽的套接字要求具有超級使用者權限。
IIOP_SSL_LISTENER_PORT	此特性指定稱為 SSL 的 IIOP 偵聽程式偵聽 IIOP 連線的 ORB 偵聽程式連接埠。
IIOP_LISTENER_PORT	此特性指定 orb-listener-1 偵聽 IIOP 連線的 ORB 偵聽程式連接埠。
JMX_SYSTEM_CONNECTOR_PORT	此特性指定 JMX 連接器進行偵聽的連接埠號。有效值為 1 到 65535。在 UNIX 中，建立在 1 到 1024 連接埠上進行偵聽的套接字要求具有超級使用者權限。
IIOP_SSL_MUTUALAUTH_PORT	此特性指定稱為 SSL_MUTUALAUTH 的 IIOP 偵聽程式偵聽 IIOP 連線的 ORB 偵聽程式連接埠。

5. 若要編輯與配置關聯的所有實例的特性的目前值，請按一下 [實例值]。

等效的 `asadmin` 指令爲：`set`。

編輯參考配置的實例的連接埠號

每個參考已命名的配置的實例最初都從該配置繼承連接埠號。由於系統中的連接埠號必須是唯一的，因而可能需要置換繼承的連接埠號。

1. 在樹形元件中，展開 [配置] 節點。
2. 選取已命名的配置的節點。
3. 在 [配置系統特性] 頁面中，按一下要編輯的連接埠號旁邊的 [實例值]。

   例如，如果按一下 SSL-port 特性旁邊的 [實例值]，將看到參考該配置的每個伺服器實例的 SSL-port 值。
4. 變更連接埠值，然後按一下 [儲存]。

等效的 `asadmin` 指令爲：`set`。

檢視已命名的配置的目標

若要檢視已命名的配置的目標，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取已命名的配置的節點。

   [配置系統特性] 頁面將顯示使用該配置的所有目標的清單。對於叢集配置，這些目標是叢集。對於實例配置，這些目標是實例。

刪除已命名的配置

若要刪除已命名的配置，請執行以下步驟：

1. 在樹形元件中，選取 [配置] 節點。
2. 在 [配置] 頁面中，選取要刪除的已命名的配置的核取方塊。

   不能刪除 `default-config` 配置。
3. 按一下 [刪除]。

等效的 `asadmin` 指令爲：`delete-config`。

第 13 章

J2EE 容器

本章說明如何配置伺服器中包含的 J2EE 容器。它包含以下小節：

- 關於 J2EE 容器
- 有關 J2EE 容器的管理主控台作業

關於 J2EE 容器

本小節介紹 Application Server 中包含的 J2EE 容器。

- J2EE 容器的類型
- Web 容器
- EJB 容器

J2EE 容器的類型

J2EE 容器爲 J2EE 應用程式元件提供執行階段支援。J2EE 應用程式元件使用容器的協定和方法存取伺服器提供的其他應用程式元件和服務。Application Server 提供應用程式用戶端容器、applet 容器、Web 容器和 EJB 容器。如需有關顯示容器的圖解，請參閱 「Application Server 架構」小節。

Web 容器

Web 容器是容納 Web 應用程式的 J2EE 容器。Web 容器透過爲開發者提供執行 Servlet 和 JavaServer Page (JSP) 的環境，延伸了 Web 伺服器的功能。

EJB 容器

企業 Bean (EJB 元件) 是包含商務邏輯的 Java 程式設計語言伺服器元件。EJB 容器提供對企業 Bean 的本機和遠端存取。

企業 Bean 分爲三種類型：階段作業 Bean、實體 Bean 和訊息導引 Bean。階段作業 Bean 表示暫態物件和程序，並且通常由單一用戶端使用。實體 Bean 表示持續性資料，通常維護在資料庫中。訊息導引 Bean 用於將訊息非同步傳送到應用程式模組和服務中。

容器負責建立企業 Bean、將企業 Bean 連結至命名服務以使其他應用程式元件可以存取企業 Bean、確定僅授權的用戶端才能存取企業 Bean 的方法、將 Bean 的狀態儲存到持續性儲存中、快取 Bean 的狀態以及在必要時啓動或鈍化 Bean。

有關 J2EE 容器的管理主控台作業

- 配置一般 Web 容器設定
- 配置一般 EJB 設定
- 配置訊息導引 Bean 設定
- 配置 EJB 計時器服務設定

配置一般 Web 容器設定

在此發行版本中，管理主控台中沒有用於 Web 容器的容器範圍的設定。

配置 Web 容器階段作業

本小節描述 Web 容器中的 HTTP 階段作業設定。HTTP 階段作業是唯一將狀態資料寫入持續性儲存的 Web 階段作業。

若要設定階段作業逾時值，請執行以下步驟：

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。

b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。

3. 選取 [Web 容器] 節點。

4. 按一下 [階段作業特性] 標籤。

5. 在 [階段作業逾時] 欄位中，輸入階段作業有效的秒數。

6. 按一下 [儲存]。

配置管理員特性

階段作業管理程式提供配置如何建立和銷毀階段作業、儲存階段作業狀態的位置以及最大階段作業數的方法。

若要變更階段作業管理員設定，請執行以下步驟：

1. 在樹形元件中，選取 [配置] 節點。

2. 選取要配置的實例：

a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。

b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。

3. 選取 [Web 容器] 節點。

4. 按一下 [管理員特性] 標籤。

5. 設定 [清除間隔] 的值。

[清除間隔] 欄位是從儲存中刪除非作用中的階段作業資料之前的秒數。

6. 設定 [最大階段作業數] 的值。

[最大階段作業數] 欄位是允許的最大階段作業數目。

7. 設定 [階段作業檔案名稱] 的值。

[階段作業檔案名稱] 欄位是包含階段作業資料的檔案。

8. 設定 [階段作業 ID 產生器類別名稱] 的值。

   [階段作業 ID 產生器類別名稱] 欄位使您可以指定用於產生唯一的階段作業 ID 的自訂類別。每個伺服器實例只允許有一個階段作業 ID 產生器類別，並且叢集中的所有實例必須使用同一階段作業 ID 產生器，以防止階段作業金鑰衝突。

   自訂階段作業 ID 產生器類別必須執行 `com.sun.enterprise.util.uuid.UuidGenerator` 介面：

```
package com.sun.enterprise.util.uuid;

public interface UuidGenerator {

    public String generateUuid();
    public String generateUuid(Object obj);  //obj is the session object

}
```

   類別必須位於 Application Server 類別路徑中。

9. 按一下 [儲存]。

配置儲存特性

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 選取 [Web 容器] 節點。
4. 按一下 [儲存特性] 標籤。
5. 設定 [清除間隔]。

   [清除間隔] 欄位是從儲存中刪除非作用中的階段作業資料之前的秒數。

6. 按一下 [儲存]。

配置一般 EJB 設定

本小節描述以下適用於伺服器上所有企業 Bean 容器的設定：

- 階段作業儲存位置
- 儲存區設定
- 快取設定

若要置換每個容器的預設值，請調整企業 Bean 的 `sun-ejb-jar.xml` 檔案中的值。如需詳細資訊，請參閱「Application Server Developer's Guide」。(有關指向此指南的連結，請參閱「更多資訊」。)

階段作業儲存位置

[階段作業儲存位置] 欄位指定在檔案系統上儲存鈍化 Bean 和持續的 HTTP 階段作業所在的目錄。

鈍化 Bean 是已將其狀態寫入到檔案系統上的檔案中的企業 Bean。通常，鈍化 Bean 已經閒置了某特定時間段的時間並且目前未被用戶端存取。

與鈍化 Bean 類似，持續的 HTTP 階段作業是已將其狀態寫入到檔案系統上的檔案中的各個 Web 階段作業。

[確定選項] 欄位用於指定容器如何快取作業事件之間的鈍化實體 Bean 實例。

[選項 B] 用於快取作業事件之間的實體 Bean 實例，並且是預設選項。[選項 C] 用於停用快取。

池設定

容器維護了一個企業 Bean 池，以便在不建立 Bean 來實現效能的情況下回應用戶端請求。這些設定僅適用於無狀態階段作業 Bean 和實體 Bean。

如果在使用已部署的企業 Bean 的應用程式中遇到效能問題，建立池或增加現有池維護的 Bean 的數目有助於提高應用程式的效能。

依預設，容器維護企業 Bean 池。

若要調整容器的企業 Bean 池的配置，請執行以下步驟：

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
    a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。

b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。

3. 選取 [EJB 容器] 節點。
4. 在 [池設定] 下的 [池的初始大小和最小大小] 欄位中，輸入容器在池中建立的 Bean 的最小數目。
5. 在 [池的最大大小] 欄位中，輸入任何時候容器在池中維護的 Bean 的最大數目。
6. 如果 Bean 處於閒置狀態的時間超過 [池閒置逾時] 中指定的時間，在 [池設定大小數量] 欄位中輸入要從池中移除的 Bean 的數目。
7. 在 [池閒置逾時] 欄位中輸入在將池中的 Bean 從池中移除之前 Bean 可以保持閒置狀態的時間 (以秒爲單位)。
8. 按一下 [儲存]。
9. 重新啓動 Application Server。

快取設定

容器維護最常用企業 Bean 的企業 Bean 資料快取。這樣，容器可以更快回應來自其他應用程式模組的企業 Bean 資料請求。本小節只適用於有狀態階段作業 Bean 和實體 Bean。

被快取的企業 Bean 處於以下三種狀態之一：使用中、閒置或鈍化。使用中企業 Bean 是目前正被用戶端存取的企業 Bean。閒置企業 Bean 的資料目前保存在快取中，但沒有用戶端存取 Bean。鈍化 Bean 的資料是被暫時儲存的，如果用戶端請求此 Bean，其資料將被讀回快取中。

若要調整被快取的企業 Bean 的設定，請執行以下步驟：

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：

   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。

   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 選取 [EJB 容器] 節點。
4. 在 [最大快取大小] 欄位中調整最大快取大小。

   增加要快取的 Bean 的最大數目，以結束建立和損毀 Bean 的系統耗用。但是，如果增加快取，伺服器將消耗更多記憶體和資源。請確定作業環境足夠用於快取設定。

5. 在 [快取設定大小數量] 欄位中調整快取設定大小數量。

   達到快取的 Bean 的最大數目之後，容器將從備份儲存中移除一些鈍化 Bean，預設設定爲 32。

6. 在 [快取閒置逾時] 欄位中，調整爲實體 Bean 排程的快取清除速率 (以秒爲單位)。

   如果快取的實體 Bean 在特定時間內一直處於閒置狀態，它將被鈍化。即將 Bean 的狀態寫入備份儲存。

7. 在 [移除逾時] 欄位中，調整將有狀態階段作業 Bean 從快取或鈍化儲存中移除之前的時間 (以秒爲單位)。

8. 在 [移除選取策略] 欄位中，配置容器用來移除有狀態階段作業 Bean 的策略。

   容器將根據在 [移除選取策略] 欄位中設定的策略決定移除哪個有狀態階段作業 Bean。容器可使用三種可能的策略從快取中移除 Bean：

   - 最近未使用 (NRU)
   - 先入先出 (FIFO)
   - 最近最少使用 (LRU)

   NRU 策略移除最近未使用的 Bean。FIFO 策略移除快取中最早的 Bean。LRU 策略移除最近最少存取的 Bean。依預設，容器使用 NRU 策略。

   通常使用 FIFO 策略移除實體 Bean。

9. 按一下 [儲存]。

10. 重新啓動 Application Server。

配置訊息導引 Bean 設定

訊息導引 Bean 的池與「配置一般 EJB 設定」中介紹的階段作業 Bean 的池類似。

依預設，容器維護訊息 Bean 池。

若要調整該池的配置，請執行以下步驟：

1. 在樹形元件中，選擇 [配置] 節點。

2. 選擇要配置的實例：

   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。

   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。

3. 選取 [EJB 容器] 節點。
4. 按一下 [MDB 設定] 標籤。
5. 在 [池設定] 下的 [初始和最小池大小] 欄位中，輸入容器在池中建立的訊息 Bean 的最小數目。
6. 在 [最大池大小] 欄位中，輸入任何時候容器在池中維護的 Bean 的最大數目。
7. 如果 Bean 處於閒置狀態的時間超過 [池閒置逾時] 中指定的時間，在 [池設定大小數量] 欄位中輸入要從池中移除的 Bean 的數目。
8. 在 [池閒置逾時] 欄位中輸入在將池中的 Bean 從池中移除之前 Bean 可以保持閒置狀態的時間 (以秒爲單位)。
9. 按一下 [儲存]。
10. 重新啓動 Application Server。

配置 EJB 計時器服務設定

計時器服務是由企業 Bean 容器提供的用於排程企業 Bean 使用的通知或事件的持續性和作業事件通知服務。所有企業 Bean (有狀態階段作業 Bean 除外) 均可從計時器服務接收通知。關閉或重新啓動伺服器時，服務設定的計時器不會被銷毀。

配置計時器服務

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 選取 [EJB 容器] 節點。
4. 按一下 [EJB 計時器服務] 標籤。
5. 在 [最小傳送間隔] 欄位中設定最小傳送間隔時間 (以毫秒爲單位)。最小傳送間隔是特定計時器的下一個計時器到期之前允許的最小毫秒數。如果此間隔時間設定得過小，可能會導致伺服器超載。
6. 在 [最大重新傳送次數] 欄位中，設定計時器服務嘗試傳送通知的最大次數。
7. 在 [重新傳送間隔] 欄位中，設定兩次重新傳送嘗試之間的間隔時間 (以毫秒爲單位)。

8. 按一下 [儲存]。
9. 重新啓動 Application Server。

將外部資料庫與計時器服務一起使用

依預設，計時器服務使用內嵌式資料庫儲存計時器。

若要使用外部資料庫儲存計時器，請執行以下步驟：

1. 按照第 129 頁的「建立 JDBC 資源」中所述爲資料庫設定 JDBC 資源。
2. 在 [計時器資料源] 欄位中輸入資源的 JNDI 名稱。
3. 按一下 [儲存]。
4. 重新啓動 Application Server。

在 <*INSTALL_DIR*>/lib/install/databases/ 中提供了 PointBase 和 Oracle 的計時器資料庫建立檔案範例。

第 14 章

配置安全性

本章描述了某些核心應用程式伺服器的安全性概念，以及如何爲 Sun Java System Application Server 8.1 2005Q1 配置安全性。本章包含以下主題：

- 關於 Application Server 安全性
- 用於安全性的管理主控台作業
- 用於範圍的管理主控台作業
- 有關 JACC 提供者的管理主控台作業
- 有關稽核模組的管理主控台作業
- 用於訊息安全性的管理主控台作業
- 有關偵聽程式和 JMX 連接器的管理主控台作業
- 有關連接器連線池的管理主控台作業
- 使用證書和 SSL
- 更多資訊

關於 Application Server 安全性

- 安全性概況
- 關於認證與授權
- 瞭解使用者、群組、角色和範圍
- 證書和 SSL 簡介
- 關於防火牆
- 使用管理主控台管理安全性

安全性概況

安全性實質上就是資料保護：如何在儲存或傳輸資料時防止對資料進行未授權的存取或破壞。Application Server 具有基於 J2EE 標準的動態可延伸安全性架構。它內置的安全性功能包括密碼學、認證與授權以及公開金鑰基礎架構。Application Server 是基於 Java 安全性模型建置的，該安全性模型使用了一個沙箱，應用程式可以在沙箱中安全地執行，而不會給系統或使用者帶來潛在的危險。本節論述以下主題：

- 瞭解應用程式和系統安全性
- 管理安全性的工具
- 管理密碼安全性
- 指定安全性職責

瞭解應用程式和系統安全性

從廣泛意義上講，應用程式安全性有兩種：

- 在**程式安全性**中，開發者編寫的應用程式代碼負責處理安全性事務。作爲管理員，您對此機制沒有任何控制權。由於程式化安全性將安全性配置硬編碼到應用程式中而不是透過 J2EE 容器對其進行管理，因此一般不提倡使用這種程式化安全性。
- 在**宣告性安全性**中，容器 (Application Server) 透過應用程式的部署描述元處理安全性。您可以透過直接編輯部署描述元或使用 `deploytool` 等工具來控制宣告性安全性。由於可以在完成應用程式開發之後變更部署描述元，因此宣告性安全性具有更大靈活性。

除了應用程式安全性以外，還有影響 Application Server 系統中所有應用程式的**系統安全性**。

程式安全性受應用程式開發者的控制，因此本文件不對其進行論述；宣告性安全性受應用程式開發者的控制要少一些，本文件中只偶爾涉及到宣告性安全性。本文件主要針對系統管理員，因此主要論述系統安全性。

管理安全性的工具

Application Server 提供了以下用於管理安全性的工具：

- 管理主控台，一種基於瀏覽器的工具，用於配置整個伺服器的安全性，管理使用者、群組和範圍以及執行系統範圍內的其他安全性作業。如需有關管理主控台的一般介紹，請參閱「管理工具」。如需有關使用管理主控台可以執行的安全性作業的概況，請參閱「使用管理主控台管理安全性」。

- asadmin，指令行工具，可以執行管理主控台能夠執行的許多作業。您還可以使用 asadmin 執行某些使用管理主控台無法執行的作業。您可以從指令提示符號或在程序檔中執行 asadmin 指令，以自動執行重複的作業。如需有關 asadmin 的一般介紹，請參閱 「管理工具」。
- deploytool，圖形化封裝和部署工具，用於編輯應用程式部署描述元，從而控制各個應用程式的安全性。由於 deploytool 主要針對應用程式開發者，因此本文件未對該工具的使用作詳細說明。如需有關使用 deploytool 的說明，請參閱此工具的線上說明以及位於 http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html 上的「The J2EE 1.4 Tutorial」。

Java 2 Platform, Standard Edition (J2SE) 提供了兩個用於管理安全性的工具：

- keytool，指令行公用程式，用於管理數位證書和金鑰對。使用 keytool 可以管理 certificate 範圍內的使用者。
- policytool，圖形化公用程式，用於管理系統範圍內的 Java 安全策略。作爲管理員，您很少會用到 policytool。

如需有關使用 keytool、policytool 和其他 Java 安全性工具的更多資訊，請參閱位於 http://java.sun.com/j2se/1.4.2/docs/tooldocs/tools.html#security 上的「Java 2 SDK Tools and Utilities」。

在 Enterprise Edition 中，還可以使用另外兩個實作網路安全性服務 (NSS) 的工具來管理安全性。如需有關 NSS 的更多資訊，請造訪 http://www.mozilla.org/projects/security/pki/nss/。用於管理安全性的工具包括：

- certutil，指令行公用程式，用於管理證書和金鑰資料庫。
- pk12util，指令行公用程式，用於以 PKCS12 格式在證書 / 金鑰資料庫和檔案之間匯入和匯出金鑰及證書。

如需有關使用 certutil、pk12util 和其他 NSS 安全性工具的更多資訊，請參閱位於 http://www.mozilla.org/projects/security/pki/nss/tools 上的「NSS Security Tools」。

管理密碼安全性

在此發行版本的 Application Server 中，包含特定網域規格的 domain.xml 檔案最初包含純文字形式的 IMQ 代理程式密碼。domain.xml 檔案中包含此密碼的元素爲 jms-host 元素的 admin-password 屬性。由於在安裝期間不能變更此密碼，因此它不會對安全性產生很大的影響。

但是，您可以使用管理主控台增加使用者和資源，並爲這些使用者和資源指定密碼。部分密碼將以純文字形式寫入 domain.xml 檔案，例如用於存取資料庫的密碼。將這些純文字形式的密碼保存在 domain.xml 檔案中可能會破壞安全性。透過執行以下步驟，您可以加密 domain.xml 中的任何密碼，包括 admin-password 屬性或資料庫密碼：

1. 在 domain.xml 檔案所在的目錄 (依預設，此目錄爲 *install_dir*/domains/*domain_dir*/config) 中，執行以下 asadmin 指令：

   ```
   asadmin create-password-alias <alias-name>
   ```

   例如，

   ```
   asadmin create-password-alias jms-password
   ```

   將顯示輸入密碼提示 (在本例中爲 admin)。請參閱 create-password-alias、list-password-aliases 和 delete-password-alias 指令的線上援助頁，以取得更多資訊。

2. 移除並取代 domain.xml 中的密碼。使用 asadmin set 指令可以完成此作業。使用 set 指令實現此目的的範例如下：

   ```
   asadmin set
   server.jms-service.jms-host.default_JMS_host.admin-password=${ALIAS=jms
   -password}
   ```

3. 重新啓動相關網域的 Application Server。

保護具有編碼密碼的檔案

某些檔案包含需要使用檔案系統許可權進行保護的編碼密碼。這些檔案包括：

- *install_dir*/domains/*domain_dir*/master-password

  此檔案包含編碼主密碼，並且應使用檔案系統許可權 600 對其進行保護。

- 對於使用 --passwordfile 引數作爲引數傳送給 asadmin 的任何已建立密碼檔案均應使用檔案系統許可權 600 對其進行保護。

變更主密碼

主密碼 (MP) 是全局性的共用密碼。它從不用於認證，也從不會在網路上傳輸。此密碼是全局安全性的要塞點；使用者可以選擇在需要時手動輸入此密碼，也可以將其隱藏在檔案中。它是系統中最敏感的資料。使用者可以透過移除此檔案強制系統提示輸入 MP。變更主密碼後，系統會將其重新儲存到主密碼金鑰存放區中。

若要變更主密碼，必須執行以下步驟：

1. 停止網域的 Application Server。使用 asadmin 指令 change-master-password 提示輸入舊密碼和新密碼，然後重新加密所有相依項目。例如，

```
asadmin change-master-password>
請輸入主密碼 >
請輸入新的主密碼 >
請再次輸入新的主密碼 >
```

2. 重新啓動 Application Server。

警告：此時，不能啓動正在執行的伺服器實例並且不能重新啓動正在執行的伺服器實例，除非已變更這些實例所對映的節點代理程式上的 SMP。如果在變更伺服器實例的 SMP 之前重新啓動了該伺服器實例，它將無法啓動。

3. 一次停止一個節點代理程式及其相關伺服器。再次執行 `asadmin change-master-password` 指令，然後重新啓動節點代理程式及其相關伺服器。
4. 繼續對下一個節點代理程式執行這些步驟，直到對所有節點代理程式均已執行這些步驟。這樣可以完成回轉變更。

變更管理員密碼

「管理密碼安全性」中論述了如何加密管理員密碼。強烈建議您加密管理員密碼。如果要在加密管理員密碼之前變更管理員密碼，請使用 `asadmin set` 指令。使用 `set` 指令實現此目的的範例如下：

```
asadmin set
server.jms-service.jms-host.default_JMS_host.admin-password=new_pwd
```

還可以使用管理主控台變更管理員密碼。若要使用管理主控台變更管理員密碼，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 展開 [安全性] 節點。
4. 展開 [範圍] 節點。
5. 選取 [`admin-realm`] 節點。
6. 在 [編輯範圍] 頁面中，按一下 [管理使用者] 按鈕。
7. 選取名爲 `admin` 的使用者。
8. 輸入新密碼並確認密碼。
9. 按一下 [儲存] 以儲存新密碼，或者按一下 [關閉] 以關閉頁面而不儲存新密碼。

指定安全性職責

將爲以下角色指定安全性職責：

- 應用程式開發者
- 應用程式部署者
- 系統管理員

應用程式開發者

應用程式開發者負責：

- 爲應用程式元件指定角色和基於角色的存取限制。
- 定義應用程式的認證方法並指定受保護的應用程式部分。

應用程式開發者可以使用 deploytool 等工具編輯應用程式部署描述元。「The J2EE 1.4 Tutorial」中的「Security」一章詳細論述了這些安全性作業，您可以從以下 URL 中檢視該指導：

http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html

應用程式部署者

應用程式部署者負責：

- 將使用者或群組 (或兩者) 對映至安全性角色。
- 存取元件方法所需的權限以滿足特定部署方案的要求。

應用程式部署者可以使用 deploytool 等工具編輯應用程式部署描述元。「The J2EE 1.4 Tutorial」中的「Security」一章詳細論述了這些安全性作業，您可以從以下 URL 中檢視該指導：

http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html

系統管理員

系統管理員負責：

- 配置安全性範圍。
- 管理使用者帳戶和群組。
- 管理稽核記錄。
- 管理伺服器證書，並配置伺服器對安全套接字層 (SSL) 的使用。
- 處理系統範圍內的其他各類安全性功能，例如連接器連線池的安全性對映、其他 JACC 提供者等等。

系統管理員使用管理主控台管理伺服器安全性設定，使用 certutil 管理證書。本文件主要針對系統管理員。

關於認證與授權

認證與授權是應用程式伺服器安全性的核心概念。以下主題論述了與認證和授權相關的內容：

- 認證實體
- 對使用者進行授權
- 指定 JACC 提供者
- 稽核認證與授權決策
- 配置訊息安全性

認證實體

認證是實體 (使用者、應用程式或元件) 用於確定其他實體是否是其宣告的實體的方法。實體使用**安全性憑證**對其自身進行認證。憑證可以是使用者名稱和密碼、數位證書或其他憑證。

通常，認證表示使用者使用使用者名稱和密碼登入某個應用程式；也可以指 EJB 從伺服器請求資源時，提供安全性憑證。通常，伺服器或應用程式要求使用者端進行認證；另外，使用者端也可以要求伺服器對其自身進行認證。如果認證是雙向的，則稱爲**相互認證**。

當實體嘗試存取受保護的資源時，Application Server 將使用爲該資源配置的認證機制來確定是否授予存取權。例如，使用者可以在 Web 瀏覽器中輸入使用者名稱和密碼，如果應用程式順利完成了對這些憑證的驗證，則表示該使用者已通過認證。在此階段作業的剩餘時間內，該使用者將始終與這個經過認證的安全性身份相關聯。

Application Server 支援四種類型的認證，如表 14-1 所示。應用程式在其部署描述元中指定所使用的認證類型。如需有關使用 deploytool 配置應用程式認證方法的更多資訊，請參閱位於以下 URL 的「The J2EE 1.4 Tutorial」：

http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html

表 14-1 Application Server 認證方法

認證方法	通訊協定	描述	使用者憑證加密
基本	HTTP (SSL 選取性)	使用伺服器的內建快顯式登入對話方塊。	無，除非使用 SSL。

表 14-1 Application Server 認證方法 (續)

基於表單	HTTP (SSL 選取性)	應用程式提供它自己的自訂登入頁面和錯誤頁面。	無，除非使用 SSL。
使用者端證書	HTTPS (基於 SSL 的 HTTP)	伺服器使用公開金鑰證書認證使用者端。	SSL

驗證單次登入

單次登入可以讓一個虛擬伺服器實例中的多個應用程式共用使用者認證狀態。使用單次登入，登入到一個應用程式的使用者也會隱式登入到需要相同認證資訊的其他應用程式。

單次登入基於群組。其部署描述元定義了相同的**群組**並使用相同的認證方法 (基本、表單、摘要、證書) 的所有 Web 應用程式均共用單次登入。

對於爲 Application Server 定義的虛擬伺服器，依預設已啓用單次登入。如需有關停用單次登入的資訊，請參閱 「配置單次登入 (SSO)」。

對使用者進行授權

使用者通過認證後，**授權**層級將決定該使用者可以執行哪些作業。使用者的授權基於其**角色**。例如，人力資源應用程式可以授權管理者檢視所有員工的個人資訊，但每個員工僅能檢視自己的個人資訊。如需有關角色的更多資訊，請參閱 「瞭解使用者、群組、角色和範圍」。

指定 JACC 提供者

JACC (Java 容器授權合約) 是 J2EE 1.4 規格的一部分，它定義可插接式授權提供者介面。這可以讓管理員設置協力廠商外掛程式模組以執行授權。

依預設，Application Server 提供符合 JACC 規格的基於檔案的簡單授權引擎。還可以指定其他協力廠商 JACC 提供者。

JACC 提供者使用 Java 認證與授權服務 (JAAS) API。JAAS 可以讓服務對使用者進行認證並強制對使用者進行存取控制。JAAS 實作了 Java 技術版本的標準可插接式認證模組 (PAM) 框架。

稽核認證與授權決策

Application Server 可以透過**稽核模組**提供對所有認證與授權決策的稽核追蹤。Application Server 提供預設的稽核模組，還提供了自訂稽核模組的功能。如需有關開發自訂稽核模組的資訊，請參閱 Application Server 「Developer's Guide」。如需有關 「Developer's Guide」的連結，請參閱 「詳細資訊」。

配置訊息安全性

訊息安全性可以讓伺服器在訊息層執行 Web 服務呼叫和回應的端對端認證。Application Server 使用 SOAP 層上的訊息安全性提供者實作訊息安全性。訊息安全性提供者提供以下資訊，例如請求訊息和回應訊息所需的認證類型。受支援的認證類型包括：

- 寄件者認證，包括使用者名稱密碼認證。
- 特性認證，包括 XML 數位簽名。

此發行版本包括兩個訊息安全性提供者。可以爲 SOAP 層的認證配置訊息安全性提供者。可以配置的提供者包括 `ClientProvider` 和 `ServerProvider`。

以 (可插接式) 認證模組的形式將訊息層安全性支援整合到 Application Server 及其使用者端容器中。依預設，Application Server 中的訊息層安全性處於停用狀態。

可以爲整個 Application Server 或者爲特定應用程式或方法配置訊息層安全性。「配置訊息安全性」論述了在 Application Server 層級配置訊息安全性。「Developer's Guide」一章中論述了如何配置應用程式層級的訊息安全性。

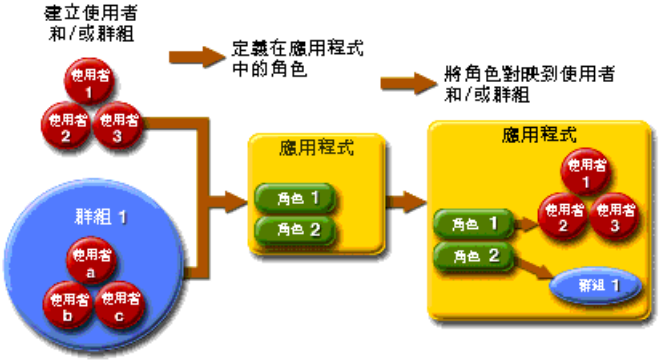
瞭解使用者、群組、角色和範圍

Application Server 對以下實體強制執行其認證與授權策略：

- 使用者：*Application Server* **中定義的**個人身份。通常，使用者是指個人、一個軟體元件 (例如企業 Bean)，甚至是一種服務。經過認證的使用者有時被稱爲**主體**。使用者有時稱爲**個人**。
- 群組：*Application Server* **中定義的**一組使用者，按照一般特性進行分類。
- 角色：**由應用程式定義的**命名授權層級。可以將角色比作開鎖的鑰匙。許多人都可以有此鑰匙的複製鑰匙。鎖不關心誰要造訪，而只關心使用的鑰匙是否正確。
- 範圍：包含使用者和群組資訊及其關聯的安全性憑證的儲存庫。範圍也被稱爲**安全策略網域**。

備註：儘管使用者和群組是爲整個 Application Server 指定的，但是每個應用程式都需要定義自己的角色。封裝和部署應用程式時，應用程式會指定使用者/群組和角色之間的對映，如下圖所示。

角色對映



使用者

使用者是已在 Application Server 中定義的個人 (或應用程式) 身份。使用者可以與群組關聯。Application Server 認證服務可以管理多個範圍中的使用者。

群組

J2EE **群組** (或簡稱群組) 是按一般特性 (例如職務或客戶設定檔) 進行分類的使用者類別。例如，假定電子商業應用程式的使用者屬於 `customer` 群組，但是大客戶可以屬於 `preferred` 群組。將使用者進行群組分類可以簡化有大量使用者時的存取控制。

角色

角色定義使用者可以存取哪些應用程式和每個應用程式的哪些部分以及使用者可以執行的作業。也就是說，角色決定了使用者的授權層級。

例如，假定在人事應用程式中，所有員工均可以存取電話號碼和電子郵件位址，但只有管理者才能存取薪水資訊。該應用程式至少可以定義兩個角色：`employee` 和 `manager`；僅允許 `manager` 角色中的使用者檢視薪水資訊。

角色與使用者群組的不同之處在於，角色在應用程式中定義功能，而使用者群組是以某一方式相關的一組使用者。例如，假定在人事應用程式中有 `full-time`、`part-time` 和 `on-leave` 幾個群組，但所有這些群組中的使用者仍是 `employee` 角色。

角色是在應用程式部署描述元中定義的。相反，群組是針對整個伺服器和範圍而定義的。應用程式開發者或部署者在每個應用程式的部署描述元中將角色對映至一個或多個群組。

範圍

範圍 (也稱爲**安全策略網域或安全網域**) 是伺服器定義和強制執行一般安全策略的範圍。在實際應用中，範圍是伺服器儲存使用者和群組資訊的儲存庫。

Application Server 預先配置了三個範圍：`file` (初始預設範圍)、`certificate` 和 `admin-realm`。您還可以設置 `ldap`、`solaris` 或自訂範圍。應用程式可以在其部署描述元中指定要使用的範圍。如果應用程式不指定範圍，Application Server 將使用其預設範圍。

在 `file` 範圍中，伺服器將使用者憑證儲存在本地名爲 `keyfile` 的檔案中。您可以使用管理主控台來管理 `file` 範圍中的使用者。如需更多資訊，請參閱「管理 file 範圍使用者」。

在 `certificate` 範圍中，伺服器將使用者憑證儲存在證書資料庫中。使用 `certificate` 範圍時，伺服器結合使用證書和 HTTPS 通訊協定認證 Web 使用者端。如需有關證書的更多資訊，請參閱「證書和 SSL 簡介」。

`admin-realm` 也是一個 `FileRealm`，它將管理員使用者憑證儲存在本地名爲 `admin-keyfile` 的檔案中。您可以使用管理主控台管理此範圍中的使用者，其方法與您管理 `file` 範圍中的使用者的方法相同。如需更多資訊，請參閱「管理 file 範圍使用者」。

在 `ldap` 範圍中，伺服器將從簡易目錄存取協定 (LDAP) 伺服器 (例如 Sun Java System Directory Server) 中取得使用者憑證。LDAP 是一種可以讓任何人在網路 (無論是公共網際網路還是企業內部網路) 中尋找組織、個人和其他資源 (例如檔案和裝置) 的協定。如需有關管理 `ldap` 範圍中的使用者和群組的資訊，請參閱您的 LDAP 伺服器文件。

在 `solaris` 範圍中，伺服器將從 Solaris 作業系統中取得使用者憑證。Solaris 9 OS 和更高版本支援此範圍。如需有關管理 `solaris` 範圍中的使用者和群組的資訊，請參閱您的 Solaris 文件。

自訂範圍是使用者憑證的任何其他儲存庫，例如關聯式資料庫或協力廠商元件。如需更多資訊，請參閱「建立自訂範圍」或「Developer's Guide」一章。

證書和 SSL 簡介

本節論述以下主題：

- 關於數位證書
- 關於安全套接字層

關於數位證書

數位證書 (或簡稱證書) 是在網際網路上唯一識別人員和資源的電子檔案。證書還可以使兩個實體之間能夠進行安全、機密的通訊。

證書有很多種類型，例如個人證書 (由個人使用) 和伺服器證書 (用於透過安全套接字層 [SSL] 技術在伺服器和使用者端之間建立安全階段作業)。如需有關 SSL 的更多資訊，請參閱 「關於安全套接字層」。

證書基於**公開金鑰密碼學**，公開金鑰密碼學使用數位**金鑰**組 (很長的數位) 對資訊進行**加密**或編碼，從而使資訊只能被預定的接受者讀取。然後，接受者對資訊進行**解密** (解碼)，即可讀取該資訊。

一個金鑰對包含一個公開金鑰和一個私密金鑰。所有者對公開金鑰進行發放並使任何人都可以使用該公開金鑰。但是所有者永遠不會發放私密金鑰；私密金鑰始終是保密的。由於金鑰與數學相關，因此使用了金鑰對中的一個金鑰進行加密的資料只能透過金鑰對中的另一個金鑰進行解密。

證書就好像一本護照：它可以識別持有者並提供其他重要資訊。證書由稱爲**憑證授權單位** (CA) 的可信任的協力廠商發行。CA 類似於護照申領辦公室：它將驗證證書持有者的身份並對證書進行簽名，以使他人無法僞造或竄改證書。CA 對證書進行簽名之後，持有者可以提供該證書作爲身份證明並建立加密的機密通訊。

最重要的是，證書會將所有者的公開金鑰連結至所有者身份。與護照將照片連結至其持有者的個人資訊類似，證書將公開金鑰連結至有關其所有者的資訊。

除了公開金鑰以外，證書通常還包括以下資訊：

- 持有者的姓名和其他標識，例如使用證書的 Web 伺服器的 URL 或個人的電子郵件位址。
- 發行證書的 CA 的名稱。
- 過期日期。

數位證書受 x.509 格式的技術規格約束。若要驗證 `certificate` 範圍中某個使用者的身份，認證服務將使用 X.509 證書的一般名稱欄位作爲主體名稱對 X.509 證書進行驗證。

關於證書鏈

Web 瀏覽器已預先配置了一組瀏覽器自動信任的**根** CA 證書。來自其他憑證授權單位的所有證書都必須附帶**證書鏈**，以驗證這些證書是否有效。證書鏈是由一系列 CA 發行的證書序列，最終以根 CA 證書結束。

證書最初產生時是**自簽名**證書。自簽名證書是其發行者 (簽名者) 與主旨 (其公開金鑰由該證書進行認證的實體) 相同的證書。如果所有者向 CA 傳送證書簽名請求 (CSR)，然後輸入回應，自簽名證書將被證書鏈取代。鏈的底部是由 CA 發行的、用於認證主旨的公開金鑰證書 (回覆)。鏈中的下一個證書是認證 CA 公開金鑰的證書。通常，這是一個自簽名證書 (即，來自 CA、用於認證其自身公開金鑰的證書)，並且是鏈中的最後一個證書。

在其他情況下，CA 可能會傳回一個證書鏈。在此情況下，鏈的底部證書是相同的 (由 CA 簽名的證書，用於認證金鑰項目的公開金鑰)，但是鏈中的第二個證書是由其他 CA 簽名的證書，用於認證您向其傳送了 CSR 的 CA 的公開金鑰。然後，鏈中的下一個證書是用於認證第二個 CA 金鑰的證書，依此類推，直至到達自簽名的**根**證書。因此，鏈中的每個證書 (第一個證書之後的證書) 都需要認證鏈中前一個證書的簽名者的公開金鑰。

關於安全套接字層

安全套接字層 (SSL) 是用於確定網際網路通訊和作業事件保護的最常見標準。Web 應用程式使用 HTTPS (基於 SSL 的 HTTP)，HTTPS 使用數位證書來確保在伺服器和使用者端之間進行安全、機密的通訊。在 SSL 連線中，使用者端和伺服器在傳送資料之前都要對資料進行加密，然後由接受者對其進行解密。

Web 瀏覽器 (使用者端) 需要與某個安全站台建立連線時，則會發生 *SSL* **交握**：

- 瀏覽器將透過網路傳送請求安全階段作業的訊息 (通常請求以 `https` 開頭而非 `http` 開頭的 URL)。
- 伺服器透過傳送其證書 (包括公開金鑰) 進行回應。
- 瀏覽器將驗證伺服器證書是否有效，並且該證書是否是由其證書位於瀏覽器的資料庫中的 CA (並且是可信任的 CA) 所簽名的。它還驗證 CA 證書是否已過期。
- 如果證書有效，瀏覽器將產生一個一次性的、唯一的**階段作業金鑰**，並使用伺服器的公開金鑰對該階段作業進行加密。然後，瀏覽器將把加密的階段作業金鑰傳送給伺服器，這樣伺服器和瀏覽器都擁有該階段作業金鑰副本。
- 伺服器可以使用其私密金鑰對訊息進行解密，然後恢復該階段作業金鑰。

交握之後，即表示使用者端已驗證了網站的身份，並且只有該使用者端和 Web 伺服器擁有該階段作業金鑰副本。從現在開始，使用者端和伺服器便可以使用該階段作業金鑰對彼此間的所有通訊進行加密。這樣就確保了使用者端和伺服器之間的通訊的安全。

SSL 標準的最新版本稱爲 TLS (傳輸層安全性)。Application Server 支援安全套接字層 (SSL) 3.0 和傳輸層安全性 (TLS) 1.0 加密協定。

若要使用 SSL，Application Server 必須擁有接受安全連線的每個外部介面或 IP 位址的證書。只有安裝了數位證書之後，大多數 Web 伺服器的 HTTPS 服務才能夠執行。請使用 「產生伺服器證書」中描述的程序來設置您的 Web 伺服器可以用於 SSL 的數位證書。

關於加密算法

加密算法是用於加密或解密的加密演算法。SSL 和 TLS 協定支援用於伺服器和使用者端彼此進行認證、傳輸證書和建立階段作業金鑰的各種加密算法。

某些加密算法比其他加密算法更強大且更安全。使用者端和伺服器可以支援不同的密碼組。從 SSL3 和 TLS 協定中選取加密算法。在安全連線期間，使用者端和伺服器同意在通訊中使用它們均已啓用的最強大的加密算法，因此通常需要啓用所有加密算法。

使用基於名稱的虛擬主機

對安全應用程式使用基於名稱的虛擬主機可能會帶來問題。這是 SSL 協定本身的設計限制。必須先進行 SSL 交握 (使用者端瀏覽器在此時接受伺服器證書)，然後才能存取 HTTP 請求。這樣，在認證之前就無法確定包含虛擬主機名稱的請求資訊，因此也不能將多個證書指定給單一 IP 位址。

如果單一 IP 位址上的所有虛擬主機均需要對照同一證書進行認證，則增加多個虛擬主機將不會影響伺服器上正常的 SSL 作業。但是請注意，大多數瀏覽器會將伺服器的網域名稱與證書中列示的網域名稱 (如果有，且主要適用於官方 CA 簽名證書) 進行對照。如果網域名稱不匹配，這些瀏覽器將顯示警告。通常在生產環境中，只將基於位址的虛擬主機與 SSL 配合使用。

關於防火牆

防火牆控制兩個或多個網路之間的資料流，並管理網路之間的連結。防火牆可能包含硬體和軟體元素。本節描述了一些一般防火牆架構及其配置。此處的資訊主要是針對 Application Server 的。如需有關特定防火牆技術的詳細資訊，請參閱防火牆供應商提供的文件。

通常，需要對防火牆進行配置，以便使用者端存取所需的 TCP/IP 連接埠。例如，如果 HTTP 偵聽程式正在連接埠 8080 上作業，則將防火牆配置爲僅允許處理連接埠 8080 上的 HTTP 請求。同樣地，如果爲連接埠 8181 設定了 HTTPS 請求，則必須將防火牆配置爲允許處理連接埠 8181 上的 HTTPS 請求。

如果需要從網際網路對 EJB 模組進行直接的 RMI-IIOP (全稱爲 Remote Method Invocations over Internet Inter-ORB Protocol [通過網際網路 ORB 交換協定的遠端方法呼叫]) 存取，則還需要開啓 RMI-IIOP 偵聽程式連接埠，但強烈建議您不要這樣做，因爲這樣可能會破壞安全性。

在雙防火牆架構中，您必須將外部防火牆配置爲允許處理 HTTP 和 HTTPS 作業事件。您必須將內部防火牆配置爲允許 HTTP 伺服器外掛程式與防火牆後面的 Application Server 進行通訊。

使用管理主控台管理安全性

管理主控台提供了對安全性的以下方面進行管理的方法：

- 伺服器安全性設定
- 範圍和 file 範圍使用者
- JACC 提供者
- 稽核模組
- 訊息安全性
- HTTP 和 IIOP 偵聽程式安全性
- 管理服務安全性
- 安全性對映

伺服器安全性設定

在 [安全性設定] 頁面中，設定整個伺服器的特性，包括指定預設範圍、匿名角色和預設的主體使用者名稱和密碼。如需更多資訊，請參閱 「配置安全性設定」。

範圍和 file 範圍使用者

「瞭解使用者、群組、角色和範圍」中介紹了範圍的概念。使用管理主控台執行以下作業：

- 建立新範圍

- 刪除現有範圍
- 修改現有範圍的配置
- 增加、修改和刪除 `file` 範圍中的使用者
- 設定預設範圍

請參閱「用於範圍的管理主控台作業」，以取得有關這些作業的詳細資訊。

JACC 提供者

「指定 JACC 提供者」中介紹了 JACC 提供者。使用管理主控台執行以下作業：

增加新的 JACC 提供者

- 刪除或修改現有 JACC 提供者

請參閱「有關 JACC 提供者的管理主控台作業」，以取得有關這些作業的詳細資訊。

稽核模組

「稽核認證與授權決策」中介紹了稽核模組。稽核是記錄重要事件 (例如錯誤或安全性漏洞) 以進行後續檢查的方法。所有認證事件都被記錄到 Application Server 記錄中。完整的存取記錄提供了 Application Server 存取事件的順序線索。

使用管理主控台執行以下作業：

- 增加新的稽核模組
- 刪除或修改現有稽核模組

請參閱「有關稽核模組的管理主控台作業」，以取得有關這些作業的詳細資訊。

訊息安全性

「配置訊息安全性」中介紹了訊息安全性的概念。使用管理主控台執行以下作業：

- 啓用訊息安全性
- 配置訊息安全性提供者
- 刪除或配置現有訊息安全性配置或提供者

請參閱「配置訊息安全性」，以取得有關這些作業的詳細資訊。

HTTP 和 IIOP 偵聽程式安全性

HTTP 服務中的每個虛擬伺服器都透過一個或多個 *HTTP* **偵聽程式**提供網路連線。如需有關 HTTP 服務和 HTTP 偵聽程式的一般資訊，請參閱「什麼是 HTTP 服務？」。

Application Server 支援 CORBA (共用物件請求代理程式架構) 物件，這類物件使用網際網路 Orb 交換協定 (IIOP) 在網路上進行通訊。*IIOP* **偵聽程式**接受來自 EJB 的遠端使用者端和其他基於 CORBA 的使用者端的進來的連線。如需有關 IIOP 偵聽程式的一般資訊，請參閱「IIOP 偵聽程式」。

使用管理主控台執行以下作業：

- 建立新的 HTTP 或 IIOP 偵聽程式，並指定該偵聽程式所使用的安全性。
- 修改現有 HTTP 或 IIOP 偵聽程式的安全性設定。

如需有關這些作業的詳細資訊，請參閱「有關偵聽程式和 JMX 連接器的管理主控台作業」。

管理服務安全性

管理服務決定伺服器實例是一般實例、網域管理伺服器 (DAS)，還是兼具兩者。使用管理服務可以配置 JSR-160 相容遠端 JMX 連接器，該連接器處理網域管理伺服器與遠端伺服器實例的節點代理程式 (管理主機電腦上的伺服器實例) 之間的通訊。

使用管理主控台執行以下作業：

- 管理管理服務
- 編輯 JMX 連接器
- 修改 JMX 連接器的安全性設定

請參閱「配置管理服務的 JMX 連接器的安全性」，以取得有關這些作業的詳細資訊。

安全性對映

「關於安全性對映」中介紹了用於連接器連線池的安全性對映的概念。使用管理主控台執行以下作業：

- 將安全性對映增加至現有連接器連線池中
- 刪除或配置現有安全性對映

請參閱「有關連接器連線池的管理主控台作業」，以取得有關這些作業的詳細資訊。

用於安全性的管理主控台作業

- 配置安全性設定
- 控制對管理工具的存取
- 配置相互認證
- 配置單次登入 (SSO)

配置安全性設定

管理主控台中的 [安全性] 頁面可以讓您設定各種系統範圍內的安全性設定。

若要編輯這些設定，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 選取 [安全性] 節點。

   將顯示 [安全性] 頁面。
4. 依需要修改值。表 14-2 說明了一般安全性選項。

表 14-2 一般安全性設定

設定	描述
稽核記錄	選取以啟用稽核記錄。如果啟用稽核記錄，伺服器將載入並執行在 [稽核模組] 設定中指定的所有稽核模組。如果停用稽核記錄，伺服器將不存取稽核模組。依預設，稽核記錄處於停用狀態。
預設範圍	伺服器用於進行認證的使用中 (預設) 範圍。除非應用程式在其部署描述元中指定了其他範圍，否則應用程式將使用此範圍。清單中將顯示所有已配置的範圍。初始預設範圍為 `file` 範圍。
匿名角色	預設或匿名角色的名稱。匿名角色將被指定給所有使用者。應用程式可以在其部署描述元中使用此角色以向任何人授予許可權。
預設主體	指定預設使用者名稱。如果未提供任何主體，伺服器將使用此預設主體。如果在此欄位中輸入了值，則請在 [預設主體密碼] 欄位中輸入相應的值。 一般伺服器作業不需要此屬性。

表 14-2 一般安全性設定 (續)

設定	描述
預設主體密碼	在 [預設主體] 欄位中指定的預設主體的密碼。 一般伺服器作業不需要此屬性。
JACC	已配置的 JACC 提供者的類別名稱。請參閱「建立 JACC 提供者」，以取得有關增加 JACC 提供者的資訊。
稽核模組	稽核模組提供者類別清單 (以逗號分隔)。此處所列示的模組必須是已配置模組。如果啟用 [稽核記錄]，則此設定必須列示稽核模組。依預設，伺服器使用名為 `default` 的稽核模組。如需有關建立新稽核模組的資訊，請參閱「建立稽核模組」。

5. 在 [附加特性] 區段輸入要傳送至 Java 虛擬機器 (JVM) 的附加特性。

   有效特性取決於 [預設範圍] 欄位中所選取範圍的類型。在以下小節中會論述有效特性：

   - 編輯 file 和 admin-realm 範圍
   - 編輯 certificate 範圍
   - 建立 solaris 範圍
   - 建立 ldap 範圍
   - 建立自訂範圍

6. 選取 [儲存] 以儲存變更，或者選取 [載入預設值] 以復原預設值。

控制對管理工具的存取

只有 `asadmin` 群組中的使用者才能存取管理主控台和 `asadmin` 指令行公用程式。

若要授予使用者對這些管理工具的存取權，請將這些工具增加至 `admin-realm` 的 `asadmin` 群組中。若要完成此操作，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 展開 [安全性] 節點。

4. 展開 [範圍] 節點。
5. 選取 [admin-realm] 節點。
6. 在 [編輯範圍] 頁面中，按一下 [管理使用者] 按鈕。

   初次完成安裝之後，安裝期間輸入的管理員使用者名稱和密碼將列在名爲 admin-keyfile 的檔案中。依預設，此使用者屬於 asadmin 群組，該群組可以授予修改 Application Server 的權限。請僅在您要爲使用者授予 Application Server 的管理員權限時，才將使用者指定至該群組。

   如果您將使用者增加至 admin-realm 範圍，但將使用者指定給 asadmin 以外的群則則使用者資訊仍將被寫入至名爲 admin-keyfile 的檔案，但使用者不具有對管理工具或 file 範圍中的應用程式的存取權。
7. 按一下 [新建] 以將新使用者增加至 admin-realm 範圍。
8. 在 [使用者 ID]、[密碼]、和 [群組清單] 欄位中輸入正確資訊。若要授予使用者修改 Application Server 的權利，請將 asadmin 群組包含在 [群組清單] 中。
9. 按一下 [確定] 將此使用者增加至 admin-realm 範圍中，或按一下 [取消] 以退出而不儲存變更。

用於範圍的管理主控台作業

- 建立範圍
  - 建立 ldap 範圍
  - 建立 solaris 範圍
  - 建立自訂範圍
- 編輯範圍
  - 編輯 file 和 admin-realm 範圍
  - 使用網路安全性服務 (NSS) 管理使用者 (Enterprise Edition)
  - 管理 file 範圍使用者
  - 編輯 certificate 範圍
- 刪除範圍
- 設定預設範圍

建立範圍

Application Server 預先配置了三個範圍：file、certificate 和 admin-realm。您還可以建立 ldap、solaris 和自訂範圍。通常，一個伺服器上會有每種類型的一個範圍，但在 Application Server 上有兩個 file 範圍：file 和 admin-realm。這兩個範圍類型相同但用於兩種不同的目的。您的系統也可以爲每個虛擬伺服器配備不同的證書資料庫。

若要建立安全性範圍，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。
3. 展開 [安全性] 節點。
4. 選取 [範圍] 節點。
5. 在 [範圍] 頁面中，按一下 [新建]。

   將顯示 [建立範圍] 頁面。
6. 在 [名稱] 欄位中爲範圍輸入名稱。
7. 指定要建立的範圍的類別名稱。表 14-3 顯示了有效選項：

表 14-3 範圍類別名稱的有效值

範圍名稱	類別名稱
file	com.sun.enterprise.security.auth.realm.file.FileRealm
證書	com.sun.enterprise.security.auth.realm.certificate.CertificateRealm
ldap	com.sun.enterprise.security.auth.realm.ldap.LDAPRealm
solaris	com.sun.enterprise.security.auth.realm.solaris.SolarisRealm
自訂	登入範圍類別的名稱

8. 增加範圍必需特性和任何所需的選擇性特性。

   若要增加特性，請執行以下步驟：
   a. 按一下 [增加特性]。

b. 在 [名稱] 欄位中，輸入特性的名稱。

- 如需有關 `file` 範圍特性的描述，請參閱「編輯 file 和 admin-realm 範圍」。
- 如需有關 `certificate` 範圍特性的描述，請參閱「編輯 certificate 範圍」。
- 如需有關 `ldap` 範圍特性的描述，請參閱「建立 ldap 範圍」。
- 如需有關 `solaris` 範圍特性的描述，請參閱「建立 solaris 範圍」。
- 如需有關自訂範圍特性的描述，請參閱「建立自訂範圍」。

c. 在 [值] 欄位中輸入特性的值。

9. 按一下 [確定]。

等效 `asadmin` 指令爲：`create-auth-realm`

建立 ldap 範圍

`ldap` 範圍使用 LDAP 伺服器資訊執行認證。使用者資訊包括使用者名稱、密碼和使用者所屬的群組。若要使用 LDAP 範圍，必須已在 LDAP 目錄中定義了使用者和群組。

若要建立 LDAP 範圍，請按照「建立範圍」中的步驟增加新範圍，然後增加表 14-4 中所示的特性。

表 14-4 `ldap` 範圍的必需特性

特性名稱	描述	值
directory	目錄伺服器的 LDAP URL。	LDAP URL 的格式為： `ldap://`*hostname*`:`*port* 例如， `ldap://myldap.foo.com:389`。
base-dn	使用者資料位置的基底辨別名稱 (DN)，由於將執行樹範圍的搜尋，因此該使用者資料的位置可以是高於使用者資料的任何層級。搜尋樹越小，效能越好。	搜尋的網域，例如： `dc=siliconvalley, dc=BayArea, dc=sun, dc=com`。
jaas-context	要用於此範圍的登入模組類型。	必須為 `ldapRealm`。

表 14-5 顯示了 ldap 範圍的選擇性特性：

表 14-5 ldap 範圍的選擇性特性

特性名稱	描述	預設
search-filter	用於尋找使用者的搜尋篩選器。	uid=%s (%s 擴展為主旨名稱)。
group-base-dn	群組資料位置的基底 DN。	與 base-dn 相同，但如果需要也可以進行調整。
group-search-filter	用於尋找使用者的群組成員關係的搜尋篩選器。	uniquemember=%d (%d 擴展為使用者元素 DN)。
group-target	包含群組名稱項目的 LDAP 屬性名稱。	CN
search-bind-dn	用於向目錄認證以執行 search-filter 查找的選擇性 DN。只有不允許進行匿名搜尋的目錄才需要。	
search-bind-password	search-bind-dn 中指定的 DN 的 LDAP 密碼。	

範例

例如，假定在 LDAP 目錄中定義了一個 LDAP 使用者 Joe Java，如下所示：

```
uid=jjava,ou=People,dc=acme,dc=com
uid=jjava
givenName=joe
objectClass=top
objectClass=person
objectClass=organizationalPerson
objectClass=inetorgperson
sn=java
cn=Joe Java
```

使用範例程式碼，在建立或編輯 ldap 範圍時，您可以輸入表 14-6 中所示的值。

表 14-6 ldap 範圍值範例

特性名稱	特性值
directory	伺服器的 LDAP URL，例如：ldap://ldap.acme.com:389
base-dn	ou=People,dc=acme,dc=com. 可以向更高層次進行搜尋 (例如 dc=acme、dc=com)，但搜尋將遍歷樹的更大部分，從而導致效能降低。
jaas-context	ldapRealm

建立 solaris 範圍

solaris 範圍從基礎 Solaris 使用者資料庫 (由系統配置確定) 取得使用者和群組資訊。solaris 範圍呼叫基礎 PAM 基礎架構以用於認證。如果已配置的 PAM 模組需要超級使用者權限，則網域必須以超級使用者身份執行才能使用該範圍。如需詳細資訊，請參閱安全性服務的 Solaris 文件。

solaris 範圍有一個必需特性 jaas-context，該特性指定要使用的登入模組的類型。特性值必須爲 solarisRealm。

備註：只有 Solaris 9 或更高版本才支援 solaris 範圍。

建立自訂範圍

除了四個內建範圍之外，您還可以建立以其他方式儲存使用者資料 (例如儲存在關聯式資料庫中) 的自訂範圍。自訂範圍的開發不在本文件的論述範圍之內。如需更多資訊，請參閱 Application Server 「 Developer's Guide 」一章。

作爲管理員，您需要瞭解的主要事項是：自訂範圍是由源自 Java 認證與授權服務 (JAAS) 套裝軟體的類別 (稱爲 LoginModule) 實作的。

若要配置 Application Server 以使用自訂範圍，請執行以下步驟：

1. 按照「建立範圍」中概述的程序，輸入自訂範圍的名稱和 LoginModule 類別的名稱。自訂範圍可以使用任何具有唯一性的名稱，例如 myCustomRealm。
2. 增加表 14-7 所示的特性：

表 14-7 自訂範圍的有效特性

特性名稱	特性值
jaas-context	LoginModule 類別名稱，例如 simpleCustomRealm
auth-type	對範圍的描述，例如「簡單範例自訂範圍」。

3. 按一下 [確定]。
4. 編輯網域的登入配置檔案 *install_dir*/domains/*domain_name*/config/login.conf，並在檔案末尾增加 JAAS LoginModule 的完全合格類別名稱，如下所示：

```
realmName {
    fully-qualified-LoginModule-classname required;
};
```

例如，

```
myCustomRealm {
    com.foo.bar.security.customrealm.simpleCustomLoginModule required;
};
```

5. 將 LoginModule 類別和所有相依類別複製到 *install_dir*/domains/*domain_name*/lib/classes 目錄中。
6. 如果主控台中顯示「需要重新啓動」，請重新啓動伺服器。
7. 確定已正確載入範圍。

   檢查 *install_dir*/domains/*domain_name*/logs/server.log 以確定伺服器已載入該範圍。伺服器應呼叫範圍的 init() 方法。

編輯範圍

若要編輯範圍，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。
3. 展開 [安全性] 節點。
4. 展開 [範圍] 節點。
5. 選取現有範圍的名稱。

   將顯示 [編輯範圍] 頁面。
6. 依需要編輯現有特性及其值。

   如需有關 file 範圍特性的資訊，請參閱「編輯 file 和 admin-realm 範圍」。若要管理 file 範圍中的使用者，請按一下 [管理使用者] 按鈕；請參閱「管理 file 範圍使用者」，以取得更多資訊。

   如需有關 certificate 範圍特性的資訊，請參閱「編輯 certificate 範圍」。
7. 若要增加附加特性，請按一下 [增加特性] 按鈕。該頁面將顯示一個新行。輸入有效的特性名稱和特性值。如需有關對可以配置的選擇性特性的描述，請參閱下列各表：
   - 表 14-4，ldap 範圍的必需特性
   - 表 14-5，ldap 範圍的選擇性特性

- 表 14-7，自訂範圍的有效特性
- 表 14-8，file 範圍的必需特性
- 表 14-9，certificate 範圍的選擇性特性

8. 按一下 [儲存] 以儲存變更。

編輯 file 和 admin-realm 範圍

伺服器在 file 範圍的名爲 keyfile 的檔案和 admin-realm 範圍的名爲 admin-keyfile 的檔案中維護所有使用者、群組和密碼資訊。對於這兩種範圍，file 特性均指定了 keyfile 的位置。表 14-8 顯示了 file 範圍的必需特性。

表 14-8 file 範圍的必需特性

特性名稱	描述	預設值
file	keyfile 的完整路徑和名稱。	*install_dir*/domains/*domain-name*/config/keyfile
jaas-context	要用於此範圍的登入模組類型。	fileRealm 是唯一的有效值

keyfile 最初爲空，因此在使用 file 範圍之前，必須先增加使用者。如需說明，請參閱「管理 file 範圍使用者」。

admin-keyfile 最初包含管理員使用者名稱、加密格式的管理員密碼和該使用者所屬的群組 (依預設爲 asadmin)。如需有關將使用者增加至 admin-realm 的更多資訊，請參閱「控制對管理工具的存取」。

備註：admin-realm 的 asadmin 群組中的使用者已被授權，可以使用管理主控台和 asadmin 工具。只能將具有伺服器管理權限的使用者增加至該群組中。

使用網路安全性服務 (NSS) 管理使用者

僅在 **Enterprise Edition** 中，您可以使用管理主控台來管理使用者 (如「管理 file 範圍使用者」中所述)，或者使用 NSS 工具來管理使用者。網路安全性服務 (NSS) 是一組程式庫，用於支援啓用安全性的用戶端和伺服器應用程式的跨平台開發。使用 NSS 建置的應用程式可以支援 SSL v2 和 v3、TLS、PKCS #5、PKCS #7、PKCS #11、PKCS #12、S/MIME、X.509 v3 證書和其他安全性標準。如需詳細資訊，請連結到以下 URL：

- 「Network Security Services (NSS)」(位於 http://www.mozilla.org/projects/security/pki/nss/)

- 「NSS Security Tools」(位於 http://www.mozilla.org/projects/security/pki/nss/tools/)
- 「Overview of NSS」(位於 http://www.mozilla.org/projects/security/pki/nss/overview.html)

管理 file 範圍使用者

使用管理主控台來管理 file 範圍使用者。file 範圍中的使用者和群組列在 keyfile 檔案中，該檔案的位置由 file 特性指定。

備註：還可以使用以下步驟將使用者增加至任何 file 範圍 (包括 admin-realm)。您只需用本小節中參照的 file 範圍取代目標範圍的名稱即可完成作業。

file 範圍中的使用者可以屬於 *J2EE* **群組** (是按照一般屬性分類的使用者類別)。例如，假定電子商業應用程式的使用者屬於 CUSTOMER 群組，但是大客戶可以屬於 PREFERRED 群組。將使用者進行群組分類可以簡化有大量使用者時的存取控制。

初次安裝 Application Server 之後，唯一的使用者是管理員在安裝期間輸入的使用者。依預設，此使用者屬於 admin-realm 範圍中的 asadmin 群組，該群組可以授予修改 Application Server 的權利。指定給該群組的任何使用者都將具有管理員權限，也就是說，這些使用者具有對 asadmin 工具和管理主控台的存取權。

若要管理 file 範圍使用者，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。
3. 展開 [安全性] 節點。
4. 展開 [範圍] 節點。
5. 選取 [file] 節點。
6. 在 [編輯範圍] 頁面中，按一下 [管理使用者] 按鈕。

   將顯示 [file 使用者] 頁面。在此頁面中執行以下作業：

   - 增加使用者
   - 編輯使用者
   - 刪除使用者

增加使用者

在 [file 使用者] 頁面中，執行以下步驟來增加新使用者：

1. 按一下 [新建] 以將新使用者增加至 `file` 範圍。
2. 在 [file 使用者] 頁面中輸入以下資訊：
   - **使用者 ID (必填)** — 使用者名稱。
   - **密碼 (必填)** — 使用者密碼。
   - **確認密碼 (必填)** — 使用者再次輸入密碼以進行驗證。
   - **群組清單 (選擇性)** — 使用者所屬的群組清單 (以逗號分隔)。無需在其他位置定義這些群組。
3. 按一下 [確定] 以將此使用者增加至 `file` 範圍的使用者清單中。按一下 [取消] 以退出而不儲存變更。

等效 `asadmin` 指令爲：`create-file-user`

編輯使用者

在 [file 使用者] 頁面中，執行以下步驟來變更使用者資訊：

1. 在 [使用者 ID] 欄中，按一下要修改的使用者名稱。

   將顯示 [編輯 file 範圍使用者] 頁面。
2. 在 [密碼] 和 [確認密碼] 欄位中輸入新密碼來變更使用者密碼。
3. 在 [群組清單] 欄位中增加或刪除群組來變更使用者所屬的群組。用逗號將群組名稱分隔開。不需要先定義群組。
4. 按一下 [儲存] 以將此使用者儲存在 `file` 範圍的使用者清單中。按一下 [關閉] 以退出而不儲存變更。

刪除使用者

在 [file 使用者] 頁面中，執行以下步驟來刪除使用者：

1. 選取要刪除的使用者名稱左側的核取方塊。
2. 按一下 [刪除]。
3. 按一下 [關閉] 返回至 [編輯範圍] 頁面。

等效 `asadmin` 指令爲：`delete-file-user`

編輯 certificate 範圍

certificate 範圍支援 SSL 認證。該範圍在 Application Server 安全性環境中設定使用者身份，並使用從信任儲存和金鑰存放區檔案 (請參閱「關於證書檔案」) 中以加密方式檢驗的用戶端證書中獲得的使用者資料填入該身份。使用 certutil 將使用者增加至這些檔案。使用 certificate 範圍，J2EE 容器可以基於每個使用者證書中的辨別名稱 (DN) 來執行授權處理。DN 是證書對其公開金鑰進行識別的實體的名稱。此名稱使用 X.500 標準，因此它在網際網路中應該是唯一的。如需有關金鑰存放區和信任儲存的更多資訊，請參閱位於以下位置的「關於 CertUtil 公用程式」(certutil 文件)。

表 14-9 列示了 certificate 範圍的選擇性特性。

表 14-9 certificate 範圍的選擇性特性

特性	描述
assign-groups	以逗號分隔的群組名稱清單。提供有效證書的所有用戶端均被指定給這些群組。例如，employee,manager，它們是使用者群組的名稱所在位置。
jaas-context	要用於此範圍的登入模組類型。對於 certificate 範圍，該值必須為 certificateRealm。

配置相互認證

- 對所有應用程式啓用相互認證
- 在應用程式中啓用相互 SSL 認證

在相互認證中，伺服器端和用戶端認證都會被啓用。若要測試相互認證，必須存在一個包含有效證書的用戶端。如需有關相互認證的資訊，請參閱位於以下位置的「The J2EE 1.4 Tutorial」中的「Security」一章：

http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html

對所有應用程式啟用相互認證

Application Server 使用 certificate 範圍進行 HTTPS 認證。

若要指定對使用此範圍的所有應用程式進行相互認證，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。

b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。

3. 展開 [安全性] 節點。
4. 展開 [範圍] 節點。
5. 選取 [certificate] 範圍。
6. 按一下 [增加特性] 按鈕。
   - 在 [名稱] 欄位中，輸入 clientAuth。
   - 在 [值] 欄位中，輸入 true。
7. 按一下 [儲存]。
8. 如果主控台中顯示「需要重新啓動」，請重新啓動 Application Server。

   重新啓動伺服器之後，需要對使用 certificate 範圍的所有應用程式進行用戶端認證。

在應用程式中啟用相互 *SSL* 認證

若要啓用對特定應用程式的相互認證，請使用 deploytool 將認證方法設定爲 Client-Certificate。如需有關使用 deploytool 的更多資訊，請參閱位於以下位置的「The J2EE 1.4 Tutorial」中的「Security」一章：

http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html。

刪除範圍

若要刪除範圍，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：

   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。

   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。
3. 展開 [安全性] 節點。
4. 選取 [範圍] 節點。
5. 按一下要刪除的範圍旁邊的方塊。
6. 按一下 [刪除]。

等效 asadmin 指令爲：delete-auth-realm

設定預設範圍

預設範圍是當應用程式部署描述元未指定範圍時，Application Server 用於認證與授權的範圍。

若要設定預設範圍，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 選取 [安全性] 節點。

   將顯示 [安全性] 頁面。
4. 在 [預設範圍] 欄位中，從下拉式清單中選取所需的範圍。
5. 按一下 [儲存] 以儲存變更，或按一下 [載入預設值] 以刪除變更並復原 Application Server 預設值。
6. 如果主控台中顯示「需要重新啓動」，請重新啓動伺服器。

有關 JACC 提供者的管理主控台作業

- 建立 JACC 提供者
- 編輯 JACC 提供者
- 刪除 JACC 提供者
- 設定使用中的 JACC 提供者

建立 JACC 提供者

JACC (Java 容器授權合約) 屬於 J2EE 1.4 規格，它定義可插接式授權提供者介面。這可以讓管理員設置協力廠商外掛程式組以執行授權。依預設，Application Server 提供簡單的、JACC 相容的、基於檔案的認證引擎。

若要建立 JACC 提供者，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。

2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。
3. 展開 [安全性] 節點。
4. 選取 [JACC 提供者] 節點。
5. 在 [JACC 提供者] 頁面中，按一下 [新建]。
6. 在 [建立 JACC 提供者] 頁面中，輸入以下資訊：
   - **名稱** — 用於識別此提供者的名稱。
   - **策略配置** — 用於實作策略配置工廠的類別名稱。預設提供者使用 com.sun.enterprise.security.provider.PolicyConfigurationFactoryImpl。
   - **策略提供者** — 用於實作策略工廠的類別名稱。預設提供者使用 com.sun.enterprise.security.provider.PolicyWrapper。
7. 按一下 [增加特性] 按鈕向提供者增加特性。有效特性包括：
   - repository：包含策略檔案的目錄。對於預設提供者，此值爲 *install_dir*/domains/*domain_dir*/generated/policy。
8. 按一下 [確定] 以儲存此配置，或按一下 [取消] 以退出而不儲存變更。

編輯 JACC 提供者

若要編輯 JACC 提供者，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。
3. 展開 [安全性] 節點。
4. 展開 [JACC 提供者] 節點。
5. 選取要編輯的 JACC 提供者的節點。

6. 在 [編輯 JACC 提供者] 頁面中，依需要修改提供者資訊：
   - **策略配置** — 用於實作策略配置工廠的類別名稱。
   - **策略提供者** — 用於實作策略工廠的類別名稱。
7. 若要增加特性，請按一下 [增加] 按鈕。輸入特性的名稱和值。有效項目包括：
   - `repository`：包含策略檔案的目錄。對於預設提供者，該值爲 `${com.sun.aas.instanceRoot}/generated/policy`。
8. 若要刪除現有特性，請按一下特性左側的核取方塊，然後按一下 [刪除特性]。
9. 按一下 [儲存] 以儲存變更，或按一下瀏覽器的 [向後] 按鈕以取消而不儲存變更。

刪除 JACC 提供者

若要刪除 JACC 提供者，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 展開 [安全性] 節點。
4. 選取 [JACC 提供者] 節點。
5. 按一下要刪除的 JACC 提供者左側的核取方塊。
6. 按一下 [刪除]。

設定使用中的 JACC 提供者

若要指定 JACC 提供者，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。

b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。

3. 選取 [安全性] 節點。

   將顯示 [安全性] 頁面。

4. 在 [JACC] 欄位中，輸入伺服器要使用的 JACC 提供者的名稱。

   如果不知道哪些 JACC 提供者可用，請展開樹中的 [JACC 提供者] 元件來檢視所有已配置的 JACC 提供者。

5. 選取 [儲存] 以儲存變更，或者選取 [載入預設值] 以復原預設值。

6. 如果主控台中顯示「需要重新啓動」，請重新啓動 Application Server。

有關稽核模組的管理主控台作業

- 建立稽核模組
- 編輯稽核模組
- 刪除稽核模組
- 設定使用中稽核模組
- 啓用和停用稽核記錄

建立稽核模組

Application Server 提供了簡單的預設稽核模組；如需更多資訊，請參閱「使用預設稽核模組」。

若要建立新的稽核模組，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：

   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。

   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。

3. 展開 [安全性] 節點。
4. 選取 [稽核模組] 節點。
5. 在 [稽核模組] 頁面中，按一下 [新建]。

6. 在 [建立稽核模組] 頁面中，輸入以下資訊：
   - **名稱** — 用於識別此稽核模組的名稱。
   - **類別名稱** — 用於實作此模組的類別之完全合格名稱。預設稽核模組的類別名稱爲 `com.sun.enterprise.security.Audit`。
7. 若要將 JVM 特性增加至此模組，請按一下 [增加特性]。指定每個特性的名稱和值。有效特性包括：
   - `auditOn` — 指定是否啓用此實作類別。有效值爲 `true` 和 `false`。
8. 按一下 [確定] 以儲存項目，或按一下 [取消] 以退出而不儲存變更。

編輯稽核模組

依預設，未開啓稽核模組。如需有關如何啓動稽核模組的更多資訊，請參閱 「啓用和停用稽核記錄」。

若要編輯稽核模組，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 展開 [安全性] 節點。
4. 展開 [稽核模組] 節點。
5. 選取要編輯的稽核模組的節點。
6. 在 [編輯稽核模組] 頁面中，依需要修改類別名稱。
7. 選取 [增加] 按鈕並輸入特性的名稱和值來輸入模組的所有附加特性。有效特性包括：
   - `auditOn` — 指定是否使用此稽核模組。有效值爲 `true` 和 `false`。
8. 透過選取要修改的名稱或值並在文字欄位中直接輸入變更來修改任何現有特性。
9. 透過選取特性左側的核取方塊並按一下 [刪除特性] 來刪除特性。
10. 按一下 [儲存] 以儲存變更，或按一下瀏覽器上的 [向後] 按鈕以取消而不儲存變更。

刪除稽核模組

若要刪除稽核模組，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 展開 [安全性] 節點。
4. 選取 [稽核模組] 節點。
5. 按一下要刪除的稽核模組左側的核取方塊。
6. 按一下 [刪除]。

啟用和停用稽核記錄

若要指定伺服器使用的稽核模組，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 選取 [安全性] 節點。

   將顯示 [安全性] 頁面。
4. 若要啓用記錄，請選取 [稽核記錄] 核取方塊。若要停用記錄，請取消選取該核取方塊。選取此選項可以載入稽核模組並確定這些模組在稽核時被 Application Server 稽核程式庫呼叫。
5. 如果要啓用稽核記錄，請指定預設稽核模組 (如「設定使用中稽核模組」中所述)。
6. 選取 [儲存] 以儲存變更。
7. 如果主控台中顯示「需要重新啓動」，請重新啓動 Application Server。

設定使用中稽核模組

若要指定伺服器使用的稽核模組，請按照「啓用和停用稽核記錄」所述啓用稽核記錄，然後執行以下步驟：

1. 在 [稽核模組] 欄位中，輸入伺服器要使用的稽核模組的名稱。(預先配置的稽核模組被稱爲 `default`。) 確定已按照「啓用和停用預設稽核模組」所述將該稽核模組的 auditOn 設定爲 true。
2. 選取 [儲存] 以儲存變更，或選取 [載入預設值] 以取消變更。
3. 如果主控台中顯示「需要重新啓動」，請重新啓動 Application Server。

使用預設稽核模組

`default` 稽核模組將認證與授權請求記錄到伺服器記錄檔中。如需有關變更記錄檔位置的資訊，請參閱「配置一般記錄設定」。

認證記錄項目包括以下資訊：

- 已嘗試進行認證的使用者的名稱。
- 處理過存取請求的範圍。
- 請求的 Web 模組 URI 或 EJB 元件。
- 請求成功或失敗。

無論是否啓用稽核記錄，Application Server 都將記錄所有被拒絕的認證事件。

授權記錄項目包括以下資訊：

- 經過認證的使用者 (如果有) 的名稱。
- 請求的 Web URI 或 EJB 元件。
- 請求成功或失敗。

啟用和停用預設稽核模組

除了啓用記錄之外，還需要設定特定稽核模組所需的所有特性。如果是預設稽核模組，執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。

b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。

3. 展開 [安全性] 節點。
4. 展開 [稽核模組] 節點。
5. 按一下 [default] 節點。
6. 將 auditOn 特性的值設定爲 true。
7. 選取 [儲存] 以儲存變更。
8. 如果主控台中顯示「需要重新啓動」，請重新啓動 Application Server。

有關偵聽程式和 JMX 連接器的管理主控台作業

- 配置 HTTP 偵聽程式的安全性
- 配置 IIOP 偵聽程式的安全性
- 配置管理服務的 JMX 連接器的安全性
- 設定偵聽程式安全性特性

配置 HTTP 偵聽程式的安全性

HTTP 服務中的每個虛擬伺服器都透過一個或多個 *HTTP* **偵聽程式**提供網路連線。使用管理主控台可以建立新的 HTTP 偵聽程式和編輯現有 HTTP 偵聽程式的安全性設定。

若要編輯現有 HTTP 偵聽程式的安全性設定，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。
3. 展開 [HTTP 服務] 節點。
4. 選取 [HTTP 偵聽程式] 節點。
5. 選取 HTTP 偵聽程式以編輯現有偵聽程式，或者按一下 [新建] 並執行「建立 HTTP 偵聽程式」中的步驟以建立新的偵聽程式。

6. 按照「設定偵聽程式安全性特性」中的步驟設定安全性特性。
7. 按一下 [儲存] 以儲存變更，或按一下瀏覽器的 [向後] 按鈕以取消而不儲存變更。

等效 asadmin 指令爲：create-http-listener

配置 IIOP 偵聽程式的安全性

Application Server 支援 CORBA (共用物件請求代理程式架構) 物件，這類物件使用網際網路 Orb 交換協定 (IIOP) 在網路上進行通訊。*IIOP* **偵聽程式**接受來自 EJB 的遠端用戶端和其他基於 CORBA 用戶端的進來的連線。使用管理主控台可以建立新的 IIOP 偵聽程式和編輯現有 IIOP 偵聽程式的設定。

若要編輯 IIOP 偵聽程式的安全性特性，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 server，請展開 [server-config] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [default-config] 節點。
3. 展開 [ORB] 節點。
4. 選取 [IIOP 偵聽程式] 節點。
5. 選取 IIOP 偵聽程式以編輯該偵聽程式，或者按一下 [新建] 並執行「建立 IIOP 偵聽程式」中的步驟來建立新的偵聽程式。
6. 按照「設定偵聽程式安全性特性」中的步驟設定安全性特性。
7. 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以復原特性的預設值。

如果建立了新的偵聽程式，則新偵聽程式現在將列在 [IIOP 偵聽程式] 頁面的 [目前偵聽程式] 表中。

等效 asadmin 指令爲：create-iiop-listener

配置管理服務的 JMX 連接器的安全性

若要編輯管理服務的 JMX 連接器的安全性特性，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。

2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 展開 [管理服務] 節點。
4. 選取要修改的管理服務。
5. 按照「設定偵聽程式安全性特性」中的步驟設定安全性特性。
6. 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以復原特性的預設值。

設定偵聽程式安全性特性

按照以下設定 HTTP 偵聽程式、IIOP 偵聽程式和 JMX 連接器安全性特性的一般步驟進行操作：

1. 在 [編輯 HTTP 偵聽程式]、[編輯 IIOP 偵聽程式] 或 [編輯 JMX 連接器] 頁面中，移至標有 [SSL] 的區段。
2. 核取 [安全性] 欄位中的 [已啓用] 方塊來啓用該偵聽程式的安全性。選取此選項時，您必須選取 [SSL3] 或 [TLS] 來指定啓用何種類型的安全性，並且必須輸入證書暱稱。
3. 如果用戶端在使用此偵聽程式時要向 Application Server 認證自身，請在 [用戶端認證] 欄位中核取 [已啓用] 方塊。
4. 如果已核取 [已啓用] 方塊，請在 [證書暱稱] 欄位中輸入金鑰存放區別名。金鑰存放區別名是識別現有伺服器金鑰對和證書的單一值。預設金鑰存放區的證書暱稱爲 `s1as`。

   若要尋找證書暱稱，請使用 `certutil` 公用程式，如「關於 CertUtil 公用程式」所述。
5. 如果已選取 [已啓用] 方塊，請選取 [SSL3] 和/或 [TLS]。依預設，SSL3 和 TLS 均處於啓用狀態。
6. 依需要啓用個別密碼組。依預設，所有受支援的密碼組均處於啓用狀態。加密算法將在「關於加密算法」中論述：
7. 選取 [儲存] 以儲存變更，或選取 [載入預設值] 以取消變更。

有關虛擬伺服器的管理主控台安全性作業

- 配置單次登入 (SSO)

配置單次登入 (SSO)

單次登入可以讓多個應用程式共用使用者登入資訊，而不要求每個應用程式都單獨進行使用者登入。使用單次登入的應用程式對使用者進行一次認證，然後此認證資訊將被傳遞到其他所有涉及的應用程式。

單次登入適用於爲同一範圍和虛擬伺服器配置的 Web 應用程式。

備註：單次登入使用 HTTP cookie 來傳輸記號，該記號將每個請求與已儲存的使用者身份關聯起來，因此僅當瀏覽器用戶端支援 cookie 時才能使用單次登入。

單次登入根據以下規則運行：

- 使用者存取 Web 應用程式中受保護的資源時，伺服器會要求使用者使用爲該 Web 應用程式定義的方法對自身進行認證。
- 經過認證之後，Application Server 將使用與該使用者關聯的角色在虛擬伺服器中的所有 Web 應用程式之間進行授權決策，而不要求使用者單獨向每個應用程式進行認證。
- 使用者登出一個 Web 應用程式時 (明確登出，或由於階段作業過期而登出)，該使用者在所有 Web 應用程式中的階段作業都將變爲無效。此後，使用者需要先登入才能存取任一應用程式中的受保護資源。

依預設，Application Server 的單次登入處於啓用狀態。若要停用單次登入或配置其他特性，請執行以下步驟。

1. 在管理主控台的樹形元件中，展開 [配置] 節點。
2. 展開要配置的實例：
   a. 若要配置特定實例，請展開該實例的配置節點。例如，對於預設實例 `server`，請展開 [`server-config`] 節點。
   b. 若要爲所有實例配置預設設定，請展開 [`default-config`] 節點。
3. 展開 [HTTP 服務] 節點。
4. 展開 [虛擬伺服器] 節點，然後選取要配置爲支援單次登入的虛擬伺服器。
5. 按一下 [增加特性]。

   一個空白的特性項目將被增加至清單末尾。

6. 在 [名稱] 欄位中輸入 `sso-enable`。
7. 在 [值] 欄位中輸入 `false` 停用 SSO，輸入 `true` 啓用 SSO。依預設，SSO 處於啓用狀態。
8. 按一下 [增加特性] 並配置所有適用的 SSO 特性，以增加或變更任何其他單次登入特性。表 14-10 說明了有效的 SSO 特性。

表 14-10 虛擬伺服器 SSO 特性

特性名稱	描述	值
sso-max-inactive-seconds	如果未接收到任何用戶端活動，在可以清除使用者的單次登入記錄之前所等待的時間 (以秒為單位)。對虛擬伺服器中的任何應用程式的存取都可以使單次登入記錄保持使用中狀態。	預設值為 300 秒 (5 分鐘)。值越高，為使用者提供的持續性就越長，但伺服器上的記憶體消耗也會越多。
sso-reap-interval-seconds	清除過期的單次登入記錄的間隔 (以秒為單位)。	預設值為 60。

9. 按一下 [儲存]。
10. 如果主控台中顯示「需要重新啓動」，請重新啓動 Application Server。

有關連接器連線池的管理主控台作業

- 關於連接器連線池
- 關於安全性對映
- 建立安全性對映
- 編輯安全性對映
- 刪除安全性對映

關於連接器連線池

連接器模組 (也稱爲資源配接器) 允許 J2EE 應用程式與企業資訊系統 (EIS) 進行互動。**連接器資源**爲應用程式提供了一個 EIS 連線。**連接器連線池**是針對特定 EIS 的一組可重複使用的連線。

安全性對映可以建立 J2EE 使用者和群組與 EIS 使用者和群組之間的對映。使用管理主控台可以建立、更新、列示和刪除連接器連線池的安全性對映。

備註：在此環境中，使用者被稱爲主體。企業資訊系統 (EIS) 是保存資訊的任何系統。它可以是主機、訊息傳送系統、資料庫系統或應用程式。

關於安全性對映

使用安全性對映可以將應用程式 (主體或使用者群組) 的呼叫者身份對映到容器管理的基於作業事件的方案中適當的 EIS 主體。應用程式主體向 EIS 發出請求後，應用程式伺服器將首先使用爲連接器連線池定義的安全性對映檢查主體來確定已對映的後端 EIS 主體。如果沒有完全匹配的主體，應用程式伺服器將使用萬用字元規格 (如果有) 來確定已對映的後端 EIS 主體。應用程式使用者需要執行 EIS 作業 (需要以 EIS 中的特定身份來執行) 時，請使用安全性對映。

建立安全性對映

連接器連線池的安全性對映將應用程式使用者和群組 (主體) 對映到 EIS 主體。應用程式使用者需要執行 EIS 作業 (需要 EIS 中的特定身份) 時，請使用安全性對映。

若要爲指定的連接器連線池建立安全性對映，請執行以下步驟。

1. 展開 [資源] 節點。
2. 展開 [連接器] 節點。
3. 選取 [連接器連線池] 節點。
4. 透過從目前池清單中選取連接器連線池的名稱來選取一個連接器連線池，或透過從目前池清單中選取 [新建] 並按照「建立連接器連線池」中的說明建立新的連接器連線池。
5. 選取 [安全性對映] 頁面。
6. 按一下 [新建] 以建立新的安全性對映。
7. 在 [建立安全性對映] 頁面中，輸入以下特性。
    - **名稱** — 輸入用於參照此特定安全性對映的名稱。
    - **使用者群組** — 要對映到適當 EIS 主體的應用程式之呼叫者身份。輸入以逗號分隔的特定於應用程式的使用者群組清單，或輸入萬用字元星號 (*) 來表示所有使用者或所有使用者群組。指定 [主體] 或 [使用者群組] 選項，但不能同時指定這兩個選項。

- **主體** — 要對映到適當 EIS 主體的應用程式之呼叫者身份。輸入以逗號分隔的特定於應用程式的主體清單，或輸入萬用字元星號 (*) 來表示所有主體。指定 [主體] 或 [使用者群組] 選項，但不能同時指定這兩個選項。

8. 在 [後端主體] 區段中，輸入以下特性。
   - **使用者名稱** — 輸入 EIS 使用者名稱。企業資訊系統 (EIS) 是保存資訊的任何系統。它可以是主機、訊息傳送系統、資料庫系統或應用程式。
   - **密碼** — 輸入 EIS 使用者的密碼。
9. 按一下 [確定] 以建立安全性對映，或按一下 [取消] 以退出而不儲存變更。

等效 `asadmin` 指令爲：`create-connector-security-map`

編輯安全性對映

若要修改指定連接器連線池的安全性對映，請執行以下步驟。

1. 展開 [資源] 節點。
2. 展開 [連接器] 節點。
3. 選取 [連接器連線池] 節點。
4. 透過從目前池清單中選取連接器連線池的名稱來選取一個連接器連線池。
5. 選取 [安全性對映] 頁面。
6. 在 [安全性對映] 頁面中，從目前安全性對映清單中選取一個安全性對映。
7. 在 [編輯安全性對映] 頁面中，依需要修改以下特性。
   - **使用者群組** — 要對映到適當 EIS 主體的應用程式之呼叫者身份。輸入以逗號分隔的特定於應用程式的使用者群組清單，或輸入萬用字元星號 (*) 來表示所有使用者或所有使用者群組。指定 [主體] 或 [使用者群組] 選項，但不能同時指定這兩個選項。
   - **主體** — 要對映到適當 EIS 主體的應用程式之呼叫者身份。輸入以逗號分隔的特定於應用程式的主體清單，或輸入萬用字元星號 (*) 來表示所有主體。指定 [主體] 或 [使用者群組] 選項，但不能同時指定這兩個選項。
8. 在 [後端主體] 區段中，輸入以下特性。
   - **使用者名稱** — 輸入 EIS 使用者名稱。企業資訊系統 (EIS) 是保存資訊的任何系統。它可以是主機、訊息傳送系統、資料庫系統或應用程式。
   - **密碼** — 輸入 EIS 使用者的密碼。

9. 按一下 [儲存] 以儲存對安全性對映的變更。

有用的 asadmin 指令爲：`list-connector-security-maps`, `update-connector-security-maps`

刪除安全性對映

若要刪除指定連接器連線池的安全性對映，請執行以下步驟。

1. 展開 [資源] 節點。
2. 展開 [連接器] 節點。
3. 選取 [連接器連線池] 節點。
4. 透過從目前池清單中選取連接器連線池的名稱來選取一個連接器連線池。
5. 選取 [安全性對映] 頁面。
6. 在 [安全性對映] 頁面中，按一下要刪除的安全性對映名稱左側的核取方塊。
7. 按一下 [刪除]。

等效 asadmin 指令爲：`delete-connector-security-map`

使用證書和 SSL

- 關於證書檔案
- 關於 Keytool 公用程式
- 產生伺服器證書
- 對數位證書進行簽名
- 刪除證書

關於證書檔案

安裝 Application Server 時將產生一個適用於內部測試的 NSS 格式的數位證書。依預設，Application Server 將其證書資訊儲存到 *install_dir*/domains/*domain_name*/config 目錄中的兩個檔案中：

- **金鑰存放區檔案** — 依預設，名爲 `key3.db`，包含 Application Server 數位證書及其私密金鑰。金鑰存放區檔案受密碼保護。使用 `asadmin change-master-password` 指令可以變更該密碼。如需有關 `certutil` 的更多資訊，請參閱、「關於 CertUtil 公用程式」。

  每個金鑰存放區項目都有唯一的別名。安裝後，Application Server 金鑰存放區會有一個別名爲 `s1as` 的項目。

- **信任儲存檔案** — 依預設，名爲 `cert8.db`，包含 Application Server 所信任的證書及其他實體的公開金鑰。對於受信任的證書，伺服器已確認證書中的公開金鑰屬於證書所有者。受信任的證書通常包括那些憑證授權單位 (CA) 的證書。

在 Platform Edition 的伺服器端，Application Server 使用 JSSE 格式，該格式使用 `keytool` 來管理證書和金鑰存放區。在 Enterprise Edition 的伺服器端，Application Server 使用 NSS 格式，該格式使用 `certutil` 來管理儲存私密金鑰和證書的 NSS 資料庫。兩種版本的用戶端 (應用程式用戶端或獨立用戶端) 上均使用 JSSE 格式。

依預設，使用金鑰存放區和信任儲存對 Application Server 進行配置，金鑰存放區和信任儲存將與範例應用程式配合使用並可用於開發。在用於生產目的時，您可能需要變更證書別名、將其他證書增加至信任儲存或變更金鑰存放區檔案和信任儲存檔案的名稱和/或位置。

變更證書檔案的位置

供開發使用的金鑰存放區檔案和信任儲存檔案儲存在 *install_dir*`/domains/`*domain_name*`/config` 目錄中。若要變更金鑰存放區檔案和信任儲存檔案的名稱和/或位置，請執行以下步驟。

1. 在 [管理主控台] 樹中，展開 [配置]。
2. 展開 [server-config] (管理配置) 節點。
3. 選取 [JVM 設定] 節點。
4. 按一下 [JVM 選項] 標籤。
5. 在 [JVM 選項] 頁面中，在 [值] 欄位中增加或修改以下值來反映證書檔案的新位置：

   `-Dcom.sun.appserv.nss.db=${com.sun.aas.instanceRoot}/`*NSS_database_directory*

   其中，*ks_name* 是金鑰存放區檔案名稱，*ts_name* 是信任儲存檔案名稱。
6. 按一下 [儲存]。

7. 如果主控台中顯示「需要重新啓動」，請重新啓動 Application Server。

關於 Keytool 公用程式

在 Platform Edition 中使用 keytool 可以設定和使用 JSSE 數位證書。J2SE SDK 附帶了 keytool，因而可以讓管理員管理公開/私密金鑰對和關聯的證書。還可以讓使用者快取正與其通訊的另一方的公開金鑰 (以證書形式)。

若要執行 keytool，必須先配置 shell 環境，以使 J2SE /bin 目錄位於路徑中，或者指令行中必須存在該工具的完整路徑。如需有關 keytool 的更多資訊，請參閱位於以下位置的 keytool 文件：

http://java.sun.com/j2se/1.4.2/docs/tooldocs/solaris/keytool.html

關於 CertUtil 公用程式

僅在 **Enterprise Edition** 中可以使用 certutil 來設定和使用 NSS 數位證書。證書資料庫工具 certutil 是可以建立和修改 Netscape Communicator cert8.db 和 key3.db 資料庫檔案的指令行公用程式。該公用程式還可以列示、產生、修改或刪除 cert8.db 檔案中的證書，並可以建立或變更密碼、產生新的公開和私密金鑰對、顯示金鑰資料庫的內容或刪除 key3.db 檔案中的金鑰對。

金鑰和證書管理程序通常以在金鑰資料庫中建立金鑰開始，然後在證書資料庫中產生和管理證書。下面列示的文件論述了如何使用 NSS 管理證書和金鑰資料庫，包括 certutil 公用程式的語法：

http://www.mozilla.org/projects/security/pki/nss/tools/certutil.html

pk12util 是指令行公用程式，用於以 PKCS12 格式在證書 / 金鑰資料庫和檔案之間匯入和匯出金鑰及證書。有關 pk12util 公用程式的更多描述，請參閱：

http://www.mozilla.org/projects/security/pki/nss/tools/pk12util.html

如需有關使用 certutil、pk12util 和其他 NSS 安全性工具的更多資訊，請參閱位於 http://www.mozilla.org/projects/security/pki/nss/tools 上的「NSS Security Tools」。

這些工具位於 *install_dir*/lib/ 目錄。

產生伺服器證書

使用 certutil 可以產生、匯入和匯出證書。請參閱 「關於 CertUtil 公用程式」，以取得有關如何執行這些作業的更多資訊。

對數位證書進行簽名

建立數位證書之後，所有者必須爲其簽名以防止僞造。電子商業站台或那些身份認證對其很重要的站台可以從知名的憑證授權單位 (CA) 購買證書。如果無需考慮認證，例如當私密安全通訊可以滿足全部需求時，則可節省獲取 CA 證書所花費的時間和費用並使用自簽名證書。

使用 CA 證書

若要使用由 CA 簽名的數位證書，請執行以下步驟：

1. 按照 CA 網站上的說明進行操作以產生證書金鑰對。
2. 下載產生的證書金鑰對。

   將證書儲存在包含伺服器金鑰存放區檔案和信任儲存檔案的目錄中，依預設該目錄爲 *install_dir*/domains/*domain-dir*/config 目錄。請參閱 「變更證書檔案的位置」，以取得有關變更此位置的說明。
3. 在 shell 中，變更至包含證書的目錄。
4. 使用 certutil 將證書匯入至本地金鑰存放區和本地信任儲存 (如有必要)。
5. 重新啓動 Application Server。

如需有關使用 certutil 的完整資訊，請參閱位於以下位置的 certutil 文件：

http://www.mozilla.org/projects/security/pki/nss/tools/certutil.html

刪除證書

若要刪除現有證書，請使用 certutil 公用程式。如需有關 certutil 公用程式的更多資訊，請參閱 「關於 CertUtil 公用程式」。

更多資訊

- 可以透過以下 URL 檢視 Java 2 Standard Edition 的安全性論述：
  http://java.sun.com/j2se/1.4.2/docs/guide/security/index.html
- 可以透過以下 URL 檢視「J2EE 1.4 Tutorial」的「Security」一章：
  http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html
- 本「**管理指南**」的「配置訊息安全性」一章。
- 「Developer's Guide」的「Securing Applications」一章。

第 15 章

配置訊息安全性

本章描述如何在 Sun Java System Application Server 8.1 2005Q1 中爲 Web 服務配置訊息層安全性。本章包含以下主題：

- 關於訊息安全性
- 用於訊息安全性的管理主控台作業

本章中的某些內容假設您對安全性和 Web 服務概念已有基本的瞭解。若要深入瞭解這些概念，請在開始閱讀本章之前先仔細閱讀「詳細資訊」中列出的內容。

關於訊息安全性

- 訊息安全性概況
- 瞭解 Application Server 中的訊息安全性
- 確保 Web 服務的安全
- 確保範例應用程式的安全
- 配置 Application Server 以實現訊息安全性

訊息安全性概況

在**訊息安全性**中，安全性資訊會插入到訊息中，以使其透過網路層傳輸並附帶訊息到達訊息目標。訊息安全性與傳輸層安全性不同 (在「J2EE 1.4 Tutorial」的「Security」一章中有說明)，因爲訊息安全性可用於將訊息保護從訊息傳輸中分離開，從而使訊息在傳輸後仍保持受保護狀態。

Web **服務安全性：***SOAP* **訊息安全性** *(WS-Security)* 爲互通 Web 服務安全性的國際標準，是由 Web 服務技術的所有主要提供者 (包括 Sun Microsystems) 在 OASIS 中協作開發的。WS-Security 是一種訊息安全性機制，它使用 XML 加密和 XML 數位簽名來確保經由 SOAP 傳送的 Web 服務訊息的安全。WS-Security 規格定義了多種安全性記號 (包括 X.509 證書、SAML 假設和使用者名稱 / 密碼記號) 的使用，以認證和加密 SOAP Web 服務訊息。

可透過以下 URL 檢視 WS-Security 規格：

http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf

瞭解 Application Server 中的訊息安全性

Sun Java System Application Server 8.1 2005Q1 在其 Web 服務用戶端和伺服器端容器中提供對 WS-Security 標準的整合式支援。整合此功能後，Web 服務安全性可由代表應用程式的 Application Server 的容器來強制執行，並且可以用於保護任何 Web 服務應用程式，無需變更應用程式的實作。Application Server 透過提供工具將 SOAP 層訊息安全性提供者和訊息保護策略連結到容器和容器中部署的應用程式，來達到此效果。

指定訊息安全性職責

在 Sun Java System Application Server 8.1 2005Q1 中，系統管理員和應用程式部署人員角色應主要負責配置訊息安全性。儘管在通常情況下，其他兩個角色之一在不涉及到開發者的條件下就可以確保現有應用程式的安全，且無需變更其實作，但在某些情況下，應用程式開發人員可能也會有所作用。以下小節定義了各種角色的職責：

- 系統管理員
- 應用程式部署人員
- 應用程式開發人員

系統管理員

系統管理員負責：

- 在 Application Server 中配置訊息安全性提供者。
- 管理使用者資料庫。
- 管理金鑰儲存區和信任儲存區檔案。

- 在使用加密並執行 1.5.0 版之前的 Java SDK 時，配置 Java 加密擴展 (JCE) 提供者。
- 安裝範例伺服器。僅在 xms 範例應用程式用於說明如何使用訊息層 Web 服務安全性時，才執行此操作。

系統管理員使用管理主控台來管理伺服器安全性設定，並使用指令行工具來管理證書資料庫。在 PE 中，證書和私密金鑰儲存在金鑰儲存區中，並由 keytool 進行管理。SE 和 EE 將證書和私密金鑰儲存在 NSS 資料庫中，並使用 certutil 對其進行管理。本文件主要適用於系統管理員。如需有關訊息安全性作業的概況，請參閱「配置 Application Server 以實現訊息安全性」。

應用程式部署人員

應用程式部署人員負責：

- 如果上游角色 (開發者或組譯者) 尚未指定任何所需的特定於應用程式的訊息保護策略，則 (在應用程式組譯時) 指定這些策略。
- 修改特定於 Sun 的部署描述元，以向 Web 服務端點和服務參照指定特定於應用程式的訊息保護策略資訊 (即 message-security-binding 元素)。

「Developer's Guide」的「Securing Applications」一章中將討論這些安全性作業。
(如需有關指向該章的連結，請參閱 「詳細資訊」。)

應用程式開發人員

應用程式開發者可以啓用訊息安全性，但並不是必須要這樣做。訊息安全性可以由系統管理員設定，以便確保所有 Web 服務的安全；如果連結到應用程式的提供者或保護策略不同於連結到容器的提供者或保護策略，訊息安全性則由應用程式部署人員設定。

應用程式開發者或組譯者負責：

- 確定應用程式是否需要特定於應用程式的訊息保護策略。如果需要，則透過與應用程式部署人員交流，確保在應用程式組譯時指定所需策略。

關於安全性記號和安全性機制

WS-Security 規格爲使用安全性記號來認證和加密 SOAP Web 服務訊息提供了可延伸機制。可以使用與 Application Server 一起安裝的 SOAP 層訊息安全性提供者來選取使用者名稱 / 密碼和 X509 證書安全性記號，以認證和加密 SOAP Web 服務訊息。選取其他安全性記號 (包括 SAML 假設) 的其他提供者將與 Application Server 的後續版本一起安裝。

關於使用者名稱記號

Application Server 使用 SOAP 訊息中的**使用者名稱記號**來建立訊息**寄件者**的認證身份。包含使用者名稱記號 (在內嵌密碼中) 的訊息收信人透過確認寄件者是否知道使用者的秘密 (即密碼)，來驗證訊息寄件者是否經過授權成爲使用者 (在記號中識別)。

使用使用者名稱記號時，必須在 Application Server 中配置有效的使用者資料庫。如需有關該主題的更多資訊，請參閱 「編輯範圍」。

關於數位簽名

Application Server 使用 XML 數位簽名將認證身份連結到訊息**內容**。用戶端使用數位簽名來建立呼叫者身份，其方法與使用傳輸層安全性時用基本認證或 SSL 用戶端證書認證建立呼叫者身份的方法相似。訊息收件者將驗證數位簽名以認證訊息內容的源 (可能與訊息寄件者不同)。

使用數位簽名時，必須在 Application Server 中配置有效的金鑰儲存區和信任儲存區檔案。如需有關該主題的更多資訊，請參閱 「關於證書檔案」。

關於加密

加密的目的是將資料修改爲只有目標讀者才能理解的形式。加密程序透過用加密元素替換原始內容來完成。與公開金鑰加密指出的那樣，可以使用加密來建立能夠閱讀訊息的一方或多方身份。

使用加密時，必須先安裝支援加密的 JCE 提供者。如需有關該主題的更多資訊，請參閱 「配置 JCE 提供者」。

關於訊息保護策略

訊息保護策略是針對請求訊息處理和回應訊息處理而定義的，並根據對源和/或收信人認證的需求來進行表示。來源認證策略代表一個需求，即必須在訊息中建立已傳送訊息或已定義訊息內容的實體之身份，以使其可以由訊息收件者進行認證。收信人認證策略代表一個需求，即必須傳送訊息，以使可以接收訊息的實體之身份可由訊息寄件者建立。提供者套用特定的訊息安全性機制，以使訊息保護策略在 SOAP Web 服務訊息的上下文中實現。

請求和回應訊息保護策略是在將提供者配置到容器時定義的。特定於應用程式的訊息保護策略 (細緻程度爲 Web 服務連接埠或作業) 也可以在應用程式或應用程式用戶端的特定於 Sun 的部署描述元中進行配置。在任何情況下，如果定義了訊息保護策略，則用戶端的請求和回應訊息保護策略必須與伺服器的請求和回應訊息保護策略相匹配 (即二者相當)。如需有關定義特定於應用程式的訊息保護策略的更多資訊，請參閱 「Developer's Guide」的 「Securing Applications」一章。「詳細資訊」包含指向該章的連結。

訊息安全性術語字彙表

以下內容描述本文件中所使用的術語。「配置 Application Server 以實現訊息安全性」中也說明了這些概念。

- 認證層

  認證層是必須執行認證處理的訊息層。Application Server 在 SOAP 層強制執行 Web 服務訊息安全性。

- 認證提供者

  在此版本的 Sun Java Systems Application Server 中，Application Server 呼叫**認證提供者**來處理 SOAP 訊息層安全性。

  - **用戶端提供者**用於建立 (透過簽名或使用者名稱/密碼) 請求訊息的源身份和/或保護 (透過加密) 請求訊息，從而使這些訊息只能由其目標收信人檢視。用戶端提供者還可以將其容器建立爲接收的回應之授權收信人 (透過成功地解密)，並驗證回應中的密碼或簽名以認證與回應相關聯的源身份。在 Application Server 中配置的用戶端提供者**可**用來保護由做爲其他服務的用戶端的伺服器端元件 (即 Servlet 和 EJB) 所傳送的請求訊息和所接收的回應訊息。

  - **伺服器端提供者**用於將其容器建立爲接收的請求之授權收信人 (透過成功地解密)，並驗證請求中的密碼或簽名以認證與該請求相關聯的源身份。伺服器端提供者還將建立 (透過簽名或使用者名稱/密碼) 回應訊息的源身份和/或保護 (透過加密) 回應訊息，以使這些訊息只能由其目標收信人檢視。伺服器端提供者僅由伺服器端容器呼叫。

- 預設伺服器提供者

  預設伺服器提供者用於爲尚未連結特定伺服器提供者的任何應用程式識別要呼叫的伺服器提供者。**預設伺服器提供者**有時被稱爲**預設提供者**。

- 預設用戶端提供者

  預設用戶端提供者用於爲尚未連結特定用戶端提供者的任何應用程式識別要呼叫的用戶端提供者。

- 請求策略

  請求策略定義與認證提供者執行的請求處理相關聯的認證策略需求。按照訊息寄件者的順序表示這些策略，從而使內容之後出現的加密需求意味著訊息收件者將在驗證簽名之前先要對訊息進行解密。

- 回應策略

  回應策略定義與認證提供者執行的回應處理相關聯的認證策略需求。按照訊息寄件者的順序表示這些策略，從而使內容之後出現的加密需求意味著訊息收件者將在驗證簽名之前先要對訊息進行解密。

確保 Web 服務的安全

透過將 SOAP 層訊息安全性提供者和訊息保護策略連結到部署有應用程式的容器或連結到應用程式提供的 Web 服務端點，來確保在 Application Server 中部署的 Web 服務的安全。透過將 SOAP 層訊息安全性提供者和訊息保護策略連結到用戶端容器或連結到由用戶端應用程式宣告的可攜式服務參照，從而在 Application Server 的用戶端容器中配置 SOAP 層訊息安全性功能。

安裝了 Application Server 後，將在 Application Server 的用戶端和伺服器端容器中配置 SOAP 層訊息安全性提供者，其中這些提供者可由容器或者由容器中部署的各個應用程式或用戶端進行連結使用。在安裝程序中，這些提供者配置簡單的訊息保護策略，即如果連結到容器或者連結到容器中的應用程式或用戶端，則將導致所有請求和回應訊息中內容的源均由 XML 數位簽名進行認證。

可以採用 Application Server 的管理介面，從而連結現有提供者以供 Application Server 的伺服器端容器使用，修改由提供者強制執行的訊息保護策略，或使用替代的訊息保護策略來建立新的提供者配置。「用於訊息安全性的管理主控台作業」中定義了這些作業。可以在應用程式用戶端容器的 SOAP 訊息層安全性配置上執行類似的管理作業 (如「啓用用戶端應用程式的訊息安全性」中所定義)。

依預設，Application Server 中的訊息層安全性處於停用狀態。若要配置 Application Server 的訊息層安全性，請按照「配置 Application Server 以實現訊息安全性」中列出的步驟進行操作。如果您要使 Web 服務安全性用於保護在 Application Server 上部署的所有 Web 服務應用程式，請按照「啓用訊息安全性的提供者」和「啓用用戶端應用程式的訊息安全性」中的步驟進行操作。

完成以上步驟 (可能包括重新啓動 Application Server) 後，Web 服務安全性將用於在 Application Server 上部署的所有 Web 服務應用程式。

配置特定於應用程式的 Web 服務安全性

透過在應用程式的特定於 Sun 的部署描述元中定義 message-security-binding 元素，來配置特定於應用程式的 Web 服務安全性功能 (在應用程式組譯時)。這些 message-security-binding 元素用於將特定提供者或訊息保護策略與 Web 服務端點或服務參照相關聯，並符合要求以使其套用到相應端點或參照的服務的特定連接埠或方法。

如需有關定義特定於應用程式的訊息保護策略的更多資訊，請參閱「Developer's Guide」的「Securing Applications」一章。「詳細資訊」包含指向該章的連結。

確保範例應用程式的安全

Application Server 隨附名爲 xms 的範例應用程式。xms 應用程式具有簡單的 Web 服務功能，它由 J2EE EJB 端點和 Java Servlet 端點共同實作。這兩個端點共用同一個服務端點介面。服務端點介面定義了單一作業 (sayHello)，此作業將獲取字串引數，並返回由呼叫引數加前置的 Hello 所組成的 String。

xms 範例應用程式用於說明 Application Server 的 WS-Security 功能之使用，以確保現有 Web 服務應用程式的安全。範例隨附的說明描述了如何啓用 Application Server 的 WS-Security 功能，以使其用於確保 xms 應用程式的安全。範例還說明了 WS-Security 功能到應用程式的直接連結 (如「配置特定於應用程式的 Web 服務安全性」中所述)，以使其特定套用於應用程式。

xms 範例應用程式安裝在以下目錄中：
install_dir\samples\webservices\security\ejb\apps\xms\

如需有關編譯、封裝和執行 xms 範例應用程式的資訊，請參閱「Developer's Guide」的「Securing Applications」一章。「詳細資訊」包含指向該章的連結。

配置 Application Server 以實現訊息安全性

Application Server 使用 SOAP 處理層中整合的訊息安全性提供者來實作訊息安全性。訊息安全性提供者取決於 Application Server 的其他安全性工具。

若要配置這些其他工具，請執行以下步驟：

1. 如果使用的 Java SDK 版本早於 1.5.0，而且使用了加密技術，請配置 JCE 提供者。

   「配置 JCE 提供者」中說明了如何配置 JCE 提供者。

2. 如果使用了使用者名稱記號，請在必要時配置使用者資料庫。使用使用者名稱 / 密碼記號時，必須配置適當的範圍，並爲該範圍配置適當的使用者資料庫。

   「編輯範圍」中說明了如何配置使用者資料庫。

3. 管理證書和私密金鑰 (如有必要)。

   「關於證書檔案」中說明了如何管理證書和私密金鑰。

如果已將 Application Server 的工具配置爲供訊息安全性提供者使用，則可能會啓用隨 Application Server 一起安裝的提供者 (如「啓用訊息安全性的提供者」中所述)。

配置 JCE 提供者

J2SE 1.4.x 中包括的 Java 加密擴展 (JCE) 提供者不支援 RSA 加密。由於由 WS-Security 定義的 XML 加密通常是基於 RSA 加密，因此，若要使用 WS-Security 來加密 SOAP 訊息，您必須下載並安裝支援 RSA 加密的 JCE 提供者。

備註： RSA 是 RSA Data Security, Inc. 開發的公開金鑰加密技術。RSA 的首字母縮略分別代表該技術的三位發明者：Rivest、Shamir 和 Adelman。

如果是在 1.5 版的 Java SDK 中執行 Application Server，則 JCE 提供者已進行正確配置。如果是在 1.4.x 版的 Java SDK 中執行 Application Server，請執行下列步驟以靜態方式將 JCE 提供者增加爲 JDK 環境的一部分。

1. 下載並安裝 JCE 提供者 JAR (Java 歸檔) 檔案。透過以下 URL 可以獲得支援 RSA 加密的 JCE 提供者之清單：

   `http://java.sun.com/products/jce/jce14_providers.html`

2. 將 JCE 提供者 JAR 檔案複製到 *<JAVA_HOME>*`/jre/lib/ext/` 中。

3. 停止 Application Server。如果未停止 Application Server 而後來又在該程序中重新啓動了 Application Server，則 Application Server 將無法識別 JCE 提供者。

4. 在任何一種文字編輯器中編輯 *<JAVA_HOME>*`/jre/lib/security/java.security` 特性檔案。將剛下載的 JCE 提供者增加到此檔案。`java.security` 檔案包含用於增加該提供者的詳細說明。通常，您需要在具有類似特性的某個位置處增加一行，其格式如下：

   ```
   security.provider.<n>=<provider class name>
   ```

   在本範例中，*<n>* 是 Application Server 評估安全性提供者時所使用的喜好設定順序。將剛才增加的 JCE 提供者的 *<n>* 值設定爲 2。

   例如，如果下載的是 The Legion of the Bouncy Castle JCE 提供者，則應增加以下行。

   ```
   security.provider.2=org.bouncycastle.jce.provider.
   BouncyCastleProvider
   ```

   確定將 Sun 安全性提供者保持在最高的喜好設定順序，其值爲 1。

   ```
   security.provider.1=sun.security.provider.Sun
   ```

   將其他安全性提供者的層級調低，從而使每個層級上只有一個安全性提供者。

以下是提供必要 JCE 提供者並將現有提供者保持在正確位置的 `java.security` 檔案範例。

```
security.provider.1=sun.security.provider.Sun
security.provider.2=org.bouncycastle.jce.provider.
BouncyCastleProvider
security.provider.3=com.sun.net.ssl.internal.ssl.Provider
security.provider.4=com.sun.rsajca.Provider
security.provider.5=com.sun.crypto.provider.SunJCE
security.provider.6=sun.security.jgss.SunProvider
```

5. 儲存並關閉檔案。
6. 重新啓動 Application Server。

用於訊息安全性的管理主控台作業

爲使用訊息安全性而對 Application Server 設定的大多數步驟都可以透過使用管理主控台、`asadmin` 指令行工具或透過手動編輯系統檔案來完成。通常，不建議使用編輯系統檔案，因爲它可能會做出無意識的變更而使 Application Server 不能正常工作，因此，如有可能，建議優先選擇使用管理主控台來配置 Application Server，然後選擇使用 `asadmin` 工具指令。僅在無管理主控台或等效 `asadmin` 時，才使用手動編輯系統檔案。

以 (可插接式) 認證模組的形式將訊息層安全性支援整合到 Application Server 及其用戶端容器中。依預設，Application Server 中的訊息層安全性處於停用狀態。以下小節提供用於啓用、建立、編輯和刪除訊息安全性配置和提供者的詳細資訊。

- 啓用訊息安全性的提供者
- 配置訊息安全性提供者
- 建立訊息安全性提供者
- 刪除訊息安全性配置
- 刪除訊息安全性提供者
- 啓用用戶端應用程式的訊息安全性

大多數情況下，在執行以上管理作業後應重新啓動 Application Server。它尤其適用於當執行完作業時，您要將管理變更的效果套用到 Application Server 上已部署的應用程式中的情況。

啟用訊息安全性的提供者

若要爲在 Application Server 中部署的 Web 服務端點啓用訊息安全性，則必須指定要在伺服器端依預設使用的提供者。如果啓用了訊息安全性的預設提供者，您還需要啓用由 Application Server 中部署的 Web 服務之用戶端所使用的提供者。「啓用用戶端應用程式的訊息安全性」中說明了有關啓用用戶端使用的提供者之資訊。

若要啓用源自已部署端點的 Web 服務呼叫之訊息安全性，您必須指定預設用戶端提供者。如果已爲 Application Server 啓用了預設用戶端提供者，則必須確定從 Application Server 中部署的端點所呼叫的所有服務均配置爲與訊息層安全性相容。

若要爲 Application Server 啓用預設提供者，請執行以下步驟。

1. 在管理主控台樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 展開 [安全性] 節點。
4. 展開 [訊息安全性] 節點。
5. 選取 [SOAP] 節點。
6. 選取 [訊息安全性] 標籤。
7. 在 [編輯訊息安全性配置] 頁面中，爲尚未連結特定提供者的所有應用程式，指定一個要用於伺服器端的提供者和一個要用於用戶端的提供者。透過修改以下選擇性特性來完成此操作：
   - **預設提供者** — 用於爲尚未連結特定伺服器提供者的任何應用程式識別要呼叫的伺服器提供者。

     依預設，沒有爲 Application Server 選取任何提供者配置。若要識別伺服器端提供者，請選取 `ServerProvider`。選取空選項意味著不會在伺服器端 (依預設) 呼叫訊息安全性提供者。

     在該欄位中，通常應選取 `ServerProvider`。

- **預設用戶端提供者** — 用於爲尚未連結特定伺服器提供者的任何應用程式識別要呼叫的用戶端提供者。

  依預設，沒有爲 Application Server 選取任何提供者配置。若要識別用戶端提供者，請選取 `ClientProvider`。選取空選項意味著不會在用戶端 (依預設) 呼叫訊息安全性提供者。

  在該欄位中，通常應選取空值。如果您要啓用預設提供者和訊息保護策略以套用到源自部署於 Application Server 上的 Web 服務端點之 Web 服務呼叫，則應選取 ClientProvider。

8. 按一下 [儲存]。
9. 如果您已啓用用戶端或伺服器提供者，並且要修改已啓用的提供者之訊息保護策略，請參閱「配置訊息安全性提供者」，以獲得有關修改在此步驟中所啓用的訊息安全性提供者之配置的資訊。

等效的 asadmin 指令爲：

- 若要指定預設伺服器提供者，請使用：

  ```
  asadmin set --user <admin-user> --port <admin-port>
  server-config.security-service.message-security-config.SOAP.
  default_provider=ServerProvider
  ```

- 若要指定預設用戶端提供者，請使用：

  ```
  asadmin set --user <admin-user> --port <admin-port>
  server-config.security-service.message-security-config.SOAP.
  default_client_provider=ClientProvider
  ```

配置訊息安全性提供者

通常應重新配置提供者以修改其訊息保護策略 (儘管提供者類型、實作類別和特定於提供者的配置特性可能也需要修改)。請按以下步驟重新配置訊息安全性提供者。

1. 在管理主控台樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 展開 [安全性] 節點。
4. 展開 [訊息安全性] 節點。

5. 選取 [SOAP] 節點。
6. 選取 [提供者] 標籤。
7. 選取要編輯的訊息安全性提供者。Application Server 隨附 `ClientProvider` 和 `ServerProvider`。
8. 在 [編輯提供者配置] 頁面的 [提供者配置] 區段，可以修改以下特性：
   - **提供者類型** — 選取 `client`、`server` 或 `client-server`，以確定是將提供者用做用戶端認證提供者、伺服器認證提供者還是兼用做兩者 (主從式提供者)。
   - **類別名稱** — 輸入提供者的 Java 實作類別。用戶端認證提供者必須實作 `com.sun.xml.wss.provider.ClientSecurityAuthModule` 介面。伺服器端提供者必須實作 `com.sun.xml.wss.provider.ServerSecurityAuthModule` 介面。一個提供者可以同時實作這兩種介面，但必須實作對應於其提供者類型的介面。
9. 在 [建立提供者配置] 頁面的 [請求策略] 區段，根據需要輸入以下**選擇性**值。這些特性是選擇性的，但如果未進行指定，則不會對請求訊息套用任何認證。

   請求策略定義與認證提供者執行的請求處理相關聯的認證策略需求。按照訊息寄件者的順序表示這些策略，從而使內容之後出現的加密請求意味著訊息收件者將在驗證簽名之前先要對訊息進行解密。
   - **認證來源** — 選取 `sender`、`content` 或空值 (空白選項)，以定義對訊息層寄件者認證的需求 (例如使用者名稱密碼)、內容認證 (例如數位簽名) 或不對請求訊息進行任何認證。如果指定爲空值，則不需要進行請求的來源認證。
   - **認證收信人** — 選取 `beforeContent` 或 `afterContent`，以定義寄件者對請求訊息的收件者之訊息層認證的需求 (例如，透過 XML 加密)。如果不指定該值，則其爲預設值 `afterContent`。

   如需有關以下訊息保護策略所導致的由 SOAP 訊息安全性提供者執行的動作之描述，請參閱 「請求策略配置和回應策略配置的動作」。
10. 在 [建立提供者配置] 頁面的 [回應策略] 區段，根據需要輸入以下**選擇性**特性。這些特性是選擇性的，但如果未進行指定，則不會對回應訊息套用任何認證。

    回應策略定義與認證提供者執行的回應處理相關聯的認證策略需求。按照訊息寄件者的順序表示這些策略，從而使內容之後出現的加密請求意味著訊息收件者將在驗證簽名之前先要對訊息進行解密。
    - **認證來源** — 選取 `sender`、`content` 或空值 (空白選項)，以定義要套用到回應訊息的訊息層寄件者認證 (例如使用者名稱密碼) 或內容認證 (例如數位簽名) 之需求。如果指定爲空值，則不需要進行回應的來源認證。

- **認證收信人** — 選取 beforeContent 或 afterContent，以定義寄件者對回應訊息的收件者之訊息層認證的需求 (例如，透過 XML 加密)。如果不指定該值，則其爲預設值 afterContent。

如需有關以下訊息保護策略所導致的由 SOAP 訊息安全性提供者執行的動作之描述，請參閱 「請求策略配置和回應策略配置的動作」。

11. 透過按一下 [增加特性] 按鈕增加其他特性。Application Server 隨附的提供者可支援下面列出的特性。如果使用的是其他提供者，請參閱其他提供者的文件，以獲得有關特性和有效值的更多資訊。

    - server.config — 包含伺服器配置資訊的 XML 檔案之目錄和檔案名稱。例如，*install_dir*/domains/*domain_dir*/config/wss-server-config.xml。

12. 按一下 [儲存]。

下面列出了等效的 asadmin 指令。若要設定回應策略，請用 response 一詞替代以下指令中的 request 一詞。

- 若要將請求策略增加到用戶端並設定認證來源，請使用：

  ```
  asadmin set --user <admin-user> --port <admin-port>
  server-config.security-service.message-security-config.SOAP.
  provider-config.ClientProvider.request-policy.auth_source=
  <sender | content>
  ```

- 若要將請求策略增加到伺服器並設定認證來源，請使用：

  ```
  asadmin set --user <admin-user> --port <admin-port>
  server-config.security-service.message-security-config.SOAP.
  provider-config.ServerProvider.request-policy.auth_source=
  <sender | content>
  ```

- 若要將請求策略增加到用戶端並設定認證收信人，請使用：

  ```
  asadmin set --user <admin-user> --port <admin-port>
  server-config.security-service.message-security-config.SOAP.
  provider-config.ClientProvider.request-policy.auth_recipient=
  <before-content | after-content>
  ```

- 若要將請求策略增加到伺服器並設定認證收信人，請使用：

  ```
  asadmin set --user <admin-user> --port <admin-port>
  server-config.security-service.message-security-config.SOAP.
  provider-config.ServerProvider.request-policy.auth_recipient=
  <before-content | after-content>
  ```

建立訊息安全性提供者

若要建立新的訊息安全性提供者，請執行以下步驟。若要配置現有提供者，請執行「配置訊息安全性提供者」中的步驟。

1. 在管理主控台樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 展開 [安全性] 節點。
4. 展開 [訊息安全性] 節點。
5. 選取 [SOAP] 節點。
6. 選取 [提供者] 標籤。
7. 在 [提供者配置] 頁面中，按一下 [新建]。
8. 在 [建立提供者配置] 頁面的 [提供者配置] 區段，輸入以下資訊：
   - **預設提供者** — 核取此欄位旁邊的方塊，以使尚未連結特定提供者的所有應用程式可以呼叫新的訊息安全性提供者。提供者是成爲預設用戶端提供者、預設伺服器提供者還是兩者兼而有之，取決於爲 [提供者類型] 選取的值。
   - **提供者類型** — 選取 `client`、`server` 或 `client-server`，以確定是將提供者用做用戶端認證提供者、伺服器認證提供者還是兼用做兩者 (主從式提供者)。
   - **提供者 ID** — 輸入此提供者配置的識別碼。此名稱將顯示在 [目前的提供者配置] 清單中。
   - **類別名稱** — 輸入提供者的 Java 實作類別。用戶端認證提供者必須實作 `com.sun.xml.wss.provider.ClientSecurityAuthModule` 介面。伺服器端提供者必須實作 `com.sun.xml.wss.provider.ServerSecurityAuthModule` 介面。一個提供者可以同時實作這兩種介面，但必須實作對應於其提供者類型的介面。
9. 在 [建立提供者配置] 頁面的 [請求策略] 區段，根據需要輸入以下**選擇性**值。這些特性是選擇性的，但如果未進行指定，則不會對請求訊息套用任何認證。
   - **認證來源** — 選取 `sender`、`content` 或空值 (空白選項)，以定義對訊息層寄件者認證的需求 (例如使用者名稱密碼)、內容認證 (例如數位簽名) 或不對請求訊息進行任何認證。如果指定爲空值，則不需要進行請求的來源認證。

- **認證收信人** — 選取 beforeContent 或 afterContent，以定義寄件者對請求訊息的收件者之訊息層認證的需求 (例如，透過 XML 加密)。如果不指定該值，則其爲預設值 afterContent。

如需有關以下訊息保護策略所導致的由 SOAP 訊息安全性提供者執行的動作之描述，請參閱 「請求策略配置和回應策略配置的動作」。

10. 在 [建立提供者配置] 頁面的 [回應策略] 區段，根據需要輸入以下**選擇性**特性。這些特性是選擇性的，但如果未進行指定，則不會對回應訊息套用任何認證。

    - **認證來源** — 選取 sender、content 或空值 (空白選項)，以定義要套用到回應訊息的訊息層寄件者認證 (例如使用者名稱密碼) 或內容認證 (例如數位簽名) 之需求。如果指定爲空值，則不需要進行回應的來源認證。
    - **認證收信人** — 選取 beforeContent 或 afterContent，以定義寄件者對回應訊息的收件者之訊息層認證的需求 (例如，透過 XML 加密)。如果不指定該值，則其爲預設值 afterContent。

    如需有關以下訊息保護策略所導致的由 SOAP 訊息安全性提供者執行的動作之描述，請參閱 「請求策略配置和回應策略配置的動作」。

11. 透過按一下 [增加特性] 按鈕增加其他特性。Application Server 隨附的提供者可支援下列特性。如果使用的是其他提供者，請參閱其他提供者的文件，以獲得有關特性和有效值的更多資訊。

    - server.config — 包含伺服器配置資訊的 XML 檔案之目錄和檔案名稱。例如，*install_dir*/domains/*domain_dir*/config/wss-server-config.xml。

12. 按一下 [確定] 儲存此配置，或按一下 [取消] 結束而不進行儲存。

等效的 asadmin 指令爲：create-message-security-provider

請求策略配置和回應策略配置的動作

表 15-1 顯示了訊息保護策略配置以及由該配置的 WS-Security SOAP 訊息安全性提供者所執行的最終訊息安全性作業。

表 15-1 訊息保護 策略與 WS-Security SOAP 訊息安全性作業對映

訊息保護策略	最終的 WS-Security SOAP 訊息保護作業
auth-source="sender"	此訊息包含 wsse:Security 標頭，此標頭包含 wsse:UsernameToken (帶有密碼)。
auth-source="content"	對 SOAP 訊息內文的內容進行簽名。此訊息包含 wsse:Security 標頭，此標頭包含表示為 ds:Signature 的訊息內文簽名。

表 15-1 訊息保護 (續) 策略與 WS-Security SOAP 訊息安全性作業對映

訊息保護策略	最終的 WS-Security SOAP 訊息保護作業
auth-source="sender" auth-recipient="before-content" OR auth-recipient="after-content"	SOAP 訊息內文的內容已進行加密，並由最終的 xend:EncryptedData 替代。此訊息包含 wsse:Security 標頭，此標頭包含 wsse:UsernameToken （帶有密碼） 和 xenc:EncryptedKey。xenc:EncryptedKey 包含用於加密 SOAP 訊息內文的金鑰。此金鑰在收信人的公開金鑰中加密。
auth-source="content" auth-recipient="before-content"	SOAP 訊息內文的內容已進行加密，並由最終的 xend:EncryptedData 替代。對 xenc:EncryptedData 進行簽名。此訊息包含 wsse:Security 標頭，此標頭包含 xenc:EncryptedKey 和 ds:Signature。xenc:EncryptedKey 包含用於加密 SOAP 訊息內文的金鑰。此金鑰在收信人的公開金鑰中加密。
auth-source="content" auth-recipient="after-content"	對 SOAP 訊息內文的內容進行簽名和加密，並由最終的 xend:EncryptedData 替代。此訊息包含 wsse:Security 標頭，此標頭包含 xenc:EncryptedKey 和 ds:Signature。xenc:EncryptedKey 包含用於加密 SOAP 訊息內文的金鑰。此金鑰在收信人的公開金鑰中加密。
auth-recipient="before-content" OR auth-recipient="after-content"	SOAP 訊息內文的內容已進行加密，並由最終的 xend:EncryptedData 替代。此訊息包含 wsse:Security 標頭，此標頭包含 xenc:EncryptedKey。xenc:EncryptedKey 包含用於加密 SOAP 訊息內文的金鑰。此金鑰在收信人的公開金鑰中加密。
未指定策略。	模組未執行任何安全性作業。

刪除訊息安全性配置

若要刪除訊息安全性配置，請執行以下步驟。

1. 在管理主控台樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 server，請選取 server-config 節點。
   b. 若要配置所有實例的預設設定，請選取 default-config 節點。
3. 展開 [安全性] 節點。
4. 選取 [訊息安全性] 節點。
5. 按一下要刪除的 [訊息安全性配置] 左側的核取方塊。
6. 按一下 [刪除]。

刪除訊息安全性提供者

若要刪除訊息安全性提供者，請執行以下步驟。

1. 在管理主控台樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 展開 [安全性] 節點。
4. 展開 [訊息安全性] 節點。
5. 選取 [SOAP] 節點。
6. 選取 [提供者] 頁面。
7. 按一下要刪除的 [提供者配置] 左側的核取方塊。
8. 按一下 [刪除]。

等效的 `asadmin` 指令爲：`delete-message-security-provider`

啟用用戶端應用程式的訊息安全性

必須配置用戶端提供者的訊息保護策略，以使其等效於將與其進行互動式操作的伺服器端提供者的訊息保護策略。在安裝 Application Server 時已配置 (但未啓用) 的提供者已符合此情況。

若要啓用用戶端應用程式的訊息安全性，請修改應用程式用戶端容器特定於 Sun Java System Application Server 的配置。

若要啓用應用程式用戶端中的預設用戶端提供者，請執行以下步驟：

1. 停止所有取決於用戶端容器描述元的用戶端應用程式。
2. 在文字編輯器中開啓 Sun 應用程式用戶端容器描述元，該描述元位於 *install_dir*`/domains/`*domain_dir*`/config/sun-acc.xml`。
3. 將**粗體**文字增加到檔案中，以啓用應用程式用戶端中的預設用戶端提供者。還提供了其他代碼，以顯示啓用用戶端應用程式的訊息安全性之代碼所在的位置。非粗體代碼在安裝中可能稍有不同。請勿變更這些非粗體文字。

```
<client-container>
  <target-server name="<your_host>" address="<your_host>" port="<your_port>"/>
  <log-service file="" level="WARNING"/>
  <message-security-config auth-layer="SOAP"
    default-client-provider="ClientProvider">
    <provider-config
      class-name="com.sun.xml.wss.provider.ClientSecurityAuthModule"
      provider-id="ClientProvider" provider-type="client">
      <request-policy auth-source="sender"/>
      <response-policy/>
       <property name="security.config"
      value="C:/Sun/AppServer/lib/appclient/wss-client-config.xml"/>
    </provider-config>
  </message-security-config>
</client-container>
```

用戶端容器中配置的訊息安全性提供者還需要具有對私密金鑰和受信任證書的存取權。這將透過爲以下系統特性 (位於應用程式用戶端啓動程序檔中) 定義適當的值來完成。

-Djavax.net.ssl.keyStore

-Djavax.net.ssl.trustStore

設定應用程式用戶端配置的請求策略和回應策略

請求策略和回應策略定義與認證提供者執行的請求處理和回應處理相關聯的認證策略需求。按照訊息寄件者的順序表示這些策略，從而使內容之後出現的加密請求意味著訊息收件者將在驗證簽名之前先要對訊息進行解密。

若要獲得訊息安全性，必須同時在伺服器和用戶端中啓用請求策略和回應策略。在用戶端和伺服器中配置策略時，請確定應用程式層級訊息連結之請求 / 回應保護的用戶端策略與伺服器策略相匹配。

若要設定應用程式用戶端配置的請求策略，請按照「啓用用戶端應用程式的訊息安全性」中所述修改應用程式用戶端容器特定於 Sun Java System Application Server 的配置。在應用程式用戶端配置檔案中，增加**粗體**文字以設定請求策略。提供的其他代碼可用做參照。非粗體代碼在安裝中可能稍有不同。請勿變更這些非粗體文字。

```
<client-container>
  <target-server name="<your_host>" address="<your_host>" port="<your_port>"/>
  <log-service file="" level="WARNING"/>
  <message-security-config auth-layer="SOAP"
    default-client-provider="ClientProvider">
    <provider-config
      class-name="com.sun.xml.wss.provider.ClientSecurityAuthModule"
```

```
        provider-id="ClientProvider" provider-type="client">
        <request-policy auth-source="sender | content"
          auth-recipient="after-content | before-content"/>
        <response-policy auth-source="sender | content"
          auth-recipient="after-content | before-content"/>
         <property name="security.config"
        value="install_dir/lib/appclient/wss-client-config.xml"/>
      </provider-config>
    </message-security-config>
  </client-container>
```

auth-source 的有效值包括 sender 和 content。auth-recipient 的有效值包括 before-content 和 after-content。「請求策略配置和回應策略配置的動作」中包含一個表，用於描述這些值的各種組合的結果。

如果不想指定請求策略或回應策略，請將該元素保留爲空，例如：

```
<response-policy/>
```

詳細資訊

- 可透過以下 URL 檢視 Java 2 Standard Edition 的安全性討論：
  http://java.sun.com/j2se/1.4.2/docs/guide/security/index.html
- 可透過以下 URL 檢視「J2EE 1.4 Tutorial」的「Security」一章：
  http://java.sun.com/j2ee/1.4/docs/tutorial/doc/index.html
- 「**管理指南**」的「配置安全性」一章。
- 「Developer's Guide」的「Securing Applications」一章。
- 可以從以下 URL 查看「Oasis Web Services：SOAP Message Security (WS-Security)」規格：
  http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf
- 可以透過以下 URL 查看「OASIS Web Services Security Username Token Profile 1.0」：
  http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf
- 可以透過以下 URL 查看「OASIS Web Services Security X.509 Certificate Token Profile 1.0」：
  http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0.pdf

- 可透過以下 URL 檢視 *XML* **簽名語法和處理**文件：
  http://www.w3.org/TR/xmldsig-core/
- 可透過以下 URL 檢視 *XML* **加密語法和處理**文件：
  http://www.w3.org/TR/xmlenc-core/

第 16 章

作業事件

透過將一個或多個步驟納入不可分的工作單元，作業事件可確保資料的完整性和一致性。本章包括下列小節：

- 關於作業事件
- 有關作業事件的管理主控台作業

關於作業事件

- 何爲作業事件？
- J2EE 技術中的作業事件

何為作業事件？

作業事件是應用程式中一系列嚴密的動作，所有動作必須成功完成，否則每個動作中的所有變更會被撤消。例如，將資金從支票帳戶轉入儲蓄帳戶是一項作業事件，步驟如下：

1. 檢查支票帳戶是否有足夠的資金來支付此轉帳操作。
2. 如果支票帳戶中有足夠的資金，則將該筆資金記入此帳戶的借方。
3. 將這些資金記入儲蓄帳戶的貸方。
4. 將此次轉帳記錄到支票帳戶記錄中。
5. 將此次轉帳記錄到儲蓄帳戶記錄中。

如果這些步驟的任何一個步驟失敗，則必須撤消在前面的步驟中所做的所有變更，而且支票帳戶和儲蓄帳戶的狀態必須與它們在作業事件開始之前的狀態相同。該事件稱爲**轉返**。如果所有步驟均成功完成，則該作業事件處於**已確定**狀態。作業事件以確定或轉返狀態結束。

J2EE 技術中的作業事件

J2EE 技術中的作業事件處理包括以下五個參與者：

- 作業事件管理員
- Application Server
- 資源管理員
- 資源介面
- 使用者應用程式。

透過實作不同的 API 和功能，每個實體均有助於提高作業事件處理的可靠性，如下所述：

- 作業事件管理員提供支援作業事件分隔、作業事件資源管理、同步化及作業事件上下文傳遞所需的服務和管理功能。
- Application Server 提供支援應用程式執行環境 (包含作業事件狀態管理) 所需的基礎架構。
- 資源管理員 (透過資源介面) 提供應用程式對資源的存取權。資源管理員參與分散式作業事件，其方法爲：執行由作業事件管理員使用的作業事件資源介面來傳遞作業事件關聯、作業事件完成以及恢復工作。關聯式資料庫伺服器便是這樣一個資源管理員。
- 資源介面是一個系統層級的軟體程式庫，應用程式伺服器或用戶端可使用該程式庫連線到資源管理員。資源介面通常專用於資源管理員。它可以作爲程式庫，在使用它的用戶端位址空間中使用。JDBC 驅動程式便是這樣一個資源介面。
- 開發用於應用程式伺服器環境的作業事件使用者應用程式使用 JNDI 來查找作業事件資料源及作業事件管理員 (可選)。應用程式可以使用企業 Bean 的宣告性作業事件屬性設定或明確的程式化作業事件分隔。

有關作業事件的管理主控台作業

Application Server 根據管理主控台中的設定來處理作業事件。

配置作業事件

本小節說明如何配置以下作業事件屬性：

- 作業事件恢復
- 作業事件逾時
- 作業事件記錄

作業事件恢復

由於伺服器當機或資源管理員當機，作業事件可能未完成。完成這些中斷的作業事件並從故障中恢復是至關重要的。Application Server 可在伺服器啓動時從這些故障中恢復並完成作業事件。

執行恢復作業時，如果無法訪問某些資源，則伺服器重新啓動作業可能被延遲，因爲伺服器正在嘗試恢復作業事件。

如果作業事件跨伺服器進行，啓動該作業事件的伺服器會聯絡其他伺服器以獲得作業事件的結果。如果無法訪問其他伺服器，則該作業事件將使用 [啓發式決策] 欄位來確定結果。

若要配置 Application Server 從作業事件中恢復的方法，請執行以下步驟：

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 選取 [作業事件服務] 節點。
4. 若要恢復未完成的作業事件，請在 [重新啓動時] 欄位中選取 [恢復]。
5. 在 [重試逾時] 欄位中，設定 Application Server 嘗試連線無法訪問的伺服器之時間 (以秒爲單位)。預設值爲 10 分鐘 (600 秒)。

6. 在 [啓發式決策] 欄位中，爲作業事件中無法訪問的伺服器設定策略。

   除非有充分的理由將 [啓發式決策] 欄位設定爲 [確定]，否則將其保留設定爲 [轉返]。確定不確定的作業事件會破壞應用程式的資料完整性。

7. 按一下 [儲存]。
8. 重新啓動 Application Server。

作業事件逾時

依預設，伺服器不會使作業事件逾時。也就是說，伺服器無限期地等待作業事件完成。如果爲作業事件設定了逾時值，而作業事件在配置的時間內未完成，則 Application Server 將轉返該作業事件。

若要設定逾時值，請執行以下步驟：

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 選取 [作業事件服務] 節點。
4. 在 [作業事件逾時] 欄位中，輸入作業事件逾時之前等待的秒數。

   作業事件逾時的預設值爲 0 秒。這將停用作業事件逾時。

5. 按一下 [儲存]。
6. 重新啓動 Application Server。

作業事件記錄

爲了保持被呼叫資源的資料完整性，同時爲了能夠從故障中恢復，作業事件記錄將記錄有關每個作業事件的資訊。作業事件記錄保留在 [作業事件記錄位置] 欄位指定的目錄之 `tx` 子目錄中。這些記錄無法進行人爲讀取。

若要設定作業事件記錄的位置，請執行以下步驟：

1. 在樹形元件中，選取 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。

b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。

3. 選取 [作業事件服務] 節點。

4. 在 [作業事件記錄位置] 欄位中輸入作業事件記錄的位置。

將建立 `tx` 子目錄，而且作業事件記錄保留在該目錄下。

預設值爲 `${com.sun.aas.instanceRoot}/logs`。
`${com.sun.aas.instanceRoot}` 變數是實例的名稱，並在您啓動一個 Application Server 實例時設定此變數。若要查看 `${com.sun.aas.instanceRoot}` 的值，請按一下 [實際值]。

5. 按一下 [儲存]。

6. 重新啓動 Application Server。

關鍵點作業可以壓縮作業事件記錄檔。關鍵點間隔是記錄中關鍵點作業之間的作業事件數目。關鍵點作業可以減小作業事件記錄檔的大小。關鍵點間隔數越大 (例如，2048)，作業事件記錄檔也越大，但關鍵點作業較少，效能可能更佳。關鍵點間隔越小 (例如，256)，記錄檔也越小，而同時由於關鍵點作業較爲頻繁，效能會略微降低。

若要設定關鍵點間隔，請執行以下步驟：

1. 在樹形元件中，選擇 [配置] 節點。

2. 選取要配置的實例：

a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。

b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。

3. 選取 [作業事件服務] 節點。

4. 在 [關鍵點間隔] 欄位中，輸入關鍵點作業之間的作業事件數目。

預設值爲 2048。

5. 按一下 [儲存]。

6. 重新啓動 Application Server。

第 17 章

配置 HTTP 服務

本章描述如何爲 Application Server 的 HTTP 服務元件配置虛擬伺服器和 HTTP 偵聽程式。

- 關於 HTTP 服務
- 有關 HTTP 服務的管理主控台作業
- 有關 HTTP 偵聽程式的管理主控台作業
- 有關虛擬伺服器的管理主控台作業

關於 HTTP 服務

- 什麼是 HTTP 服務？
- 虛擬伺服器
- HTTP 偵聽程式

什麼是 HTTP 服務？

HTTP 服務是 Application Server 的元件，提供用於部署 Web 應用程式和使 HTTP 用戶端能夠存取所部署 Web 應用程式的工具。(請參閱第 104 頁的「部署 Web 應用程式」。) 這些工具通過兩種相關物件提供，即虛擬伺服器和 HTTP 偵聽程式。

虛擬伺服器

虛擬伺服器 (有時稱爲虛擬主機) 是一種物件，允許同一實體伺服器託管多個網際網路網域名稱。同一個實體伺服器上託管的所有虛擬伺服器共用該實體伺服器的網際網路通訊協定 (IP) 位址。虛擬伺服器將某個伺服器的網域名稱 (例如 www.aaa.com) 與執行 Application Server 的特定伺服器關聯起來。

備註：請勿將網際網路網域與 Application Server 的管理網域混淆。

例如，假設您要在實體伺服器上託管以下這些網域：

```
www.aaa.com
www.bbb.com
www.ccc.com
```

同時假設 www.aaa.com、www.bbb.com 和 www.ccc.com 都分別具有與之關聯的 Web 模組 web1、web2 和 web3。

這意味著以下 URL 將全部由您的實體伺服器處理：

```
http://www.aaa.com:8080/web1
http://www.bbb.com:8080/web2
http://www.ccc.com:8080/web3
```

第一個 URL 將被對映到虛擬主機 www.aaa.com，第二個 URL 將被對映到虛擬主機 www.bbb.com，第三個 URL 將被對映到虛擬主機 www.ccc.com。

另一方面，由於未向 www.bbb.com 註冊 web3，以下 URL 將導致 404 回覆碼：

```
http://www.bbb.com:8080/web3
```

若要使此對映有效，請確定 www.aaa.com、www.bbb.com 和 www.ccc.com 均可解析爲實體伺服器的 IP 位址。需要向您的網路的 DNS 伺服器註冊這些網域名稱。此外，在 UNIX 系統上，應將這些網域新增到 /etc/hosts 檔案中 (如果 /etc/nsswitch.conf 檔案中的 hosts 設定包括 files)。

啓動 Application Server 時，將自動啓動以下虛擬伺服器：

- 名爲 server 的虛擬伺服器，用於託管所有使用者定義的 Web 模組
- 名爲 __asadmin 的虛擬伺服器，用於託管所有與管理相關的 Web 模組 (特別是管理主控台)。該伺服器是一個受限制的伺服器，您不能將 Web 模組部署到該虛擬伺服器上。

如果是在非生產環境中開發、測試和部署 Web 服務，通常您只需要使用 server 虛擬伺服器。在生產環境中，其他虛擬伺服器可以爲使用者和客戶提供託管工具，這樣，儘管只有一個實體伺服器，但每個使用者和客戶好像都有自己的 Web 伺服器。

HTTP 偵聽程式

每個虛擬伺服器都通過一個或多個 HTTP 偵聽程式來提供伺服器與用戶端之間的連線。每個 HTTP 偵聽程式都是具有 IP 位址、連接埠號、伺服器名稱以及預設虛擬伺服器的偵聽插槽。

HTTP 偵聽程式必須具備唯一的連接埠號和 IP 位址組合。例如，通過將 IP 位址指定爲 0.0.0.0，HTTP 偵聽程式可以在機器的給定連接埠上偵聽所有的已配置 IP 位址。或者，HTTP 偵聽程式可以爲每個偵聽程式指定唯一的 IP 位址，但使用相同的連接埠。

由於 HTTP 偵聽程式是 IP 位址和連接埠號的組合，因此您可以擁有多個 IP 位址相同但連接埠號不同 (如 1.1.1.1:8081 和 1.1.1.1:8082) 的 HTTP 偵聽程式，或 IP 位址不同但連接埠號相同 (如 1.1.1.1:8081 和 1.2.3.4:8081) 的 HTTP 偵聽程式 (如果已將機器配置爲可以回應這些位址)。

不過，如果 HTTP 偵聽程式使用 0.0.0.0 IP 位址 (在一個連接埠上的所有 IP 位址上偵聽)，您便無法建立其他 IP 位址的 HTTP 偵聽程式 (在用於特定 IP 位址的同一個連接埠上偵聽)。例如，如果 HTTP 偵聽程式使用 0.0.0.0:8080 (連接埠 8080 上的所有 IP 位址)，則其他 HTTP 偵聽程式不能使用 1.2.3.4:8080。

由於執行 Application Server 的系統通常只能存取一個 IP 位址，因此 HTTP 偵聽程式通常使用 0.0.0.0 IP 位址和不同的連接埠號，其中每個連接埠號用於不同目的。如果系統可以存取多個 IP 位址，則每個位址可以用於不同目的。

依預設，Application Server 啓動時，它具有以下 HTTP 偵聽程式：

- 兩個分別名爲 `http-listener-1` 和 `http-listener-2` 的 HTTP 偵聽程式，這兩個偵聽程式與名爲 `server` 的虛擬伺服器相關聯。名爲 `http-listener-1` 的偵聽程式未啓用安全性；而名爲 `http-listener-2` 的偵聽程式啓用了安全性。
- 名爲 `admin-listener` 的 HTTP 偵聽程式，該偵聽程式與名爲 `__asadmin` 的虛擬伺服器相關聯。該偵聽程式未啓用安全性。

所有這些偵聽程式均使用 IP 位址 0.0.0.0 和在安裝 Application Server 程序中指定爲 HTTP 伺服器連接埠號的連接埠號。如果 Application Server 使用預設連接埠號值，則 `http-listener-1` 使用連接埠 8080，`http-listener-2` 使用連接埠 8181，`admin-listener` 使用連接埠 4849。

每個 HTTP 偵聽程式均有一個預設虛擬伺服器。當請求 URL 的主機部分與 HTTP 偵聽程式關聯的任何虛擬伺服器 (在虛擬伺服器的 `http-listeners` 屬性中列示 HTTP 偵聽程式，即可將虛擬伺服器與該 HTTP 偵聽程式關聯起來) 均不匹配時，HTTP 偵聽程式會將請求 URL 路由到預設虛擬伺服器。

此外，還應在 HTTP 偵聽程式中指定接收器執行緒的數目。接收器執行緒就是等待連線的執行緒。執行緒接受連線並將其放入佇列 (稱爲連線佇列) 中，在佇列中將由工作者執行緒接受這些連線。配置足夠多的接收器執行緒，以便在發生新的請求時總有一個可用，但又需要數目相當少，以免給系統造成太重負擔。連線佇列中既包括接收器執行緒剛剛接受的新連線，又包括由持續作用連線管理子系統管理的持續性連線。

一組請求處理執行緒將從連線佇列中擷取進來的 HTTP 請求並處理這些請求。這些執行緒將剖析 HTTP 標頭、選擇適當的虛擬伺服器並通過請求處理引擎處理請求。如果沒有更多要處理的請求，但連線可以保持持續性 (透過使用 HTTP/1.1 或傳送 `Connection:keep-alive` 標頭)，請求處理執行緒將假定連線處於閒置狀態，並將連線傳送給持續作用連線管理子系統。

持續作用子系統將定期輪詢此類閒置連線，並在連線佇列中對那些活動的連線進行排隊，以便將來進行處理。請求處理執行緒將再次從連線佇列中擷取連線並處理其請求。持續作用子系統是多執行緒的，可以管理大約數萬個連線。透過將大量連線分成較小的子集，使用有效的輪詢技術來確定哪些連線已就緒並具有請求，以及哪些連線由於處於閒置狀態的時間較長而被視爲已關閉 (超過允許的持續作用逾時的最大值)。

HTTP 偵聽程式的伺服器名稱即爲在重新導向程序中由伺服器傳送給用戶端的 URL 中出現的主機名稱。此屬性會影響伺服器自動產生的 URL；但不會影響儲存在伺服器中目錄和檔案的 URL。如果伺服器使用一個別名，則該名稱應爲此別名。如果用戶端傳送了一個 `Host:` 標頭，則在重新導向中該主機名稱將取代 HTTP 偵聽程式的伺服器名稱。

要使用不同於原始請求中指定的連接埠號的連接埠號，請指定重新導向連接埠。如果出現以下某一種情況，則會發生**重新導向**：

- 如果用戶端嘗試存取已不存在於指定 URL 處的資源 (即該資源已移至其他位置)，伺服器將傳回一個指定的回應碼，並在回應的位置標頭中包含新的位置，從而將用戶端重新導向到新位置 (而不是傳回 404)。
- 如果用戶端嘗試通過常規 HTTP 連接埠存取受保護 (例如 SSL) 的資源，則伺服器會將此請求重新導向到啓用了 SSL 的連接埠上。在此情況下，伺服器將在位置回應標頭中傳回一個新的 URL，其中的原始非安全連接埠已被替代爲啓用了 SSL 的連接埠。用戶端隨後將連線到這個新的 URL。

此外，還應指定是否爲 HTTP 偵聽程式啓用安全性以及使用哪種類型的安全性 (如使用哪一個 SSL 協定以及哪些密碼)。

若要存取部署在 Application Server 上的 Web 應用程式，請使用 URL `http://localhost:8080/` (或者，如果是安全應用程式，則使用 `http://localhost:8181/`) 和爲此 Web 應用程式指定的環境根。若要存取管理主控台，請使用 URL `https://localhost:4849/` 或 `https://localhost:4849/asadmin/` (其預設環境根)。

由於虛擬伺服器必須指定一個現有的 HTTP 偵聽程式，並且不能指定其他虛擬伺服器已使用的 HTTP 偵聽程式，因此在建立新的虛擬伺服器之前，應至少建立一個 HTTP 偵聽程式。

有關 HTTP 服務的管理主控台作業

- 配置 HTTP 服務
- 配置 HTTP 服務存取記錄

配置 HTTP 服務

若要配置 HTTP 服務，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 選取 [HTTP 服務] 節點。
4. 在 [HTTP 服務] 頁面中，您可以設定套用到該服務的所有 HTTP 偵聽程式的特性。

   下表列示了這些特性。

表 17-1 HTTP 服務特性

特性名稱	描述	預設值
`traceEnabled`	如果設定為 true，則啟用 TRACE 作業。將此特性設定為 false 將使 Application Server 較少地受到跨站台程序檔攻擊的影響。	FALSE

表 17-1 HTTP 服務特性 (續)

特性名稱	描述	預設值
monitoringCacheEnabled	如果設定為 true，Application Server 將快取 HTTP 服務的統計資訊的本地值，以回應統計資訊查詢。此值可用於提高效能。 如果設定為 false，Application Server 將查詢 HTTP 服務的每個統計資訊值。	TRUE
monitoringCacheRefreshInMillis	指定更新監視快取之前的間隔時間 (以毫秒為單位)。	5000
sslCacheEntries	指定可以快取的 SSL 階段作業的數目。其值沒有上限。	10000
sslSessionTimeout	指定 SSL2 階段作業逾時之前的秒數。	100
ssl3SessionTimeout	指定 SSL3 階段作業逾時之前的秒數。	86400
sslClientAuthDataLimit	指定用戶端證書交握階段緩衝的應用程式資料的最大數量 (以位元組為單位)。	1048576
sslClientAuthTimeout	指定用戶端證書交握階段逾時之前的秒數。	60
keepAliveQueryMeanTime	指定所需的持續作用延時 (以毫秒為單位)。	100
keepAliveQueryMaxSleepTime	指定輪詢持續作用連線的進一步請求之後休眠時間的上限 (以毫秒為單位)。	100
stackSize	指定原生執行緒的最大堆疊大小。	隨作業系統/機器而定
statsProfilingEnabled	如果設定為 false，將停用 HTTP 服務的監視統計資訊的記錄 (可以提高效能)。如果將此特性設定為 false，則啟用 HTTP 服務的監視功能不會有任何作用。	TRUE
chunkedRequestBufferSize	指定無塊請求資料的預設緩衝區大小 (以位元組為單位)。	8192
chunkedRequestTimeoutSeconds	指定無塊請求資料的預設逾時 (以秒為單位)。	60
dnsCacheEnabled	如果設定為 true，則允許使用者監視與 DNS 快取相關的統計資訊。該特性僅在選取了 [HTTP 通訊協定] 標籤中的 [DNS 查找] 方塊之後才有效。否則，將忽略該特性設定。	FALSE

5. 按一下 [存取記錄] 標籤以配置存取記錄自動重建。按一下其他標籤以配置請求處理、持續作用子系統、連線池、HTTP 通訊協定和 HTTP 檔案快取。
6. 按一下 [儲存]。

配置 HTTP 服務存取記錄

使用此頁面可以爲虛擬伺服器啓用和配置存取記錄自動重建。這些記錄位於 *domain_root_dir*/*domain_dir*/logs/access 目錄中，其命名如下：

*virtual_server_name*_access_log.*yyyy-mm-dd*.txt

按一下 [載入預設值] 可以載入預設值。若要變更這些記錄的自動重建特性，請執行以下操作：

- 核取 [檔案自動重建] 方塊以啓用檔案自動重建。依預設，將啓用檔案自動重建。
- 從 [自動重建策略] 下拉式清單中選擇策略。(唯一可用的策略爲 time。)
- 在 [自動重建間隔] 欄位中，鍵入數值以指定存取記錄自動重建之間的分鐘數。僅當自動重建策略爲 time 時，此欄位才有效。預設值爲 1440 分鐘。
- 在 [自動重建後綴] 欄位中，鍵入字串值以指定自動重建後新增到記錄檔名稱中的後綴。預設值爲 %YYYY;%MM;%DD;-%hh;h%mm;m%ss;s。
- 在 [格式] 欄位中，輸入字串值以指定存取記錄的格式。請使用下表中顯示的格式。預設格式爲 %client.name% %auth-user-name% %datetime% %request% %status% %response.length%。

表 17-2 存取記錄格式的記號值

資料	記號
用戶端主機名稱	%client.name%
用戶端 DNS	%client.dns%
系統日期	%datetime%
完整 HTTP 請求行	%request%
狀態	%status%
回應特性長度	%response.length%
參考標頭	%header.referer%
使用者代理程式	%header.user-agent%
HTTP 方法	%http-method%
HTTP URI	%http-uri%
HTTP 查詢字串	%query-str%
HTTP 通訊協定版本	%http-version%
接受標頭	%header.accept%
日期標頭	%header.date%

表 17-2 存取記錄格式的記號值 (續)

資料	記號
If-Modified-Since 標頭	%header.if-mod-since%
授權標頭	%header.auth%
按照 RFC 2616 定義的任何有效 HTTP 標頭值 (any 也是有效的標頭值；此處將其指定為變數)	%header.*any*%
授權使用者的名稱	%auth-user-name%
Cookie 的值	%cookie.value%
虛擬伺服器 ID	%vs.id%

- 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以返回預設設定。

配置 HTTP 服務請求處理執行緒

使用此頁面可以爲 HTTP 服務配置請求處理執行緒：

- 按一下 [載入預設值] 可以載入預設值。
- 在 [執行緒計數] 欄位中，鍵入數值以指定請求處理執行緒的最大數目。預設值爲 128。
- 在 [初始執行緒計數] 欄位中，鍵入伺服器啓動時可用的請求處理執行緒的數目。預設值爲 48。
- 在 [執行緒遞增量] 欄位中，鍵入請求數目超過初始執行緒計數時要新增的請求處理執行緒的數目。預設值爲 10。
- 在 [請求逾時] 欄位中，鍵入請求逾時之前的秒數。其預設值爲 30 秒。
- 在 [緩衝區長度] 欄位中，鍵入請求處理執行緒用於讀取請求資料的緩衝區的大小 (以位元組爲單位)。預設值爲 4096 位元組。
- 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以返回預設設定。

配置 HTTP 服務的持續作用子系統

使用此頁面可以配置 HTTP 服務的持續作用子系統。

- 按一下 [載入預設值] 可以載入預設值。

- 在 [執行緒計數] 欄位中，鍵入要使用的持續作用執行緒的數目。預設值爲 1。
- 在 [最大連線數] 欄位中，鍵入要維護的持續性連線的最大數目。預設值爲 256。
- 在 [逾時] 欄位中，鍵入持續作用連線應保持開啓狀態的最大秒數。其預設值爲 30 秒。
- 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以返回預設設定。

另請參閱：

- HTTP 偵聽程式

配置 HTTP 服務連線池

使用此頁面可以爲 HTTP 服務連線佇列配置連線池：

- 按一下 [載入預設值] 可以載入預設值。
- 在 [最大擱置計數] 欄位中，鍵入 HTTP 偵聽程式允許的最大擱置連線數目。預設值爲 4096。
- 在 [佇列大小] 欄位中，鍵入連線佇列大小的最大值 (以位元組爲單位)。此值還指定伺服器可以擁有的最大未完成連線數目。預設值爲 4096。
- 在 [接收緩衝區大小] 欄位中，爲 HTTP 偵聽程式鍵入接收緩衝區的大小。預設值爲 4096。
- 在 [傳送緩衝區大小] 欄位中，爲 HTTP 偵聽程式鍵入傳送緩衝區的大小。預設值爲 8192。
- 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以返回預設設定。

為 HTTP 服務配置 HTTP 通訊協定

使用此頁面可以爲 HTTP 服務配置 HTTP 通訊協定：

- 按一下 [載入預設值] 可以載入預設值。
- 在 [版本] 欄位中，鍵入要使用的 HTTP 通訊協定的版本 (`HTTP/1.0` 或 `HTTP/1.1`)。預設值爲 `HTTP/1.1`。
- 選取 [DNS 查找] 方塊以啓用用戶端 DNS 項目的查找。預設值爲 false。
- 取消選取 [SSL] 方塊可以全域停用伺服器中的安全性。將此值設定爲 true 可以將 SSL 用於任何啓用了安全性的偵聽程式。預設值爲 true。

- 在 [已強制回應類型] 欄位中，鍵入沒有與副檔名匹配的可用 MIME 對應時使用的回應類型。預設值爲 `text/html; charset=iso-8859-1`。
- 在 [預設回應類型] 欄位中，鍵入預設回應類型。預設值爲 `text/html; charset=iso-8859-1`。此值是以分號分隔的字串，由特性類型、編碼、語言和字元集組成。
- 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以返回到預設設定。

為 HTTP 服務配置 HTTP 檔案快取

使用此頁面可以爲 HTTP 服務配置 HTTP 檔案快取。

檔案快取儲存靜態特性，以便伺服器可以快速處理此類特性的請求。

- 按一下 [載入預設值] 可以載入預設值。
- 核取 [全域] 方塊可以啓用檔案快取。預設值爲 true。
- 核取 [檔案傳輸] 方塊可以在 Windows 中啓用 `TransmitFileSystem` 方法。預設值爲 false。
- 在 [最長存在時間] 欄位中，鍵入有效快取項目的最長存在時間 (以秒爲單位)。其預設值爲 30 秒。
- 在 [最大檔案計數] 欄位中，鍵入檔案快取中的最大檔案數目。預設值爲 1024。
- 在 [Hash 初始大小] 欄位中，鍵入 hash 儲存區的初始數目。預設值爲零。
- 在 [中型檔案大小限制] 欄位中，鍵入可快取爲記憶體對映檔案的檔案大小的最大值 (以位元組爲單位)。預設值爲 537,600 位元組。
- 在 [中型檔案大小] 欄位中，鍵入快取爲記憶體對映檔案的所有檔案大小的總值 (以位元組爲單位)。預設值爲 10,485,760 位元組。
- 在 [小型檔案大小限制] 欄位中，鍵入可讀入記憶體的檔案大小的最大值 (以位元組爲單位)。預設值爲 2048 位元組。
- 在 [小型檔案大小] 欄位中，鍵入讀入記憶體的所有檔案大小的總值 (以位元組爲單位)。預設值爲 1,048,576 位元組。
- 從 [已啓用檔案快取] 下拉式清單中選擇 `ON` 或 `OFF`，以在檔案大小小於中型檔案大小限制時啓用或停用檔案特性的快取。預設值爲 `ON`。
- 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以返回到預設設定。

有關虛擬伺服器的管理主控台作業

- 建立虛擬伺服器
- 編輯虛擬伺服器
- 刪除虛擬伺服器

建立虛擬伺服器

若要建立虛擬伺服器，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 展開 [HTTP 服務] 節點。
4. 選取 [虛擬伺服器] 節點。
5. 在 [虛擬伺服器] 頁面中，按一下 [新建]。將顯示 [建立虛擬伺服器] 頁面。
6. 在 [ID] 欄位中，爲虛擬伺服器鍵入唯一的名稱。此值用於內部識別虛擬伺服器，而不會提供給 HTTP 用戶端。提供給 HTTP 用戶端的主機名稱必須在 [主機] 欄位中指定。
7. 在 [主機] 欄位中，鍵入主機名稱或在執行伺服器的電腦名稱。請使用已向網路的 DNS 伺服器註冊 (對於 UNIX 系統，則是已在 `/etc/hosts` 檔案中註冊) 的實際主機名稱或虛擬主機名稱。
8. 在相對的 [狀態] 區域中，選取 [開啓]、[關閉] 或 [停用]。預設值爲 [開啓]。
9. 將 [HTTP 偵聽程式] 欄位保留爲空白。在建立 HTTP 偵聽程式並將其與此伺服器關聯時，系統將自動填寫此欄位。

   (使用此欄位時，需要指定現有的 HTTP 偵聽程式。但是，您不能指定由其他虛擬伺服器使用的偵聽程式；否則，伺服器記錄中將顯示錯誤。由於偵聽程式在建立時必須與一個現有的虛擬伺服器相關聯，因此所有的現有偵聽程式均已被其他虛擬伺服器使用。)

10. 在 [預設 Web 模組] 下拉式清單中，選擇已部署的 Web 模組 (如果有)，此模組將回應所有無法對映到已部署至虛擬伺服器的其他 Web 模組的請求。

如果不指定預設 Web 模組，系統將使用具有空環境根的 Web 模組。如果沒有環境根爲空的 Web 模組，將建立和使用系統預設 Web 模組。

11. 在 [記錄檔] 欄位中，鍵入將記錄此虛擬伺服器的記錄訊息的檔案的路徑名稱。如果要將記錄訊息傳送到預設伺服器記錄 *domain_root_dir*/*domain_dir*/logs/server.log，請將此欄位保留爲空。

12. 在 [附加特性] 區域中，按一下 [增加特性] 以便爲虛擬伺服器增加特性。無論是否指定特性，新的伺服器都會將預設特性 docroot 和 accesslog 設定爲預設值。

13. 按一下 [確定] 以儲存該虛擬伺服器。

下表列示了可用的特性。

表 17-3 虛擬伺服器特性

特性名稱	描述
docroot	伺服器的文件根目錄的絕對路徑。 預設值為 *domain_root_dir*/*domain_dir*/docroot。
accesslog	伺服器存取記錄的絕對路徑。 預設值為 *domain_root_dir*/*domain_dir*/logs/access。
sso-enabled	如果設置為 false，則此虛擬伺服器將停用單次登入，使用者必須在使用虛擬伺服器上的每個應用程式時都分別進行驗證。 Servlet 和 JSP 頁面支援在 Application Server 上跨應用程式的單次登入。此功能允許需要同一使用者登入資訊的多個應用程式共用登入資訊，使用者不必在使用每個應用程式時都分別進行登入。 預設值為 true。
sso-max-inactive-seconds	指定如果未接收到任何用戶端活動，在清除使用者的單次登入記錄前等待的時間 (以秒為單位)。由於單次登入套用到同一個虛擬伺服器上的多個應用程式，因此對其中任何一個應用程式的存取都可以使單次登入記錄保持使用中狀態。 預設值為 300 秒 (5 分鐘)。較高的值為使用者提供了較長的單次登入持續性，但會佔用伺服器的更多記憶體。
sso-reap-interval-seconds	指定清除過期的單次登入記錄的時間間隔 (以秒為單位)。 預設值為 60。

表 17-3 虛擬伺服器特性 (續)

特性名稱	描述

allowLinking	如果設置為 true，則將為部署到該虛擬伺服器上的所有 Web 應用程式提供符號連結資源。透過使用 sun-web.xml 檔案中的 sun-web-app 特性 allowLinking，各個 Web 應用程式可以置換該設定。
	`<sun-web-app>`
	`  <property name="allowLinking" value="{true|false}"/>`
	`</sun-web-app>`
	預設值為 true。

等效的 asadmin 指令爲：create-virtual-server

編輯虛擬伺服器

若要編輯虛擬伺服器，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 server，請選取 server-config 節點。
   b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 default-config 節點。
3. 展開 [HTTP 服務] 節點。
4. 選取 [虛擬伺服器] 節點。
5. 選擇要編輯的虛擬伺服器。
6. 在 [編輯虛擬伺服器] 頁面中，您可以執行以下作業：
   - 在 [主機] 欄位中變更主機名稱。
   - 變更 [狀態] 設定的值。
   - 新增或移除 HTTP 偵聽程式。
   - 變更 [預設 Web 模組] 的選擇。
   - 變更 [記錄檔] 的值。
   - 新增、移除或修改特性。

7. 按一下 [儲存] 以儲存變更。

刪除虛擬伺服器

若要刪除虛擬伺服器，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 展開 [HTTP 服務] 節點。
4. 選取 [虛擬伺服器] 節點。
5. 在 [虛擬伺服器] 頁面中，核取要刪除的虛擬伺服器名稱旁邊的方塊。
6. 按一下 [刪除]。

可以刪除 `__asadmin` 虛擬伺服器，但是建議不要這樣做。如果打算這樣做，請先將 Application Server 的 `domain.xml` 檔案中的 `virtual-server` 元素複製到安全的位置，以便可以在需要時復原這些設定。

等效的 `asadmin` 指令爲：`delete-virtual-server`

有關 HTTP 偵聽程式的管理主控台作業

- 建立 HTTP 偵聽程式
- 編輯 HTTP 偵聽程式
- 刪除 HTTP 偵聽程式

建立 HTTP 偵聽程式

若要建立 HTTP 偵聽程式，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 展開 [HTTP 服務] 節點。
4. 選取 [HTTP 偵聽程式] 節點。
5. 在 [HTTP 偵聽程式] 頁面中，按一下 [新建]。將顯示 [建立 HTTP 偵聽程式] 頁面。
6. 在 [名稱] 欄位中，鍵入偵聽程式的名稱。
7. 如果您不希望在伺服器重新啓動時啓用偵聽程式，請在 [偵聽程式] 欄位中取消核取 [已啓用] 方塊。
8. 在 [網路位址] 欄位中，如果希望偵聽程式偵聽此伺服器的所有 IP 位址，請鍵入 0.0.0.0 並使用唯一的連接埠值。否則，請鍵入此伺服器的有效 IP 位址。
9. 在 [偵聽程式連接埠] 欄位中，如果 [網路位址] 欄位爲 0.0.0.0，請鍵入唯一的連接埠值，或者，如果要使用其他 IP 位址，請鍵入所需的連接埠值。
10. 從 [預設虛擬伺服器] 下拉式清單中選擇一個虛擬伺服器。
11. 在 [伺服器名稱] 欄位中，鍵入伺服器傳送給用戶端的 URL 中使用的主機名稱。如果您的伺服器使用一個別名，則該名稱應爲此別名。如果伺服器未使用別名，請將此欄位保留爲空。

12. 在 [進階] 區域中，可以執行以下作業：

    - 若要將請求重新導向到其他連接埠，請在 [重新導向連接埠] 欄位中鍵入一個值。如果滿足以下兩個條件，Application Server 將自動重新導向請求：

      - 此偵聽程式支援非 SSL 請求。
      - 接收了匹配安全性限制需要 SSL 傳輸的請求。

      依預設，Application Server 使用原始請求中指定的連接埠號。

    - 變更接收器執行緒的數目。

    - 取消核取 [Powered By] 方塊以在 Servlet 產生的 HTTP 回應標頭中停用 `X-Powered-By:Servlet/2.4` 標頭。

      Java Servlet 2.4 規格中定義了此標頭，其容器可以新增到 Servlet 產生的回應。類似地，JavaServer Pagesô (JSPô) 2.0 規格中定義了一個 `X-Powered-By:JSP/2.0` 標頭，此標頭將新增到使用 JSP 技術的回應 (選擇性)。依預設，將爲 Web 應用程式包含 `X-Powered-By:JSP/2.0` 標頭。這些標頭的目標是幫助網站管理員收集關於使用 Servlet 和 JSP 技術方面的統計資料。

      如需有關啓用和停用 JSP 頁面的 `X-Powered-By` 標頭的資訊，請參閱「Application Server Developer's Guide」中的「Deployment Descriptor Files」一章。請參閱第 48 頁的「更多資訊」以獲得指向此文件的連結。

      根據生產環境的不同，可以省略 `X-Powered-By` 標頭的產生，以隱藏基礎技術。

13. 若要建立不安全的偵聽程式，請按一下 [確定]。

在此頁面的 [SSL] 區段，您可以將偵聽程式配置爲使用 SSL 安全性、TLS 安全性或同時使用 SSL 和 TLS 安全性。

若要設定安全偵聽程式，請執行以下操作：

1. 在 [安全性] 欄位中核取 [啓用] 方塊。
2. 若要強制用戶端在使用此偵聽程式時自行向伺服器進行驗證，請在 [用戶端驗證] 欄位中核取 [啓用] 方塊。
3. 在 [證書暱稱] 欄位中輸入現有伺服器金鑰組和證書的名稱。請參閱「安全性」一章以取得更多資訊。
4. 在 [SSL3/TLS] 區段：

   a. 選取要在偵聽程式上啓用的安全性協定。選取 [SSL3]、[TLS] 或兩者。

b. 選取協定所使用的密碼組。若要啓用所有密碼組，請核取 [所有支援的密碼組]。您還可以啓用單一密碼組。

此時，偵聽程式將在已指定爲預設虛擬伺服器的虛擬伺服器的 [HTTP 偵聽程式] 欄位中列示。

等效的 asadmin 指令爲：create-http-listener, create-ssl

編輯 HTTP 偵聽程式

若要編輯 HTTP 偵聽程式，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 server，請選取 server-config 節點。
   b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 default-config 節點。
3. 展開 [HTTP 服務] 節點。
4. 選取 [HTTP 偵聽程式] 節點。
5. 選取要編輯的 HTTP 偵聽程式。
6. 在 [編輯 HTTP 偵聽程式] 頁面中，修改其中的設定。
7. 按一下 [儲存] 以儲存變更。

刪除 HTTP 偵聽程式

若要刪除 HTTP 偵聽程式，請執行以下步驟：

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 server，請選取 server-config 節點。
   b. 若要爲將來的實例 (實例使用 default-config 的副本) 配置預設設定，請選取 default-config 節點。
3. 展開 [HTTP 服務] 節點。

4. 選取 [HTTP 偵聽程式] 節點。
5. 在 [HTTP 偵聽程式] 頁面中，核取要刪除的 HTTP 偵聽程式名稱旁邊的方塊。
6. 按一下 [刪除]。

可以刪除 `http-listener-1`、`http-listener-2` 和 `admin-listener` HTTP 偵聽程式，但是建議不要這樣做。如果打算這樣做，請先將 Application Server 的 `domain.xml` 檔案中的 `http-listener` 元素複製到安全的位置，以便可以在需要時復原這些設定。

等效的 `asadmin` 指令爲：`delete-http-listener`

第 18 章

配置物件請求代理程式

本章描述如何配置物件請求代理程式 (ORB) 和 IIOP 偵聽程式。它包含以下小節：

- 關於物件請求代理程式
- 有關 ORB 的管理主控台作業
- 有關 IIOP 偵聽程式的管理主控台作業

關於物件請求代理程式

- CORBA
- 什麼是 ORB？
- IIOP 偵聽程式

CORBA

Application Server 支援標準的協定和格式集來確定互通的功能。這些協定之間的協定是由 CORBA 定義的。

CORBA (共用物件請求代理程式架構) 模型以請求分散式物件服務或伺服器服務的用戶端爲基礎，透過明確定義的介面，以遠端方法請求形式發送物件請求。遠端方法請求傳送有關需要執行的作業的資訊，其中包括被呼叫方法的服務供應商的物件名稱 (稱爲物件參考) 和參數 (如果有)。CORBA 自動處理物件註冊、物件位置、物件啓動、請求非多工、錯誤處理、排列與作業派送等網路程式設計作業。

什麼是 ORB？

物件請求代理程式 (ORB) 是 CORBA 的中央元件。ORB 提供所需的基礎架構來識別並尋找物件、處理連線管理、傳送資料並請求通訊。

CORBA 物件之間從不直接進行通訊，該物件是透過遠端存根向在本機機器中執行的 ORB 發出請求。然後，本機 ORB 將請求發送至使用網際網路 Orb 交換協定 (縮寫爲 IIOP) 的另一台機器中的 ORB。然後，遠端 ORB 找到適當的物件、處理請求並傳回結果。

使用 RMI-IIOP，應用程式或物件可將 IIOP 用作遠端方法呼叫 (RMI) 協定。企業 Bean (EJB 模組) 的遠端用戶端通過 RMI-IIOP 與 Application Server 進行通訊。

IIOP 偵聽程式

IIOP 偵聽程式是一個偵聽插槽，它接受來自企業 Bean 的遠端用戶端和其他基於 CORBA 的用戶端的進來的連線。可以爲 Application Server 配置多個 IIOP 偵聽程式。爲每個偵聽程式指定一個連接埠號、一個網路位址和 (選擇性地) 多個安全性屬性。如需更多資訊，請參閱第 293 頁的「建立 IIOP 偵聽程式」。

有關 ORB 的管理主控台作業

- 配置 ORB

配置 ORB

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 選取 [ORB] 節點。

4. 從 [執行緒池 ID] 下拉式清單中選擇 ORB 使用的執行緒池。

   ORB 使用執行緒池回應來自通過 RMI-IIOP 進行通訊的企業 Bean 的遠端用戶端和其他用戶端的請求。如需更多資訊，請參閱第 297 頁的「Application Server 中的執行緒池」和第 298 頁的「建立執行緒池」。

5. 在 [最大訊息片段大小] 欄位中，設定 IIOP 訊息的最大片段大小。

   大於此大小的訊息將被分段。

6. 在 [連線總數] 欄位中，設定所有 IIOP 偵聽程式的最大進來的連線數。
7. 如果需要進行 IIOP 用戶端認證，請選取 [需要] 核取方塊。
8. 按一下 [儲存] 以儲存變更，或者按一下 [載入預設值] 以載入預設值。
9. 重新啓動伺服器。

有關 IIOP 偵聽程式的管理主控台作業

- 建立 IIOP 偵聽程式
- 編輯 IIOP 偵聽程式
- 刪除 IIOP 偵聽程式

建立 IIOP 偵聽程式

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 展開 [ORB] 節點。
4. 選取 [IIOP 偵聽程式]。
5. 按一下 [新建]。
6. 在 [名稱] 欄位中輸入用於識別偵聽程式的名稱。

7. 在 [網路位址] 欄位中輸入偵聽程式的網路位址。

   此位址可以是 IP 位址，也可以是 DNS 可解析的主機名稱。

8. 在 [偵聽程式連接埠] 欄位中，輸入偵聽程式要進行偵聽的連接埠號。
9. 在 [偵聽程式] 欄位中，核取 [啓用] 方塊以啓用偵聽程式。
10. 在 [附加特性] 區域中，爲應用程式所需的特性提供值。
11. 若要建立不安全的偵聽程式，請按一下 [確定]。

在此頁面的 [安全性] 區段中，您可以將偵聽程式配置爲使用 SSL 安全性、TLS 安全性或同時使用 SSL 和 TLS 安全性。

若要設定安全偵聽程式，請執行以下操作：

1. 在 [安全性] 欄位中核取 [啓用] 方塊。
2. 若要強制用戶端在使用此偵聽程式時自行向伺服器進行驗證，請在 [用戶端驗證] 欄位中核取 [啓用] 方塊。
3. 在 [證書暱稱] 欄位中輸入現有伺服器金鑰組和證書的名稱。
4. 在 [SSL3/TLS] 區段：
   a. 核取要在偵聽程式上啓用的安全性協定。核取 [SSL3] 或 [TLS]，或同時啓用這兩種協定。
   b. 選取協定所使用的密碼組。若要啓用所有密碼組，請核取 [所有支援的密碼組]。您還可以啓用單一密碼組。
5. 按一下 [確定]。

現在，該偵聽程式將列示在 [IIOP 偵聽程式] 頁面的 [目前偵聽程式] 表中。

等效的 asadmin 指令爲：`create-iiop-listener`, `create-ssl`

編輯 IIOP 偵聽程式

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。

3. 展開 [ORB] 節點。
4. 選取 [IIOP 偵聽程式] 節點。
5. 在 [目前偵聽程式] 表中選取要修改的偵聽程式。
6. 修改該偵聽程式的設定。如需有關可修改欄位的描述，請參閱第 293 頁的「建立 IIOP 偵聽程式」。
7. 如果變更了偵聽程式的連接埠號，請重新啓動伺服器。

刪除 IIOP 偵聽程式

1. 在樹形元件中，展開 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 展開 [ORB] 節點。
4. 選取 [IIOP 偵聽程式] 節點。
5. 在 [目前偵聽程式] 表中核取要刪除的偵聽程式。
6. 按一下 [刪除]。

等效的 `asadmin` 指令爲：`delete-iiop-listener`

第 19 章

執行緒池

本章描述如何建立、編輯和刪除執行緒池。它包含以下小節：

- 關於執行緒池
- 有關執行緒池的管理主控台作業

關於執行緒池

本小節描述執行緒池及其在 Application Server 中的工作方式。

Application Server 中的執行緒池

Java 虛擬機器 (JVM) 可以支援一次執行多個執行緒。爲了提昇效能，Application Server 維護了一個或多個執行緒池。可以將特定的執行緒池指定給連接器模組和 ORB。

一個執行緒池可以爲多個連接器模組和企業 Bean 提供服務。請求執行緒處理使用者對應用程式元件的請求。伺服器收到請求時，它會將該請求指定給執行緒池中的空閑執行緒。執行緒會執行用戶端請求，並傳回結果。例如，如果請求需要使用目前被佔用的系統資源，則此執行緒將等待，直至此資源可用時，才允許請求使用此資源。

指定爲來自應用程式的請求保留的最大執行緒數和最小執行緒數。可以在這兩個值之間，動態調整執行緒池。指定的最小執行緒池大小將通知伺服器爲應用程式請求至少分配該大小的保留執行緒數。該數目可以增加到所指定的最大執行緒池大小。

增加程序可用的執行緒數目，可讓程序同時回應更多的應用程式請求。

透過將 Application Server 執行緒分到不同的執行緒池中，避免在一個資源介面或應用程式佔用 Application Server 中的所有執行緒時出現執行緒不足的情況。

有關執行緒池的管理主控台作業

- 建立執行緒池
- 編輯執行緒池
- 刪除執行緒池

建立執行緒池

1. 在樹形元件中，選擇 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 選取 [執行緒池] 節點。
4. 在 [目前的池] 下按一下 [新建]。
5. 在 [執行緒池 ID] 欄位中輸入執行緒池的名稱。
6. 在 [最小執行緒池大小] 欄位中，輸入爲此佇列中的請求提供服務的執行緒池中執行緒的最小數目。

   創設此執行緒池時將預先建立這些執行緒。
7. 在 [最大執行緒池大小] 欄位中，輸入爲此佇列中的請求提供服務的執行緒池中執行緒的最大數目。

   這是存在於此執行緒池中的執行緒數上限。
8. 在 [閒置逾時] 欄位中輸入數值 (以秒爲單位)，超過此時間段之後將從池中移除閒置執行緒。
9. 在 [工作佇列數] 欄位中，輸入由此執行緒池提供服務的工作佇列的總數。
10. 按一下 [確定]。
11. 重新啓動 Application Server。

等效的 `asadmin` 指令爲：`create-threadpool`

編輯執行緒池

1. 在樹形元件中，選擇 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 選取 [執行緒池] 節點。
4. 在 [目前的池] 下選取要變更的執行緒池的名稱。
5. 在 [最小執行緒池大小] 欄位中，輸入爲此佇列中的請求提供服務的執行緒池中執行緒的最小數目。

   創設此執行緒池時將預先建立這些執行緒。
6. 在 [最大執行緒池大小] 欄位中，輸入爲此佇列中的請求提供服務的執行緒池中執行緒的最大數目。

   這是存在於此執行緒池中的執行緒數上限。
7. 在 [閒置逾時] 欄位中輸入數值 (以秒爲單位)，超過此時間段之後將從池中移除閒置執行緒。
8. 在 [工作佇列數] 欄位中，輸入由此執行緒池提供服務的工作佇列的總數。
9. 按一下 [儲存]。
10. 重新啓動 Application Server。

刪除執行緒池

1. 在樹形元件中，選擇 [配置] 節點。
2. 選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要配置所有實例的預設設定，請選取 `default-config` 節點。
3. 選取 [執行緒池] 節點。
4. 在 [目前的池] 表中選取要刪除的執行緒池名稱。
5. 按一下 [刪除]。

6. 重新啓動 Application Server。

等效的 asadmin 指令爲：delete-threadpool

第 20 章

配置記錄

本章簡要描述如何使用管理主控台來配置記錄和檢視伺服器記錄。它包含以下小節：

- 關於記錄
- 有關記錄的管理主控台作業

關於記錄

- 記錄檔的記錄
- 記錄程式名稱空間階層結構

記錄檔的記錄

Application Server 使用在 JSR 047 中指定的 Java 2 平台記錄 API。Application Server 記錄訊息記錄在伺服器記錄中，通常位於 *domain_root_dir*/*domain_dir*/logs/server.log 中。

domain_root_dir/*domain_dir*/logs/ 目錄中除了包含伺服器記錄之外，還包含兩種其他類型的記錄。access 子目錄中包含 HTTP 服務存取記錄，tx 子目錄中包含作業事件服務記錄。如需有關這些記錄的資訊，請參閱第 279 頁的「配置 HTTP 服務存取記錄」和第 269 頁的「配置作業事件」。

Application Server 的元件產生記錄輸出。應用程式元件也可以產生記錄輸出。

應用程式元件可以使用 Apache Commons Logging Library 來記錄訊息。但是，建議採用平台標準 JSR 047 API 以獲得更好的記錄配置。

記錄檔的記錄遵循以下統一格式：

[#|*yyyy-mm-dd*T*hh*:*mm*:*ss*.*SSS*-*Z*| **記錄層級** | **產品名稱版本** | **記錄程式名稱** | **關鍵字值對** | **訊息** |#]

例如：

```
[#|2004-10-21T13:25:53.852-0400|INFO|sun-appserver-ee8.1|javax.enterprise.
system.core|_ThreadID=13;|CORE5004:Resource
Deployed:[cr:jms/DurableConnectionFactory].|#]
```

在本範例中，

- [# 和 #] 標示該記錄的開始和結束。
- 垂直列 (|) 用於分隔記錄欄位。
- 2004-10-21T13:25:53.852-0400 指定了日期和時間。
- **記錄層級**爲 INFO。此層級可以是以下任何值：SEVERE、WARNING、INFO、CONFIG、FINE、FINER 和 FINEST。
- **產品名稱版本**爲 sun-appserver-ee8.1。
- **記錄程式名稱**是用於識別記錄模組源的階層式記錄程式名稱空間，在此示例中爲 javax.enterprise.system.core。
- **關鍵字值對**爲關鍵字名稱和值，通常爲執行緒 ID，如 _ThreadID=14;。
- **訊息**是記錄訊息的文字。對於所有的 Application Server SEVERE 和 WARNING 訊息以及多種 INFO 訊息，它均以包含模組代碼和數值的訊息 ID 開頭 (在此示例中爲 CORE5004)。

以後的版本中，可能會變更或增強記錄檔的記錄格式。

記錄程式名稱空間階層結構

Application Server 爲它的每個模組都提供了記錄程式。下表按照模組名稱和名稱空間在管理主控台的 [記錄層級] 頁面中的顯示方式 (請參閱第 305 頁的「配置記錄層級」)，以字母順序列出每個記錄程式的模組名稱和名稱空間。[記錄層級] 頁面中未顯示表中最後三個模組。

表 20-1 Application Server 記錄程式名稱空間

模組名稱	名稱空間
管理	javax.enterprise.system.tools.admin
Classloader	javax.enterprise.system.core.classloading

表 20-1 Application Server 記錄程式名稱空間 (續)

模組名稱	名稱空間
CMP	javax.enterprise.system.container.cmp
配置	javax.enterprise.system.core.config
連接器	javax.enterprise.resource.resourceadapter
CORBA	javax.enterprise.resource.corba
部署	javax.enterprise.system.tools.deployment
EJB 容器	javax.enterprise.system.container.ejb
JavaMail	javax.enterprise.resource.javamail
JAXR	javax.enterprise.resource.webservices.registry
JAX-RPC	javax.enterprise.resource.webservices.rpc
JDO	javax.enterprise.resource.jdo
JMS	javax.enterprise.resource.jms
JTA	javax.enterprise.resource.jta
JTS	javax.enterprise.system.core.transaction
MDB 容器	javax.enterprise.system.container.ejb.mdb
命名	javax.enterprise.system.core.naming
節點代理程式 (僅限於 Enterprise Edition)	javax.ee.enterprise.system.nodeagent
根	javax.enterprise
SAAJ	javax.enterprise.resource.webservices.saaj
安全性	javax.enterprise.system.core.security
伺服器	javax.enterprise.system
同步 (僅限於 Enterprise Edition)	javax.ee.enterprise.system.tools.synchronization
Util	javax.enterprise.system.util
檢驗器	javax.enterprise.system.tools.verifier
Web 容器	javax.enterprise.system.container.web
核心	javax.enterprise.system.core
系統輸出 (System.out.println)	javax.enterprise.system.stream.out
系統錯誤 (System.err.println)	javax.enterprise.system.stream.err

有關記錄的管理主控台作業

- 配置一般記錄設定
- 配置記錄層級
- 檢視伺服器記錄

配置一般記錄設定

1. 在樹形元件中，展開 [節點代理程式] 節點或 [配置] 節點。
2. 選取節點代理程式，或者選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 `server`，請選取 `server-config` 節點。
   b. 若要爲將來的實例 (實例使用 `default-config` 的副本) 配置預設設定，請選取 `default-config` 節點。
3. 對於節點代理程式，請選取 [記錄程式設定] 標籤。對於配置，請選取 [記錄程式設定] 節點。
4. 在 [記錄設定] 頁面中，可以使用以下欄位來自訂記錄：
   - **記錄檔**：若要爲伺服器記錄檔指定替代名稱或位置，請在文字欄位中鍵入新的路徑名稱。依預設，該位置爲 *domain_root_dir*`/`*domain_dir*`/logs/server.log`。
   - **警示**：若要經由 JMX 框架路由 `SEVERE` 和 `WARNING` 訊息，請選取 [啓用] 核取方塊。
   - **寫入系統記錄**：僅在 Solaris 和 Linux 系統中，除了要將記錄輸出傳送給伺服器記錄外，還要將其傳送給 `syslog` 工具，請選取 [啓用] 核取方塊。
   - **記錄處理程式**：若要將記錄傳送到 `server.log` 或 `syslog` 以外的目標，您可以插入自訂記錄處理程式。自訂處理程式必須延伸 `java.util.logging.Handler` 類別 (JSR 047 相容 API)。在 [記錄處理程式] 欄位中鍵入處理程式的絕對類別名稱。還應將處理程式類別置於 Application Server 類別路徑中，以便在伺服器啓動期間安裝該處理程式。自訂處理程式的記錄檔的記錄具有在第 301 頁的「記錄檔的記錄」中描述的格式。

- **記錄篩選器**：若要篩選傳送給目標 (如 server.log、syslog 或由自訂記錄處理程式指定的目標) 的記錄檔的記錄，可以插入自訂記錄篩選器。該自訂篩選器必須實作介面 java.util.logging.Filter。在 [記錄篩選器] 欄位中鍵入篩選器的絕對類別名稱。還應將篩選器類別置於 Application Server 類別路徑中，以便在伺服器啓動期間安裝該篩選器。
- **檔案自動重建限制**：如果伺服器記錄的位元組數達到了指定大小，請建立一個名爲 server.log 的新的空檔案，並將舊檔案重新命名爲 server.log_*date*，其中 *date* 是自動重建檔案的日期和時間。預設值爲 2 百萬位元組。該限制的最小值爲 500 千位元組，如果指定較低的值，則要在達到 500 千位元組時該檔案才自動重建。若要關閉記錄檔自動重建，請將該值設定爲 0。
- **檔案自動重建時間限制**：達到指定的分鐘數之後才自動重建伺服器記錄。預設值爲零，這表示檔案達到 [檔案自動重建限制] 欄位中指定的大小即會自動重建。**如果指定了非零的時間限制，則該時間限制優先於上述大小限制**。

5. 按一下 [儲存]，以儲存變更。按一下 [檢視記錄檔] 以檢視伺服器記錄。

配置記錄層級

1. 在樹形元件中，展開 [節點代理程式] 節點或 [配置] 節點。
2. 選取節點代理程式，或者選取要配置的實例：
   a. 若要配置特定的實例，請選取該實例的配置節點。例如，對於預設實例 server，請選取 server-config 節點。
   b. 若要爲將來的實例 (使用 default-config 的副本) 配置預設設定，請選取 default-config 節點。
3. 對於節點代理程式，請選取 [記錄層級] 標籤。對於配置，請選取 [記錄程式設定] 節點，然後選取 [記錄層級] 標籤。
4. 在 [模組記錄層級] 頁面中，請從要變更記錄層級的一個或多個模組相對的下拉式清單中選擇一個新值。預設層級爲 INFO，表示處於該層級或更高層級 (WARNING、SEVERE) 的訊息將顯示在記錄中。可以選擇以下任一值 (以最高層級至最低層級的順序列示)：
   - SEVERE
   - WARNING
   - INFO
   - CONFIG

- FINE
- FINER
- FINEST
- OFF

5. 使用 [附加特性] 區域可以爲任何應用程式記錄程式配置記錄層級。特性名稱是記錄程式名稱空間，其值爲上述八個可能的層級之一。例如，特性名稱可爲 samples.logging.simple.servlet，值可爲 FINE。

   還可以使用此區域來變更子模組的記錄層級，例如 CORBA 模組的傳輸子模組：

   javax.enterprise.resource.corba.*ORBId*.transport

6. 按一下 [儲存] 以儲存變更，或按一下 [載入預設值] 以復原預設值。

對 System.out.println 的呼叫均在 INFO 層級使用記錄程式名稱 javax.enterprise.system.stream.out 予以記錄。對 System.err.println 的呼叫均在 WARNING 層級使用記錄程式名稱 javax.enterprise.system.stream.err 予以記錄。若要從這些源關閉記錄，請在 [附加特性] 區域中將記錄程式名稱的值指定爲 OFF。

對記錄層級設定所做的變更將立即生效。同時將在 domain.xml 檔案中儲存這些變更，以供伺服器重新啓動時使用。

檢視伺服器記錄

1. 在樹形元件中，展開要檢視其記錄的伺服器實例的節點。
2. 在 [一般資訊] 頁面中，按一下 [檢視記錄檔]。

使用 [搜尋條件] 區域自訂和篩選記錄檢視器。使用如下所示的基本欄位：

- **實例名稱**：從下拉式清單中選擇一個實例名稱，以檢視該伺服器實例的記錄。預設值爲目前伺服器實例。
- **記錄檔**：從下拉式清單中選擇一個記錄檔名稱，以檢視該記錄的特性。預設爲 server.log。
- **時間戳記**：若要檢視最新的訊息，請選取 [最近] (預設值)。如果只檢視特定時間段內的訊息，請選取 [特定範圍] 並在顯示的 [從] 欄位和 [到] 欄位中鍵入日期和時間值。對於時間值，其語法必須採用以下形式 (*SSS* 表示毫秒)：

  hh:*mm*:*ss*.*SSS*

  例如：

```
17:10:00.000
```

如果 [從] 欄位中的時間值遲於 [到] 欄位中的時間值，將顯示錯誤訊息。

- **記錄層級**：若要按照記錄層級篩選訊息，請從下拉式清單中選擇一種記錄層級。依預設，將顯示伺服器記錄中選取記錄層級和更嚴重記錄層級的所有訊息。選取標有「不包括更高層級的訊息」的核取方塊將僅顯示所選擇層級的訊息。

  若要確定您要檢視的訊息都顯示在伺服器記錄中，請先在 [記錄層級] 頁面中設定適當的記錄層級。請參閱第 305 頁的「配置記錄層級」。

  如果您選擇基於記錄層級篩選記錄訊息，則將只顯示符合指定篩選條件的訊息。不過，這種篩選不影響那些記錄到伺服器記錄中的訊息。

將顯示伺服器記錄中最新的 40 個項目以及在 [記錄設定] 和 [記錄層級] 頁面中指定的設定。

按一下 [時間戳記] 標頭旁邊的三角形對這些訊息進行排序，以使最新的訊息顯示在最後。

若要檢視訊息的格式化版本，請按一下標示的連結

（詳細資訊）

將顯示標有 [記錄項目詳細資訊] 的視窗，該視窗包含了訊息的格式化版本。

在項目清單的末尾，按一下按鈕以檢視記錄檔中較早或較晚的項目。

按一下 [搜尋條件] 區域中的 [進階搜尋] 以進一步細化記錄檢視器的搜尋條件。使用如下所示的 [進階選項] 欄位：

- **記錄程式**：若要依模組篩選，請從下拉式清單中選擇一個或多個名稱空間。按住 Shift 鍵並按一下或按住 Ctrl 鍵並按一下來選擇多個名稱空間。

  選取更高層級的名稱空間也就選取了該名稱空間下的所有名稱空間。例如，選取 `javax.enterprise.system` 的同時也就選取了該名稱空間下所有模組的記錄程式：`javax.enterprise.system.core`、`javax.enterprise.system.tools.admin` 等等。

- **自訂記錄程式**：若要檢視特定於特定應用程式的記錄程式中的訊息，請在文字欄位中鍵入記錄程式名稱，每行一個名稱。如果應用程式具有多個模組，您可以檢視任何模組，也可以檢視所有模組。例如，假定應用程式具有使用以下名稱的記錄程式：

  ```
  com.mycompany.myapp.module1
  com.mycompany.myapp.module2
  com.mycompany.myapp.module3
  ```

  若要檢視應用程式中的所有模組的訊息，請鍵入 `com.mycompany.myapp`。若要只檢視 `module2` 的訊息，則鍵入 `com.mycompany.myapp.module2`。

如果指定了一個或多個自訂記錄程式，則僅當您在 [記錄程式] 區域中明確地指定 Application Server 模組的訊息之後，才會顯示這些訊息。

- **名稱值對**：若要檢視特定執行緒的輸出，請在文字欄位中鍵入該執行緒的關鍵字名稱和值。關鍵字名稱爲 `_ThreadID`。例如：

  ```
  _ThreadID=13
  ```

  假定 `com.mycompany.myapp.module2` 在多個執行緒中執行。若要限制記錄檢視器使其只顯示單一執行緒的輸出，請在 [自訂記錄程式] 欄位中指定該模組的記錄程式，然後在此欄位中指定執行緒 ID。

- **顯示**：若要一次檢視 40 條以上的訊息 (預設值)，請從下拉式清單中選擇其他可用的值 (100、250 或 1000)。

  若要檢視堆疊追蹤，請取消選取 [限制過長訊息] 核取方塊。依預設，檢視器中不會顯示堆疊追蹤，若要檢視堆疊追蹤，請按一下訊息的（詳細資訊）連結。

按一下 [基本搜尋] 可以隱藏 [進階選項] 區域。

第 21 章

監視元件和服務

本章包含有關使用 Application Server 管理主控台監視元件的資訊。它包含以下小節：

- 關於監視
- 有關啓用和停用監視功能的管理主控台作業
- 有關檢視監視資料的管理主控台作業

關於監視

- 監視 Application Server
- 監視概述
- 關於可監視物件的樹狀結構
- 關於受監視的元件和服務的統計資訊

監視 Application Server

使用監視功能可以觀察在 Sun Java System Application Server Enterprise Edition 8.1 2005Q1 的伺服器實例中所部署的各種元件和服務的執行階段狀態。利用有關執行階段元件和程式狀態的資訊，可以確定效能瓶頸以便進行調校、和容量規劃、預測故障、在發生故障時分析根本原因，以及確保一切執行正常。

啓用監視功能會因增加系統耗用而使效能降低。

監視概述

若要監視 Application Server，請執行以下步驟：

1. 使用管理主控台或 asadmin 工具啓用對特定服務和元件的監視功能。

   如需有關此步驟的更多資訊，請參閱「有關啓用和停用監視功能的管理主控台作業」。

2. 使用管理主控台或 asadmin 工具檢視指定服務或元件的監視資料。

   如需有關此步驟的更多資訊，請參閱「有關檢視監視資料的管理主控台作業」。

關於可監視物件的樹狀結構

Application Server 使用樹狀結構來追蹤可監視物件。由於監視物件的樹是動態的，因此在實例中增加、更新或移除元件時該樹會相應地發生變更。樹中的根物件是伺服器實例名稱 (例如 server)。(在 Platform Edition 中，僅允許使用一個伺服器實例。)

以下指令顯示了樹的頂層：

```
asadmin> list --monitor server
server.applications
server.http-service
server.connector-service
server.jms-service
server.jvm
server.orb
server.resources
server.thread-pools
```

以下小節描述了這些子樹：

- 應用程式樹
- HTTP 服務樹
- 連接器服務樹
- 連接器服務樹
- JMS 服務樹
- ORB 樹
- 執行緒池樹

應用程式樹

以下示意圖顯示了企業應用程式的各種元件的頂層節點和子節點。具有可用的監視統計資訊的節點標有星號 (*)。如需更多資訊，請參閱 「EJB 容器統計資訊」和「Web 容器統計資訊」。

圖 21-1 應用程式節點樹狀結構

```
應用程式
   |--- 應用程式 1
   |     |--- ejb 模組 1
   |     |         |--- ejb1 *
   |     |                   |--- 快取 ( 用於實體 /sfsb) *
   |     |                   |--- 池 ( 用於 slsb/mdb/ 實體 ) *
   |     |                   |--- methods
   |     |                        |---method1 *
   |     |                        |---method2 *
   |     |                   |--- stateful-session-store (for sfsb)*
   |     |                   |--- timers (for s1sb/entity/mdb) *
   |     |--- web-module-1
   |     |         |--- virtual-server-1 *
   |     |                        |---servlet1  *
   |     |                        |---servlet2  *
   |--- standalone-web-module-1
   |     |         |----- virtual-server-2 *
   |     |                        |---servlet3 *
   |     |                        |---servlet4 *
   |     |         |----- virtual-server-3 *
   |     |                       |---servlet3 *(same servlet on different vs)
   |     |                       |---servlet5 *
   |--- standalone-ejb-module-1
   |     |         |--- ejb2 *
   |     |                   |--- cache (for entity/sfsb) *
   |     |                   |--- pool (for slsb/mdb/entity) *
   |     |                   |--- methods
   |     |                        |--- method1 *
   |     |                        |--- method2 *
   |--- application2
```

HTTP 服務樹

以下示意圖顯示了 HTTP 服務的節點。具有可用的監視資訊的節點標有星號 (*)。請參閱 「HTTP 服務樹」。

圖 21-2 HTTP 服務示意圖 (PE 版)

```
http-service
        |--- virtual-server-1
           |--- http-listener-1 *
           |--- http-listener-2 *
        |--- virtual-server-2
           |--- http-listener-1 *
           |--- http-listener-2 *
```

圖 21-3 HTTP 服務示意圖 (EE 版)

```
http-service *
            |---connection-queue *
            |---dns *
            |---file-cache *
            |---keep-alive *
            |---pwc-thread-pool *
            |---virtual-server-1*
            |        |--- request *
            |---virtual-server-2*
            |        |--- request *
```

資源樹

資源節點包含 JDBC 連線池和連接器連線池等池的可監視屬性。以下示意圖顯示了各種資源元件的頂層節點和子節點。具有可用的監視統計資訊的節點標有星號 (*)。請參閱 「JDBC 連線池統計資訊」和 「JMS/ 連接器服務統計資訊」。

圖 21-4 資源示意圖

```
resources
       |---connection-pool1(either connector-connection-pool or jdbc)*
       |---connection-pool2(either connector-connection-pool or jdbc)*
```

連接器服務樹

連接器服務節點包含連接器連線池等池的可監視屬性。以下示意圖顯示了各種連接器服務元件的頂層節點和子節點。具有可用的監視統計資訊的節點標有星號 (*)。請參閱 「JMS/ 連接器服務統計資訊」。

圖 21-5 連接器服務示意圖

```
connector-service
        |--- resource-adapter-1
        |       |-- connection-pools
        |       |       |-- pool-1 (All pool stats for this pool)
        |       |-- work-management (All work mgmt stats for this RA)
```

JMS 服務樹

JMS 服務節點包含連接器連線池等池的可監視屬性。以下示意圖顯示了各種 JMS 服務元件的頂層節點和子節點。具有可用的監視統計資訊的節點標有星號 (*)。

圖 21-6 JMS 服務示意圖

```
jms-service
        |-- connection-factories [AKA conn. pools in the RA world]
        |       |-- connection-factory-1 (All CF stats for this CF)
        |-- work-management (All work mgmt stats for the MQ-RA)
```

ORB 樹

ORB 節點包含連線管理程式的可監視屬性。以下示意圖顯示了 ORB 元件的頂層節點和子節點。具有可用的監視統計資訊的節點標有星號 (*)。請參閱 「ORB 中連線管理程式的統計資訊」。

圖 21-7 ORB 示意圖

```
orb
        |--- connection-managers
        |       |--- connection-manager-1 *
        |       |--- connection-manager-1 *
```

執行緒池樹

執行緒池節點包含連線管理程式的可監視屬性。以下示意圖顯示了 ORB 元件的頂層節點和子節點。具有可用的監視統計資訊的節點標有星號 (*)。請參閱 「執行緒池統計資訊」。

圖 21-8 執行緒池示意圖

```
thread-pools
       |   |--- thread-pool-1 *
       |   |--- thread-pool-2 *
```

關於受監視的元件和服務的統計資訊

本小節描述可用的監視統計資訊：

- EJB 容器統計資訊
- Web 容器統計資訊
- HTTP 服務統計資訊
- JDBC 連線池統計資訊
- JMS/ 連接器服務統計資訊
- ORB 中連線管理程式的統計資訊
- 執行緒池統計資訊
- 作業事件服務統計資訊
- Java 虛擬機器 (JVM) 統計資訊
  - J2SE 5.0 中的 JVM 統計資訊
- 生產 Web 容器 (PWC) 統計資訊

EJB 容器統計資訊

表 21-1 中介紹了 EJB 統計資訊。

表 21-1 EJB 統計資訊

屬性名稱	資料類型	描述
createcount	計數 統計資訊	呼叫 EJB 的 create 方法的次數。
removecount	計數 統計資訊	呼叫 EJB 的 remove 方法的次數。
pooledcount	範圍 統計資訊	處於匯集狀態的實體 Bean 的數目。
readycount	範圍 統計資訊	處於就緒狀態的實體 Bean 的數目。
messagecount	計數 統計資訊	訊息導引 Bean 收到的訊息數。
methodreadycount	範圍 統計資訊	處於 MethodReady 狀態的有狀態或無狀態階段作業 Bean 的數目。
passivecount	範圍 統計資訊	處於 Passive 狀態的有狀態階段作業 Bean 的數目。

表 21-2 中列示了可用於 EJB 方法呼叫的統計資訊。

表 21-2 EJB 方法統計資訊

屬性名稱	資料類型	描述
methodstatistic	時間 統計資訊	作業被呼叫的次數；呼叫期間所花費的總時間等資訊。
totalnumerrors	計數 統計資訊	執行方法導致異常的次數。如果為 EJB 容器啟用了監視功能，則此統計資訊是為無狀態和有狀態階段作業 Bean 和實體 Bean 而收集的。
totalnumsuccess	計數 統計資訊	方法成功執行的次數。如果為 EJB 容器啟用了監視功能，則此統計資訊是為無狀態和有狀態階段作業 Bean 和實體 Bean 收集的。
executiontime	計數 統計資訊	上次成功/不成功嘗試執行方法作業所花費的時間 (ms)。如果在 EJB 容器中啟用了監視功能，則此統計資訊是為無狀態和有狀態階段作業 Bean 和實體 Bean 而收集的。

表 21-3 中列示了 EJB 階段作業儲存的統計資訊。

表 21-3 EJB 階段作業儲存統計資訊

屬性名稱	資料類型	描述
currentSize	範圍 統計資訊	目前位於儲存中的鈍化階段作業數或檢查點階段作業數。
activationCount	計數 統計資訊	從儲存中啟動的階段作業數。
activationSuccessCount	計數 統計資訊	從儲存中成功啟動的階段作業數
activationErrorCount	計數 統計資訊	上次成功/不成功嘗試執行方法作業所花費的時間 (ms)。如果在 EJB 容器中啟用了監視功能，則此統計資訊是為無狀態和有狀態階段作業 Bean 和實體 Bean 而收集的。
passivationCount	計數 統計資訊	使用此儲存鈍化 (取消啟動) 的階段作業數。
passivationSuccessCount	計數 統計資訊	使用此儲存成功鈍化的階段作業數。
passivationErrorCount	計數 統計資訊	無法使用此儲存鈍化的階段作業數。
expiredSessionCount	計數 統計資訊	此儲存移除的過期階段作業數。
passivatedBeanSize	計數 統計資訊	由此儲存鈍化的總位元組數，包括總數、最小值和最大值。
passivationTime	計數 統計資訊	將 Bean 鈍化到儲存所花費的時間，包括總時間值、最小值和最大值。
checkpointCount (僅限於 EE)	計數 統計資訊	使用此儲存進行階段作業檢查點操作的階段作業數。
checkpointSuccessCount (僅限於 EE)	計數 統計資訊	成功進行檢查點操作的階段作業數。
checkpointErrorCount (僅限於 EE)	計數 統計資訊	無法進行檢查點操作的階段作業數。
checkpointedBeanSize (僅限於 EE)	值 統計資訊	由該儲存進行檢查點操作的 Bean 的總數。
checkpointTime (僅限於 EE)	時間 統計資訊	通過檢查點操作將 Bean 放入儲存中所花費的時間。

表 21-4 中列示了可用於 EJB 池的統計資訊。

表 21-4 EJB 池統計資訊

屬性名稱	資料類型	描述
numbeansinpool	已連結 範圍 統計資訊	相關聯池中的 EJB 數，提供有關池的變更方式的資訊。
numthreadswaiting	已連結 範圍 統計資訊	等待自由 Bean 的執行緒數，指出請求可能擁塞。
totalbeanscreated	計數 統計資訊	從開始收集資料以來相關聯池中所建立的 Bean 的數目。
totalbeansdestroyed	計數 統計資訊	自開始收集資料以來從相關聯池中銷毀的 Bean 的數目。
jmsmaxmessagesload	計數 統計資訊	針對要服務的訊息導引 Bean 而一次載入 JMS 階段作業的最大訊息數。預設值為 1。僅套用於訊息導引 Bean 的池。

表 21-5 中列示了可用於 EJB 快取的統計資訊。

表 21-5 EJB 快取統計資訊

屬性名稱	資料類型	描述
cachemisses	已連結 範圍 統計資訊	使用者請求未在快取中找到 Bean 的次數。
cachehits	已連結 範圍 統計資訊	使用者請求找到快取記憶體中某個項目的次數。
numbeansincache	已連結 範圍 統計資訊	快取記憶體中 Bean 的數目。這便是快取記憶體目前的大小。
numpassivations	計數 統計資訊	鈍化的數目。僅套用於有狀態階段作業 Bean。
numpassivationerrors	計數 統計資訊	鈍化期間發生的錯誤數。僅套用於有狀態階段作業 Bean。
numexpiredsessionsremoved	計數 統計資訊	清除執行緒所移除的過期階段作業數目。僅套用於有狀態階段作業 Bean。
numpassivationsuccess	計數 統計資訊	鈍化成功完成的次數。僅套用於有狀態階段作業 Bean。

表 21-6 中列示了可用於計時器的統計資訊。

表 21-6 計時器統計資訊

統計資訊	資料類型	描述
numtimerscreated	計數統計資訊	系統中建立的計時器的數目。
numtimersdelivered	計數統計資訊	系統所傳送的計時器的數目。
numtimersremoved	計數統計資訊	從系統中移除的計時器的數目。

Web 容器統計資訊

圖 21-1 中所示的物件樹中包含了 Web 容器。系統爲每個單獨的 Web 應用程式都顯示了 Web 容器統計資訊。表 21-7 中顯示了可用於 Servlet 的 Web 容器的統計資訊，表 21-8 中顯示了可用於 Web 模組的統計資訊。

表 21-7 Web 容器 (Servlet) 統計資訊

統計資訊	單位	資料類型	注釋
errorcount	數目	計數統計資訊	回應碼大於或等於 400 的情況的累積次數。
maxtime	毫秒	計數統計資訊	Web 容器等待請求的最長時間。
processingtime	毫秒	計數統計資訊	處理每個請求所需時間的累積值。處理時間是總請求處理時間除以請求計數所得的平均值。
requestcount	數目	計數統計資訊	到目前為止所處理的請求總數。

表 21-8 中顯示了可用於 Web 模組的統計資訊。

表 21-8 Web 容器 (Web 模組) 統計資訊

統計資訊	資料類型	注釋
jspcount	計數統計資訊	已載入 Web 模組的 JSP 頁面的數目。
jspreloadcount	計數統計資訊	已重新載入 Web 模組的 JSP 頁面的數目。
sessionstotal	計數統計資訊	已為 Web 模組建立的階段作業總數。
activesessionscurrent	計數統計資訊	Web 模組的目前處於使用中狀態的階段作業數。
activesessionshigh	計數統計資訊	Web 模組的同時處於使用中狀態的階段作業最大數。

表 21-8 Web 容器 (Web 模組) 統計資訊 (續)

統計資訊	資料類型	注釋
rejectedsessionstotal	計數統計資訊	Web 模組的被拒絕的階段作業總數。是指由於允許處於使用中狀態的階段作業數已達到最大值而未被建立的階段作業數。
expiredsessionstotal	計數統計資訊	Web 模組的已過期階段作業的總數。
sessionsize (僅限於 EE)	平均範圍統計資訊	Web 模組的階段作業大小。值可以為大、小或平均值，或以位元組為單位 (用於串列化階段作業)。
containerlatency (僅限於 EE)	平均範圍統計資訊	整個延時請求中 Web 容器部分的延時。值可以是長、短或平均值。
sessionpersisttime (僅限於 EE)	平均範圍統計資料	將 HTTP 階段作業狀態保持到 Web 模組的後端儲存中所花費的時間 (以 ms 為單位，短、長或平均值)。
cachedsessionscurrent (僅限於 EE)	計數統計資訊	快取在記憶體中用於 Web 模組的目前階段作業數。
passivatedsessionscurrent (僅限於 EE)	計數統計資訊	Web 模組的目前已鈍化的階段作業數。

HTTP 服務統計資訊

表 21-9 中顯示了可用於 HTTP 服務的統計資訊。這些統計資訊僅適用於 Platform Edition。如需有關企業版上的 HTTP 服務的統計資訊，請參閱表 21-32。

表 21-9 HTTP 服務統計資訊 (僅適用於 Platform Edition)

統計資訊	單位	資料類型	注釋
bytesreceived	位元組	計數統計資訊	每個請求處理器接收到的位元組累積值。
bytessent	位元組	計數統計資訊	每個請求處理器所傳送的位元組累積值。
currentthreadcount	數目	計數統計資訊	目前位於偵聽程式執行緒池中的處理執行緒數。
currentthreadsbusy	數目	計數統計資訊	處理請求的偵聽程式執行緒池中目前正在使用的請求處理執行緒的數目。

表 21-9 HTTP 服務統計資訊 (僅適用於 Platform Edition)(續)

統計資訊	單位	資料類型	注釋
errorcount	數目	計數統計資訊	錯誤計數的累積值，錯誤計數是指回應碼大於或等於 400 這類情況發生的次數。
maxsparethreads	數目	計數統計資訊	可以存在的未使用回應處理執行緒的最大數目。
minsparethreads	數目	計數統計資訊	可以存在的未使用回應處理執行緒的最小數目。
maxthreads	數目	計數統計資訊	偵聽程式所建立的請求處理執行緒的最大數目。
maxtime	毫秒	計數統計資訊	處理執行緒的最長時間。
processing-time	毫秒	計數統計資訊	處理每個請求所花費時間的累積值。處理時間是總請求處理時間除以請求計數所得的平均值。
request-count	數目	計數統計資訊	到目前為止所處理的請求總數。

JDBC 連線池統計資訊

用於在執行階段監視 JDBC 資源，以測量效能並擷取資源使用情況。由於建立 JDBC 連線的成本很高並且常常會導致應用程式出現效能瓶頸問題，因此對 JDBC 連線池如何釋放和建立新連線以及正在等待從特定池中擷取連線的執行緒數的監視是至關重要的。

表 21-10 中顯示了可用於 JDBC 連線池的統計資訊。

表 21-10 JDBC 連線池統計資訊

統計資訊	單位	資料類型	描述
numconnfailedvalidation	數目	計數統計資訊	從開始時間到上次取樣時間為止在連線池中驗證失敗的連線總數。

表 21-10 JDBC 連線池統計資訊 (續)

統計資訊	單位	資料類型	描述
numconnused	數目	範圍統計資訊	提供連線使用統計資訊。目前正使用的連線的總數，以及有關使用過的連線的最大數目 (高水印) 的資訊。
numconnfree	範圍統計資訊	計數統計資訊	上次取樣時池中的自由連線總數。
numconntimedout	計數統計資訊	已連結範圍統計資訊	開始時間與上次取樣時間之間池中的逾時連線總數。
averageconnwaittime	數目	計數統計資訊	指示嘗試與連接器連線池建立成功連線請求的平均等待時間。
waitqueuelength	數目	計數統計資訊	佇列中正在等待處理的連線請求數。
connectionrequestwaittime		範圍統計資訊	連線請求的最長和最短等待時間。目前值表示上次由池處理的請求的等待時間。
numconncreated	毫秒	計數統計資訊	自上次重設以來建立的實體連線數。
numconndestroyed	數目	計數統計資訊	自上次重設以來已銷毀的實體連線數。
numconnacquired	數目	計數統計資訊	從池中獲取的邏輯連線數。
numconnreleased	數目	計數統計資訊	釋放到池中的邏輯連線數。

JMS/ 連接器服務統計資訊

表 21-11 中顯示了可用於連接器連線池的統計資訊。表 21-12 中顯示了連接器工作管理的統計資訊。

表 21-11 連接器連線池統計資訊

統計資訊	單位	資料類型	描述
numconnfailed validation	數目	計數 統計資訊	從開始時間到上次取樣時間為止在連線池中驗證失敗的連線總數。
numconnused	數目	範圍 統計資訊	提供連線使用統計資訊。目前正使用的連線總數，以及有關使用過的連線的最大數目的資訊 (高水印)。
numconnfree	數目	範圍 統計資訊	上次取樣時池中的自由連線總數。
numconntimedout	數目	計數 統計資訊	開始時間與上次取樣時間之間池中的逾時連線總數。
averageconnwaittime	數目	計數 統計資訊	連線由連線池處理之前的平均等待時間。
waitqueuelenght	數目	計數 統計資訊	佇列中正在等待處理的連線請求數。
connectionrequest waittime		範圍 統計資訊	連線請求的最長和最短等待時間。目前值表示上次由池處理的請求的等待時間。
numconncreated	毫秒	計數統計資訊	自上次重設以來建立的實體連線數。
numconndestroyed	數目	計數 統計資訊	自上次重設以來已銷毀的實體連線數。
numconnacquired	數目	計數 統計資訊	從池中獲取的邏輯連線數。
numconnreleased	數目	計數 統計資訊	釋放到池中的邏輯連線數。

表 21-12 中列示了可用於連接器工作管理的統計資訊。

表 21-12 連接器工作管理統計資訊

統計資訊	資料類型	描述
activeworkcount	範圍 統計資訊	由連接器執行的工作物件數。
waitqueuelength	範圍 統計資訊	執行前在佇列中等待的工作物件數。

表 21-12 連接器工作管理統計資訊 (續)

統計資訊	資料類型	描述
workrequestwaittime	範圍統計資訊	工作物件在被執行前所等待的最長和最短時間。
submittedworkcount	計數統計資訊	由連接器模組提交的工作物件數。
rejectedworkcount	計數統計資訊	Application Server 拒絕的工作物件數。
completedworkcount	計數統計資訊	完成的工作物件數。

ORB 中連線管理程式的統計資訊

表 21-13 中列示了可用於 ORB 中連線管理程式的統計資訊。

表 21-13 ORB 中連線管理程式的統計資訊

統計資訊	單位	資料類型	描述
connectionsidle	數目	計數統計資訊	提供與 ORB 的閒置連線的總數。
connectionsinuse	數目	計數統計資訊	提供與 ORB 的正在使用的連線總數。
totalconnections	數目	綁定範圍統計資訊	與 ORB 的連線總數。

執行緒池統計資訊

表 21-14 中顯示了可用於執行緒池的統計資訊。

表 21-14 執行緒池統計資訊

統計資訊	單位	資料類型	描述
averagetimeinqueue	毫秒	範圍統計資訊	在得到處理之前請求在佇列中等待的平均時間 (以毫秒為單位)。
averageworkcompletion-time	毫秒	範圍統計資訊	完成指定所花費的平均時間 (以毫秒為單位)。
currentnumberofthreads	數目	綁定範圍統計資訊	目前的請求處理執行緒的數目。
numberofavailablethreads	數目	計數統計資訊	可用的執行緒數。
numberofbusythreads	數目	計數統計資訊	處於忙碌狀態的執行緒數。

表 21-14 執行緒池統計資訊 (續)

統計資訊	單位	資料類型	描述
totalworkitemsadded	數目	計數統計資訊	到目前為止增加到工作佇列中的工作項目總數。

作業事件服務統計資訊

作業事件服務允許用戶端凍結作業事件子系統，以便轉返作業事件，並確定在凍結期間進行處理的作業事件。表 21-15 中顯示了可用於作業事件服務的統計資訊。

表 21-15 作業事件服務統計資訊

統計資訊	資料類型	描述
activecount	計數統計資訊	目前處於使用中狀態的作業事件數。
activeids	字串統計資訊	目前處於使用中狀態的作業事件的 ID。每個此類作業事件均可在凍結作業事件服務後轉返。
committedcount	計數統計資訊	已確定的作業事件數。
rolledbackcount	計數統計資訊	已轉返的作業事件數。
狀態	字串統計資訊	指示作業事件是否已凍結。

Java 虛擬機器 (JVM) 統計資訊

JVM 具有始終處於啓用狀態的可監視屬性。表 21-16 中顯示了可用於 JVM 的統計資訊。

表 21-16 JVM 統計資訊

統計資訊	資料類型	描述
heapsize	綁定範圍統計資訊	具有 JVM 的記憶體堆疊大小的較高和較低連結的常駐記憶體足跡。
uptime	計數統計資訊	JVM 已執行的時間。

J2SE 5.0 中的 *JVM* 統計資訊

如果 Application Server 被配置爲執行 J2SE 5.0 或更高版本，則可以從 JVM 中獲得附加監視資訊。將監視層級設定爲 「低」以啓用這些附加資訊的顯示。將監視層級設定爲 「高」還可以檢視與系統中每個活性執行緒相關的資訊。如需有關 J2SE 5.0 中可用的附加監視功能的更多資訊，請參閱以下 URL 中標題爲 「Monitoring and Management for the Java Platform」的文件：

http://java.sun.com/j2se/1.5.0/docs/guide/management/

在以下位置討論了 J2SE 5.0 監視工具：

http://java.sun.com/j2se/1.5.0/docs/tooldocs/#manage

表 21-17 中顯示了可用於在 J2SE 5.0 的 JVM 中進行類別載入的統計資訊。

表 21-17 J2SE 5.0 的 JVM 統計資訊 — 類別載入

統計資訊	資料類型	描述
loadedclasscount	計數統計資訊	目前載入 JVM 的類別的數目。
totalloadedclasscount	計數統計資訊	自 JVM 開始執行以來已載入的類別的總數。
unloadedclasscount	計數統計資訊	自 JVM 開始執行以來已從 JVM 中卸載的類別的數目。

表 21-18 中顯示了可用於在 J2SE 5.0 的 JVM 中進行編譯的統計資訊。

表 21-18 J2SE 5.0 的 JVM 統計資訊 — 編譯

統計資訊	資料類型	描述
totalcompilationtime	計數統計資訊	編譯所花費的累積時間 (以毫秒為單位)。

表 21-19 中顯示了可用於在 J2SE 5.0 的 JVM 中進行資源回收的統計資訊。

表 21-19 J2SE 5.0 的 JVM 統計資訊 — 資源回收

統計資訊	資料類型	描述
collectioncount	計數統計資訊	已發生的回收的總數。
collectiontime	計數統計資訊	累積的收集時間 (以毫秒為單位)。

表 21-20 中顯示了可用於 J2SE 5.0 的 JVM 中的記憶體的統計資訊。

表 21-20 J2SE 5.0 的 JVM 統計資訊 — 記憶體

統計資訊	資料類型	描述
objectpendingfinalizationcount	計數統計資訊	擱置結束操作的物件的大約數目。
initheapsize	計數統計資訊	最初由 JVM 請求的堆疊的大小。
usedheapsize	計數統計資訊	目前正在使用的堆疊的大小。
maxheapsize	計數統計資訊	可用於記憶體管理的最大記憶體容量 (以位元組為單位)。
committedheapsize	計數統計資訊	確定可由 JVM 使用的記憶體容量 (以位元組為單位)。
initnonheapsize	計數統計資訊	最初由 JVM 請求的非堆疊區域的大小。
usednonheapsize	計數統計資訊	目前正在使用的非堆疊區域的大小。
maxnonheapsize	計數統計資訊	可用於記憶體管理的最大記憶體容量 (以位元組為單位)。
committednonheapsize	計數統計資訊	確定可由 JVM 使用的記憶體容量 (以位元組為單位)。

表 21-21 中顯示了可用於 J2SE 5.0 的 JVM 中的作業系統的統計資訊。

表 21-21 J2SE 5.0 的 JVM 統計資訊 — 作業系統

統計資訊	資料類型	描述
arch	字串統計資訊	作業系統架構。
availableprocessors	計數統計資訊	可用於 JVM 的處理器的數目。
name	字串統計資訊	作業系統名稱。
version	字串統計資訊	作業系統版本。

表 21-22 中顯示了可用於 J2SE 5.0 的 JVM 中執行階段的統計資訊。

表 21-22 J2SE 5.0 的 JVM 統計資訊 — 執行階段

統計資訊	資料類型	描述
name	字串統計資訊	表示正在執行的 JVM 的名稱
vmname	字串統計資訊	JVM 實作名稱。
vmvendor	字串統計資訊	JVM 實作供應商。

表 21-22 J2SE 5.0 的 JVM 統計資訊 — 執行階段 (續)

統計資訊	資料類型	描述
vmversion	字串統計資訊	JVM 實作版本。
specname	字串統計資訊	JVM 規格名稱。
specvendor	字串統計資訊	JVM 規格供應商。
specversion	字串統計資訊	JVM 規格版本。
managementspecversion	字串統計資訊	由 JVM 實作的管理規格版本。
classpath	字串統計資訊	系統類別載入器搜尋類別檔案時所使用的類別路徑。
librarypath	字串統計資訊	Java 程式庫路徑。
bootclasspath	字串統計資訊	啟動程式類別載入器搜尋類別檔案時所使用的類別路徑。
inputarguments	字串統計資訊	傳送給 JVM 的輸入引數。不包括 main 方法的引數。
uptime	計數統計資訊	JVM 的正常執行時間 (以毫秒為單位)。

表 21-23 中顯示了可用於 J2SE 5.0 的 JVM 中的 ThreadInfo 的統計資訊。

表 21-23 J2SE 5.0 的 JVM 統計資訊 — 執行緒資訊

統計資訊	資料類型	描述
threadid	計數統計資訊	執行緒 ID。
threadname	字串統計資訊	執行緒名稱。
threadstate	字串統計資訊	執行緒狀態。
blockedtime	計數統計資訊	執行緒進入 BLOCKED 狀態以來所經歷的時間 (以毫秒為單位)。如果已停用執行緒競爭狀態監視功能，則傳回 -1。
blockedcount	計數統計資訊	執行緒進入 BLOCKED 狀態的總次數。
waitedtime	計數統計資訊	執行緒處於 WAITING 狀態所經歷的時間 (以毫秒為單位)。如果已停用執行緒競爭狀態監視功能，則傳回 -1。
waitedcount	計數統計資訊	執行緒處於 WAITING 或 TIMED_WAITING 狀態的總次數。
lockname	字串統計資訊	監視鎖定的字串表示，該監視所定為執行緒被暫停而無法進入或執行緒等待通過 Object.wait 方法接收通知。

表 21-23 J2SE 5.0 的 JVM 統計資訊 — 執行緒資訊 (續)

統計資訊	資料類型	描述
lockownerid	計數統計資訊	包含某個物件的監視鎖定的執行緒 ID，該執行緒在該物件上發生區段化。
lockownername	字串統計資訊	包含某個物件的監視所定的執行緒名稱，該執行緒在該物件上發生區段化。
stacktrace	字串統計資訊	與該執行緒相關聯的堆疊追蹤。

表 21-24 中顯示了可用於 J2SE 5.0 的 JVM 中的執行緒的統計資訊。

表 21-24 J2SE 5.0 的 JVM 統計資訊 — 執行緒

統計資訊	資料類型	描述
threadcount	計數統計資訊	目前的活性常駐程式執行緒和非常駐程式執行緒數。
peakthreadcount	計數統計資訊	JVM 啟動或峰重設以來的峰活性執行緒計數。
totalstartedthreadcount	計數統計資訊	JVM 啟動以來建立和/或啟動的執行緒總數。
daemonthreadcount	計數統計資訊	目前的活性常駐程式執行緒數。
allthreadids	字串統計資訊	所有活性執行緒 ID 清單。
currentthreadcputime	計數統計資訊	如果已啟用 CPU 時間測量，則表示目前執行緒的 CPU 時間 (以納秒為單位)。如果已停用 CPU 時間測量，則傳回 -1。
monitordeadlockedthreads	字串統計資訊	處於監視死結狀態的執行緒 ID 清單。

生產 Web 容器 (PWC) 統計資訊

可用於 Application Server 的 Enterprise Edition (EE) 上的以下 PWC 元件和服務的統計資訊：

- 表 21-25，PWC 虛擬伺服器
- 表 21-26，PWC 請求
- 表 21-27，PWC 檔案快取
- 表 21-28，PWC 持續作用
- 表 21-29，PWC DNS
- 表 21-30，PWC 執行緒池
- 表 21-31，PWC 連線佇列
- 表 21-32，PWC HTTP 服務

表 21-25 中列示了 PWC 虛擬伺服器的統計資訊。

表 21-25 PWC 虛擬伺服器統計資訊 (僅限於 EE)

屬性名稱	資料類型	描述
id	字串 統計資訊	虛擬伺服器的 ID。
mode	字串 統計資訊	虛擬伺服器所處的模式。選項包括 unknown 或 active。
hosts	字串 統計資訊	由該虛擬伺服器提供服務的主機的名稱。
interfaces	字串 統計資訊	配置了虛擬伺服器的介面 (偵聽程式) 的類型。

表 21-26 中列示了可用於 PWC 請求的統計資訊。

表 21-26 PWC 請求統計資訊 (僅限於 EE)

屬性名稱	資料類型	描述
method	字串 統計資訊	用於請求的方法。
uri	字串 統計資訊	上一個被處理的 URI。
countrequests	計數 統計資訊	已處理的請求數。

表 21-26 PWC 請求統計資訊 (僅限於 EE)(續)

屬性名稱	資料類型	描述
countbytestransmitted	計數 統計資訊	傳輸的位元組數，如果此資訊不可用，則值為 0。
countbytesreceived	計數 統計資訊	接收的位元組數，如果此資訊不可用，則值為 0。
ratebytesreceived	計數 統計資訊	資料在某段伺服器定義的時間間隔內的傳輸速率，如果此資訊不可用，則值為 0。
maxbytestransmissionrate	計數 統計資訊	資料在某段伺服器定義的時間間隔內的最大傳輸速率，如果此資訊不可用，則值為 0。
countopenconnections	計數 統計資訊	目前開啟的連線數，如果此資訊不可用，則值為 0。
maxopenconnections	計數 統計資訊	同時開啟的連線的最大數目，如果此資訊不可用，則值為 0。
count2xx	計數 統計資訊	代碼為 2XX 的回應的總數。
count3xx	計數 統計資訊	代碼為 3XX 的回應的總數。
count4xx	計數 統計資訊	代碼為 4XX 的回應的總數。
count5xx	計數 統計資訊	代碼為 5XX 的回應的總數。
countother	計數 統計資訊	具有其他回應代碼的回應總數。
count200	計數 統計資訊	代碼為 200 的回應的總數。
count302	計數 統計資訊	代碼為 302 的回應的總數。
count304	計數 統計資訊	代碼為 304 的回應的總數。
count400	計數 統計資訊	代碼為 400 的回應的總數。
count401	計數 統計資訊	代碼為 401 的回應的總數。
count403	計數 統計資訊	代碼為 403 的回應的總數。
count404	計數 統計資訊	代碼為 404 的回應的總數。

表 21-26 PWC 請求統計資訊 (僅限於 EE)(續)

屬性名稱	資料類型	描述
count503	計數 統計資訊	代碼為 503 的回應的總數。

快取資訊小節提供了有關目前檔案快取的使用方式的資訊。表 21-27 中列示了 PWC 檔案快取的統計資訊。

表 21-27 PWC 檔案快取統計資訊 (僅限於 EE)

屬性名稱	資料類型	描述
flagenabled	計數 統計資訊	指示是否啟用了檔案快取。禁用時有效值為 0，啟用時有效值為 1。
secondsmaxage	計數 統計資訊	有效快取項目的最長存在時間 (以秒為單位)。
countentries	計數 統計資訊	目前的快取項目數。單一快取項目表示單一 URI。
maxentries	計數 統計資訊	同步式快取項目的最大數目。
countopenentries	計數 統計資訊	與開啟的檔案關聯的項目數。
maxopenentries	計數 統計資訊	與開啟的檔案關聯的同步式快取項目的最大數目。
sizeheapcache	計數 統計資訊	用於快取內容的堆疊儲存區空間。
maxheapcachesize	計數 統計資訊	用於快取檔案內容的最大堆疊儲存區空間。
sizemmapcache	計數 統計資訊	記憶體對映的檔案內容所使用的位址空間大小。
maxmmapcachesize	計數 統計資訊	檔案快取用於記憶體對映檔案內容的最大位址空間大小。
counthits	計數 統計資訊	快取查找成功的次數。
countmisses	計數 統計資訊	快取查找失敗的次數。
countinfohits	計數 統計資訊	檔案資訊查找成功的次數。
countinfomisses	計數 統計資訊	快取的檔案資訊遺失的數目。

表 21-27 PWC 檔案快取統計資訊 (僅限於 EE)(續)

屬性名稱	資料類型	描述
countcontenthits	計數 統計資訊	快取的檔案內容的命中次數。
countcontentmisses	計數 統計資訊	檔案資訊查找失敗的次數。

本小節提供有關伺服器的 HTTP 層級持續作用的系統的資訊。表 21-28 中列示了可用於 PWC 持續作用的統計資訊。

表 21-28 PWC 持續作用統計資訊 (僅限於 EE)

屬性名稱	資料類型	描述
countconnections	計數 統計資訊	處於持續作用模式的連線數。
maxconnections	計數 統計資訊	允許同步處於持續作用模式的最大連線數。
counthits	計數 統計資訊	處於持續作用模式的連線隨後進行了有效請求的總次數。
countflushes	計數 統計資訊	伺服器關閉持續作用的連線的次數。
countrefusals	計數 統計資訊	可能由於有太多的持續性連線而使伺服器無法將連線傳送到持續作用執行緒的次數。
counttimeouts	計數 統計資訊	伺服器因用戶端連線逾時且沒有任何活動而終止持續作用連線的次數。
secondstimeout	計數 統計資訊	關閉閒置持續作用連線之前經歷的時間 (以秒為單位)。

DNS 快取快取 IP 位址和 DNS 名稱。依預設，伺服器的 DNS 快取處於停用狀態。單一快取項目表示單一 IP 位址或 DNS 名稱查找。表 21-29 中列示了可用於 PWC DNS 的統計資訊。

表 21-29 PWC DNS 統計資訊 (僅限於 EE)

屬性名稱	資料類型	描述
flagcacheenabled	計數 統計資訊	指示是否啟用了 DNS 快取。禁用時為 0，啟用時為 1。
countcacheentries	計數 統計資訊	目前位於快取中的 DNS 項目數。

表 21-29 PWC DNS 統計資訊 (僅限於 EE)(續)

屬性名稱	資料類型	描述
maxcacheentries	計數 統計資訊	快取中可容納的 DNS 項目的最大數目。
countcachehits	計數 統計資訊	DNS 快取查找成功的次數。
countcachemisses	計數 統計資訊	DNS 快取查找失敗的次數。
flagasyncenabled	計數 統計資訊	指示是否啟用了非同步 DNS 查找。禁用時為 0，啟用時為 1。
countasyncnamelookups	計數 統計資訊	非同步 DNS 名稱查找的總數。
countasyncaddrlookups	計數 統計資訊	非同步 DNS 位址查找的總數。
countasynclookupsinprogress	計數 統計資訊	正在進行的非同步查找的數目。

表 21-30 中列示了 PWC 執行緒池的統計資訊。

表 21-30 PWC 執行緒池統計資訊 (僅限於 EE)

屬性名稱	資料類型	描述
id	字串 統計資訊	執行緒池 ID。
countthreadsidle	計數 統計資訊	目前處於閒置狀態的請求處理執行緒數。
countthreads	計數 統計資訊	目前的請求處理執行緒的數目。
maxthreads	計數 統計資訊	可同時存在的請求處理執行緒的最大數目。
countqueued	計數 統計資訊	佇列等候由此執行緒池進行處理的請求數。
peakqueued	計數 統計資訊	佇列中同步容納的最大請求數。
maxqueued	計數 統計資訊	佇列一次可容納的最大請求數。

連線佇列是請求被處理前保存這些請求的佇列。連線佇列的統計資訊顯示佇列中的階段作業數以及連線被接受前的平均延遲。表 21-31 中列示了 PWC 連線佇列的統計資訊。

表 21-31 PWC 連線佇列統計資訊 (僅限於 EE)

屬性名稱	資料類型	描述
id	字串 統計資訊	連線佇列的 ID。
counttotalconnections	計數 統計資訊	已接受的連線總數。
countqueued	計數 統計資訊	目前位於佇列中的連線數。
peakqueued	計數 統計資訊	佇列中同步容納的最大連線數。
maxqueued	計數 統計資訊	連線佇列的最大大小。
countoverflows	計數 統計資訊	佇列太滿而無法容納連線的次數。
counttotalqueued	計數 統計資訊	已佇列的連線總數。某個給定連線可能會被多次佇列，因此 counttotalqueued 可能大於或等於 counttotalconnections。
tickstotalqueued	計數 統計資訊	連線在佇列中所花費的週期總數。週期是由系統決定的時間單位。
countqueued1minuteaverage	計數 統計資訊	前 1 分鐘內處於佇列狀態的平均連線數。
countqueued5minuteaverage	計數 統計資訊	前 5 分鐘內處於佇列狀態的平均連線數。
countqueued15minuteaverage	計數 統計資訊	前 15 分鐘內處於佇列狀態的平均連線數。

表 21-32 中列示了 PWC HTTP 服務的統計資訊。

表 21-32 PWC HTTP 服務統計資訊 (僅限於 EE)

屬性名稱	資料類型	描述
id	字串 統計資訊	HTTP 服務的實例名稱。
versionserver	字串 統計資訊	HTTP 服務的版本編號。

表 21-32 PWC HTTP 服務統計資訊 (僅限於 EE)(續)

屬性名稱	資料類型	描述
timestarted	字串 統計資訊	啟動 HTTP 服務的時間 (GMT)。
secondsrunning	計數 統計資訊	HTTP 服務啟動以來所經歷的時間 (以秒為單位)。
maxthreads	計數 統計資訊	每個實例中的最大工作者執行緒數。
maxvirtualservers	計數 統計資訊	每個實例中可以配置的最大虛擬伺服器數目。
flagprofilingenabled	計數 統計資訊	是否啟用了 HTTP 服務效能設定檔。有效值為 0 或 1。
flagvirtualserveroverflow	計數 統計資訊	指示是否配置了多於 maxvirtualservers 的虛擬伺服器。如果設定為 1，則不會為所有的虛擬伺服器追蹤統計資料。
load1minuteaverage	計數 統計資訊	前 1 分鐘內請求的平均負載。
load5minuteaverage	計數 統計資訊	前 5 分鐘內請求的平均負載。
load15minuteaverage	計數 統計資訊	前 15 分鐘內請求的平均負載。
ratebytestransmitted	計數 統計資訊	在某段伺服器定義的時間間隔內資料的傳輸速率。如果此資訊不可用，則結果為 0。
ratebytesreceived	計數 統計資訊	在某段伺服器定義的時間間隔內資料的接收速率。如果此資訊不可用，則結果為 0。

有關啟用和停用監視功能的管理主控台作業

- 使用管理主控台配置監視層級
- 使用 asadmin 工具配置監視功能

使用管理主控台配置監視層級

1. 存取 [監視服務] 頁面。若要完成此操作，請執行以下步驟：
   a. 在樹中，展開 [配置] 節點。
   b. 在樹中，展開要配置爲用於監視功能的伺服器實例的節點，例如 `server-config`。
   c. 在樹中，選取 [監視]。
2. 在 [監視服務] 頁面中，從要變更其監視層級的元件或服務相對的組合方塊中選擇適當的值。

   依預設，除始終可監視的 Java 虛擬機器 (JVM) 之外，其他所有元件和服務的監視功能都是關閉的。若要啓用監視功能，請在組合方塊中選取 [低] 或 [高]。若要關閉監視功能，請在組合方塊中選取 [關閉]。可以開啓或關閉以下元件和服務的監視功能：

   - **JVM** — 將此選項的監視層級設定爲「低」以監視 Java 虛擬機器。
   - **HTTP 服務** — 將此選項的監視層級設定爲 LOW 以監視所有 HTTP 偵聽程式和虛擬伺服器。
   - **作業事件服務** — 將此選項的監視層級設定爲 LOW 以監視任意作業事件子系統。
   - **JMS/連接器服務** — 將此選項的監視層級設定爲 LOW 以監視任意 Java 訊息服務 (JMS)。
   - **ORB** — 將此選項的監視層級設定爲 LOW 以監視 Application Server 核心及其連線管理程式使用的系統 ORB。
   - **Web 容器** — 將此選項的監視層級設定爲 LOW 以監視所有部署的 Servlet。
   - **EJB 容器** — 將此選項的監視層級設定爲 LOW 以監視所有部署的 EJB、EJB 池和 EJB 快取。將此方法設定爲 HIGH 還可以監視 EJB 商業方法。
   - **JDBC 連線池** — 將此選項的監視層級設定爲 LOW 以監視所有 JDBC 連線池。
   - **執行緒池** — 將此選項的監視層級設定爲 LOW 以監視所有執行緒池。

3. 按一下 [儲存]。

此版本中沒有 [附加監視服務特性]，因此請忽略 [附加特性] 表。

等效的 asadmin 指令爲：set，例如，若要開啓對 HTTP 服務的監視功能，請使用以下 asadmin 指令：

```
asadmin> set --user admin_user
server.monitoring-service.module-monitoring-levels.http-service=LOW
```

使用 asadmin 工具配置監視功能

若要關閉監視功能，或設定監視元件或服務的層級，您可以按照「使用管理主控台配置監視層級」中的說明使用管理主控台，也可以按照本小節中的說明使用 asadmin 工具。

1. 使用 get 指令可以查找目前已對哪些服務和元件啓用了監視功能：

   ```
   asadmin> get --user admin_user
   server.monitoring-service.module-monitoring-levels.*
   ```

   傳回：

   ```
   server.monitoring-service.module-monitoring-levels.
   connector-connection-pool = OFF
   server.monitoring-service.module-monitoring-levels.
   connector-service = OFF
   server.monitoring-service.module-monitoring-levels.ejb-container = OFF
   server.monitoring-service.module-monitoring-levels.http-service = OFF
   server.monitoring-service.module-monitoring-levels.jdbc-connection-pool
   = OFF
   server.monitoring-service.module-monitoring-levels.jms-service = OFF
   server.monitoring-service.module-monitoring-levels.jvm = OFF
   server.monitoring-service.module-monitoring-levels.orb = OFF
   server.monitoring-service.module-monitoring-levels.thread-pool = OFF
   server.monitoring-service.module-monitoring-levels.transaction-service
   = OFF
   server.monitoring-service.module-monitoring-levels.web-container = OFF
   ```

2. 使用 set 指令可以啓用監視功能。

   例如，若要啓用對 HTTP 服務的監視功能，請輸入以下指令：

   ```
   asadmin> set --user admin_user
   server.monitoring-service.module-monitoring-levels.http-service=LOW
   ```

   若要停用監視功能，請使用 set 指令並將監視層級指定爲 OFF。

有關檢視監視資料的管理主控台作業

- 在管理主控台中檢視監視資料
- 使用 asadmin 工具檢視監視資料

在管理主控台中檢視監視資料

若要使用 Application Server 管理主控台檢視伺服器實例中部署的元件或服務的監視資料，請執行以下步驟。如需有關每個元件或服務的屬性的更多描述，請參閱 「「關於受監視的元件和服務的統計資訊」。

1. 存取 [監視] 頁面。若要完成此操作，請執行以下步驟：
   a. 在樹形元件中，展開 [獨立實例] 節點，例如 `server (Admin Server)`。
   b. 從清單中選取一個獨立伺服器實例。
   c. 選取 [監視] 頁面。
   d. 在 [監視] 頁面上選取 [監視] 標籤。
2. 在 [檢視] 清單中，選取已部署到伺服器實例並已爲其啓用了監視功能的元件或服務。

   所選取元件或服務的監視資料顯示在 [檢視] 欄位下。如需有關可監視特性的描述，請參閱 「「關於受監視的元件和服務的統計資訊」。

   如果已啓用 JVM、伺服器、執行緒池、HTTP 服務和作業事件服務的監視功能，則可以在此頁面中檢視這些元件和服務的監視資料。「關於可監視物件的樹狀結構」中顯示了說明這些元件和服務的組織方式的圖解。
3. 如果在此清單中沒有看到要監視的元件或服務，請選取 [配置監視] 連結來啓用和停用所選取元件和服務的監視功能。選取 [關閉] 以停用對元件或服務的監視功能。選取 [LOW] 或 [HIGH] 以啓用對元件或服務的監視功能。

   如需有關啓用和停用監視功能的更多資訊，請參閱 「使用管理主控台配置監視層級」或「使用 asadmin 工具配置監視功能」。

4. 選取 [監視] 頁面的 [**應用程式**] 標籤可以檢視已部署到伺服器實例並已爲其啓用了監視功能的應用程式元件的監視資料。從 [應用程式] 清單中選取應用程式。從 [元件] 清單中選取特定元件。

   如果未顯示應用程式或元件的監視資料，請選取 [配置監視] 連結來啓用或停用元件或服務的監視功能。若要監視應用程式，請開啓執行這些應用程式的容器：例如，對於 Web 應用程式的 Web 容器和/或對於 EJB 應用程式的 EJB 容器，請選取 [Low] 或 [High]。

   如果未顯示應用程式的監視資料，則該應用程式很可能不存在或未運行。僅當應用程式存在、其監視功能已啓用且正在運行時，應用程式監視資料才可用。執行了應用程式後，將在監視註冊中註冊該應用程式並顯示其監視資料。

   使用管理主控台監視遠端應用程式和實例。遠端實例必須正在執行並已設定配置才能執行此操作。

   所選取元件的監視資料顯示在所選取元件下面。如需有關可監視特性的描述，請參閱 「「關於受監視的元件和服務的統計資訊」。可以在 「關於可監視物件的樹狀結構」中檢視到用於說明這些元件和服務如何針對應用程式進行組織的圖解。

5. 選取 [**資源**] 頁面可以檢視已部署到伺服器實例中並已對其啓用監視功能的資源的監視資料。從 [檢視] 清單中選取資源。如果未顯示您要檢視其監視資料的資源，請選取 [配置監視] 連結來啓用或停用資源的監視功能。

   如果未顯示資源的監視資料，則該資源很可能不存在或未運行。僅當資源存在、其監視功能已在 HIGH 層級啓用且其正在運行的情況下，該資源的監視資料才可用。例如，如果您已建立一個 JDBC 連接器服務，但使用該連接器服務的應用程式尚未從服務請求連接器，則說明該服務尚未建立。因而，不存在任何服務，也沒有任何可用的監視資料。JDBC 應用程式執行並且從服務請求連接器之後，將在監視註冊中註冊該服務並顯示其監視資料。

   所選取元件或服務的監視資料顯示在 [檢視] 欄位下。如需有關可監視特性的描述，請參閱 「「關於受監視的元件和服務的統計資訊」。可以在 「關於可監視物件的樹狀結構」中檢視到用於說明這些元件和服務如何針對資源進行組織的圖解。

6. 選取 [**作業事件**] 頁面以凍結作業事件子系統，進而轉返作業事件並確定凍結時正在進行的作業事件。若要啓用作業事件服務的監視功能，請選取 [配置監視] 連結並確定將 [作業事件服務] 設定爲 LOW。若要凍結作業事件服務以便轉返作業事件，請選取 [凍結]。若要轉返作業事件，請選取作業事件旁邊的核取方塊，並按一下 [轉返]。

等效的 `asadmin` 指令爲：`get --monitor`，例如，若要檢視 JVM 的監視資料，請使用以下 `asadmin` 指令：

```
asadmin> get --monitor server.jvm.*
```

使用 asadmin 工具檢視監視資料

- 使用 asadmin 工具檢視監視資料
- 瞭解和指定含點名稱
- list 和 get 指令的範例
- Petstore 範例
- list 和 get 指令在所有層級上的預期輸出

使用 asadmin 工具檢視監視資料

若要使用 asadmin 工具檢視監視資料，請使用後跟可監視物件的含點名稱的 asadmin list 和 asadmin get 指令。使用 asadmin 工具檢視監視資料的一般方法如下：

1. 若要檢視可監視物件的名稱，請使用 asadmin list 指令。例如，若要檢視伺服器實例上已啓用監視功能的應用程式元件和子系統的清單，請在終端機視窗中鍵入以下指令：

   ```
   asadmin> list --monitor server
   ```

   上述指令將傳回已啓用監視功能的應用程式元件和子系統的清單，例如：

   ```
   server.resources
   server.connector-service
   server.orb
   server.jms-service
   server.jvm
   server.applications
   server.http-service
   server.thread-pools
   ```

   如需有關使用 list 指令的更多範例，請參閱「list 和 get 指令的範例」。如需有關可與 list 指令一起使用的含點名稱的更多資訊，請參閱「瞭解和指定含點名稱」。

2. 若要顯示已啓用監視功能的應用程式元件或子系統的監視統計資訊，請使用 asadmin get 指令。若要獲得統計資訊，請在終端機視窗中鍵入 asadmin get 指令，並指定在先前步驟中由 list 指令顯示的名稱。以下範例嘗試獲取某個特定物件的子系統的所有屬性：

   ```
   asadmin> get --monitor server.jvm.*
   ```

此指令傳回以下屬性和資料：

```
server.jvm.dotted-name = server.jvm
server.jvm.heapsize-current = 21241856
server.jvm.heapsize-description = Provides statistical information
about the JVM's memory heap size.
server.jvm.heapsize-highwatermark = 21241856
server.jvm.heapsize-lastsampletime = 1080232913938
server.jvm.heapsize-lowerbound = 0
server.jvm.heapsize-lowwatermark = 0
server.jvm.heapsize-name = JvmHeapSize
server.jvm.heapsize-starttime = 1080234457308
server.jvm.heapsize-unit = bytes
server.jvm.heapsize-upperbound = 518979584
server.jvm.uptime-count = 1080234457308
server.jvm.uptime-description = Provides the amount of time the JVM has
been running.
server.jvm.uptime-lastsampletime = 1080234457308
server.jvm.uptime-name = JvmUpTime
server.jvm.uptime-starttime = 1080232913928
server.jvm.uptime-unit = milliseconds
```

Sun Java System Application Server Enterprise Edition 8.1 2005Q1 如需有關使用 get 指令的更多範例，請參閱「list 和 get 指令的範例」。如需有關可與 get 指令一起使用的含點名稱的更多資訊，請參閱「瞭解和指定含點名稱」。

瞭解和指定含點名稱

在 asadmin list 和 get 指令中，指定可監視物件的含點名稱。所有子物件都是使用點 (.) 字元作爲分隔符來命名的，因而這些子物件的名稱也稱爲**含點名稱**。如果子節點爲單一型，則僅需要監視物件類型來命名物件；否則，需要形式爲 type.name 的名稱來命名物件。

例如，http-service 就是其中一種有效的可監視物件類型，並且爲單一型。若要命名表示實例 server 的 http-service 的單一型子節點，則含點名稱爲：

```
server.http-service
```

另一個範例，application 爲一種有效的可監視物件類型，但並非單一型。例如，若要命名表示應用程式 Petstore 的非單一型子節點，則含點名稱爲：

```
server.applications.petstore
```

含點名稱也可以命名可監視物件中的特定屬性。例如，http-service 具有名爲 bytesreceived-lastsampletime 的可監視屬性。以下名稱可以命名 bytesreceived 屬性：

```
server.http-service.server.http-listener-1.
  bytesreceived-lastsampletime
```

管理員不需要知道 asadmin list 和 get 指令的有效含點名稱。使用 list 指令可以顯示可用的可監視物件，而使用帶有萬用字元參數的 get 指令可以檢查任意可監視物件的所有可用屬性。

使用具有含點名稱的 list 和 get 指令的基本假設爲：

- 使用任何具有含點名稱且後面不跟萬用字元 (*) 的 list 指令，得到的結果爲目前節點的直接子節點。例如，list --monitor server 列出了屬於 server 節點的所有直接子節點。
- 使用任何具有含點名稱且後跟 .* 形式的萬用字元的 list 指令，得到的結果爲目前節點的子節點階層樹。例如，list --monitor server.applications.* 列示了 applications 的所有子節點及其後續子節點等。
- 使用任何具有含點名稱並且前面或後面帶有 **dottedname* 或 *dotted* name* 或 **dotted name *** 形式的萬用字元的 list 指令，得到的結果爲所有節點以及匹配常規表示式 (由提供的匹配式樣所建立) 的子節點。
- 使用後跟 .* 或 * 的 get 指令，得到的結果爲一組屬性及其值，它們屬於目前要匹配的節點。

如需更多資訊，請參閱 「list 和 get 指令在所有層級上的預期輸出」。

list 和 get 指令的範例

本小節包含下列主題：

- list --monitor 指令的範例
- get --monitor 指令的範例
- Petstore 範例

list --monitor 指令的範例

list 指令提供有關目前監視的指定伺服器實例名稱的應用程式元件與子系統的資訊。使用該指令，您可以檢視伺服器實例的可監視元件與子元件。如需有關 list 範例的更完整的清單，請參閱 「list 和 get 指令在所有層級上的預期輸出」。

範例 *1*

```
asadmin> list --monitor server
```

上述指令將傳回已啓用監視功能的應用程式元件和子系統的清單，例如：

```
server.resources
server.orb
server.jvm
server.jms-service
server.connector-service
server.applications
server.http-service
server.thread-pools
```

還可以列示指定的伺服器實例中目前所監視的應用程式。這對於使用 get 指令從某個應用程式中獲取特定的監視統計資訊很有幫助。

範例 *2*

```
asadmin> list --monitor server.applications
```

傳回：

```
server.applications.adminapp
server.applications.admingui
server.applications.myApp
```

如需更爲複雜的範例，請參閱 「Petstore 範例」。

get --monitor 指令的範例

該指令擷取下列受監視資訊：

- 一個元件或子系統內的全部受監視屬性
- 一個元件或子系統內特定的受監視屬性

當特定元件或子系統所請求的屬性不存在時，便會傳回錯誤。同樣，當元件或子系統所請求的特定屬性不在作用中時，也會傳回錯誤。

請參閱 「list 和 get 指令在所有層級上的預期輸出」，以取得有關使用 get 指令的更多資訊。

範例 *1*

嘗試從某個子系統中獲取某個特定物件的所有屬性：

```
asadmin> get --monitor server.jvm.*
```

傳回：

```
server.jvm.dotted-name = server.jvm
server.jvm.heapsize-current = 21241856
server.jvm.heapsize-description = Provides statistical information
about the JVM's memory heap size.
server.jvm.heapsize-highwatermark = 21241856
server.jvm.heapsize-lastsampletime = 1080232913938
server.jvm.heapsize-lowerbound = 0
server.jvm.heapsize-lowwatermark = 0
server.jvm.heapsize-name = JvmHeapSize
server.jvm.heapsize-starttime = 1080234457308
server.jvm.heapsize-unit = bytes
server.jvm.heapsize-upperbound = 518979584
server.jvm.uptime-count = 1080234457308
server.jvm.uptime-description = Provides the amount of time the JVM has
been running.
server.jvm.uptime-lastsampletime = 1080234457308
server.jvm.uptime-name = JvmUpTime
server.jvm.uptime-starttime = 1080232913928
server.jvm.uptime-unit = milliseconds
```

範例 *2*

嘗試從 J2EE 應用程式中取得全部屬性：

```
asadmin> get --monitor server.applications.myJ2eeApp.*
```

傳回：

```
No matches resulted from the wildcard expression.
CLI137 Command get failed.
```

該 J2EE 應用程式層級上沒有提供可監視的屬性，因而顯示此回覆。

範例 *3*

嘗試從某個子系統中取得特定屬性：

```
asadmin> get --monitor server.jvm.uptime-lastsampletime
```

傳回：

```
server.jvm.uptime-lastsampletime = 1093215374813
```

範例 *4*

嘗試從某個子系統屬性中取得不明的屬性：

```
asadmin> get --monitor server.jvm.badname
```

傳回：

```
No such attribute found from reflecting the corresponding Stats
interface:[badname]
CLI137 Command get failed.
```

Petstore 範例

以下範例說明如何將 asadmin 工具用於監視目的。

使用者要檢查在將範例 Petstore 應用程式部署到 Application Server 後在該應用程式中呼叫某個方法的次數。部署該應用程式的實例名稱爲 server。配合使用 list 與 get 指令來存取所需的方法統計資訊。

1. 啓動 Application Server 和 asadmin 工具。
2. 設定一些有用的環境變數，以避免使用每個指令時都輸入這些變數：

   ```
   asadmin>export AS_ADMIN_USER=admin AS_ADMIN_PASSWORD=admin123
   ```

   ```
   asadmin>export AS_ADMIN_HOST=localhost AS_ADMIN_PORT=4848
   ```

3. 列示實例 server 的可監視元件：

   ```
   asadmin>list --monitor server*
   ```

   傳回 (輸出類似於以下內容)：

   ```
   server
   server.applications
   server.applications.CometEJB
   server.applications.ConverterApp
   server.applications.petstore
   server.http-service
   server.resources
   server.thread-pools
   ```

   可監視元件清單包括 thread-pools、http-service、resources 以及所有已部署 (並已啓用) 的 applications。

4. 列示 Petstore 應用程式中的可監視子元件 (可以使用 -m 代替 --monitor)：

   ```
   asadmin>list -m server.applications.petstore
   ```

   傳回：

   ```
   server.applications.petstore.signon-ejb_jar
   server.applications.petstore.catalog-ejb_jar
   server.applications.petstore.uidgen-ejb_jar
   server.applications.petstore.customer-ejb_jar
   ```

```
server.applications.petstore.petstore-ejb_jar
server.applications.petstore.petstore\.war
server.applications.petstore.AsyncSenderJAR_jar
server.applications.petstore.cart-ejb_jar
```

5. 列示 Petstore 應用程式的 EJB 模組 `signon-ejb_jar` 中的可監視子元件：

   ```
   asadmin>list -m server.applications.petstore.signon-ejb_jar
   ```

   傳回：

   ```
   server.applications.petstore.signon-ejb_jar.SignOnEJB
   server.applications.petstore.signon-ejb_jar.UserEJB
   ```

6. 列示 Petstore 應用程式的 EJB 模組 `signon-ejb_jar` 的實體 Bean `UserEJB` 中的可監視子元件：

   ```
   asadmin>list -m server.applications.petstore.signon-ejb_jar.UserEJB
   ```

   傳回 (出於空間考量，移除了含點名稱)：

   ```
   server.applications.petstore.signon-ejb_jar.UserEJB.bean-cache
   server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods
   server.applications.petstore.signon-ejb_jar.UserEJB.bean-pool
   ```

7. 列示實體 Bean `UserEJB` (位於 Petstore 應用程式的 EJB 模組 `signon-ejb_jar` 中) 的方法 `getUserName` 中的可監視子元件：

   ```
   asadmin>list  -m server.applications.petstore.signon-ejb_jar.
   UserEJB.bean-methods.getUserName
   ```

   傳回：

   ```
   Nothing to list at server.applications.petstore.signon-ejb_jar.
   UserEJB.bean-methods.getUserName. To get the valid names beginning with
   a string, use the wildcard "*" character.For example, to list all names
   that begin with "server", use "list server*".
   ```

8. 該方法沒有可監視的子元件。獲取方法 `getUserName` 的所有可監視統計資訊。

   ```
   asadmin>get -m server.applications.petstore.signon-ejb_jar.
   UserEJB.bean-methods.getUserName.*
   ```

   傳回：

   ```
   server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
   getUserName.executiontime-count = 0
   server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
   getUserName.executiontime-description = Provides the time in
   milliseconds spent during the last successful/unsuccessful attempt
   to execute the operation.
   server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
   ```

```
getUserName.executiontime-lastsampletime = 1079981809259
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.executiontime-name = ExecutionTime
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.executiontime-starttime = 1079980593137
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.executiontime-unit = count
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-count = 0
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-description = Provides the number of times
an operation was called, the total time that was spent during the
invocation and so on.
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-lastsampletime = 1079980593137
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-maxtime = 0
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-mintime = 0
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-name = ExecutionTime
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-starttime = 1079980593137
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-totaltime = 0
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.methodstatistic-unit =
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumerrors-count = 0
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumerrors-description = Provides the total number of
errors that occured during invocation or execution of an operation.
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumerrors-lastsampletime = 1079981809273
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumerrors-name = TotalNumErrors
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumerrors-starttime = 1079980593137
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumerrors-unit = count
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumsuccess-count = 0
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumsuccess-description = Provides the total number of
successful invocations of the method.
```

```
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumsuccess-lastsampletime = 1079981809255
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumsuccess-name = TotalNumSuccess
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumsuccess-starttime = 1079980593137
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.totalnumsuccess-unit = count
```

9. 如果還需要獲取執行時間等特定統計資訊，請使用如下指令：

```
asadmin>get -m server.applications.petstore.signon-ejb_jar.
UserEJB.bean-methods.getUserName.executiontime-count
```

傳回：

```
server.applications.petstore.signon-ejb_jar.UserEJB.bean-methods.
getUserName.executiontime-count = 1
```

list 和 get 指令在所有層級上的預期輸出

下表顯示了樹的各個層級的指令、含點名稱及相應的輸出。

表 21-33 頂層

指令	含點名稱	輸出

`list -m`	`server`	`server.applications`
		`server.thread-pools`
		`server.resources`
		`server.http-service`
		`server.transaction-service`
		`server.orb.connection-managers`
		`server.orb.connection-managers.`
		`orb\.Connections\.Inbound\.`
		`AcceptedConnections`
		`server.jvm`

`list -m`	`server.*`	此節點下的子節點的階層結構。
`get -m`	`server.*`	僅顯示一條訊息，說明此節點上沒有屬性。

表 21-34 顯示了應用程式層級的指令、含點名稱及相應的輸出。

表 21-34 應用程式層級

指令	含點名稱	輸出
list -m	server.applications 或 *applications	appl1 app2 web-module1_war ejb-module2_jar ...
list -m	server.applications.* 或 *applications.*	此節點下的子節點的階層結構。
get -m	server.applications.* 或 *applications.*	僅顯示一條訊息，說明此節點上沒有屬性。

表 21-35 顯示了應用程式層級上的獨立模組和企業應用程式的指令、含點名稱及相應的輸出。

表 21-35 應用程式 — 企業應用程式和獨立模組

指令	含點名稱	輸出
list -m	server.applications.app1 或 *app1 **備註：僅當已部署了企業應用程式時，此層級才可用。如果部署了獨立模組，則此層級不可用。**	ejb-module1_jar web-module2_war ejb-module3_jar web-module3_war ...
list -m	server.applications.app1.* 或 *app1.*	此節點下的子節點的階層結構。
get -m	server.applications.app1.* 或 *app1.*	僅顯示一條訊息，說明此節點上沒有屬性。
list -m	server.applications.app1.ejb-module1_jar 或 *ejb-module1_jar 或 server.applications.ejb-module1_jar	bean1 bean2 bean3 ...

表 21-35 應用程式 — 企業應用程式和獨立模組 (續)

指令	含點名稱	輸出

`list -m`	`server.applications.app1.ejb-module1_jar`
	或
	`*ejb-module1_jar`
	或
	`server.applications.ejb-module1_jar`
		此節點下的子節點的階層結構。

`get -m`	`server.applications.app1.ejb-module1_jar.*`
	或
	`*ejb-module1_jar.*`
	或
	`server.applications.ejb-module1_jar.*`
		僅顯示一條訊息，說明此節點上沒有屬性。

`list -m`	`server.applications.app1.ejb-module1_jar.bean1`
	備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。
		子節點清單：
		`bean-pool`
		`bean-cache`
		`bean-method`

`list -m`	`server.applications.app1.ejb-module1_jar.bean1` **備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。**	子節點的階層結構及該節點和所有後續子節點的所有屬性的清單。

`get -m`	`server.applications.app1.ejb-module1_jar.bean1.*`
	備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。
		以下屬性及其關聯值：
		`CreateCount_Count`
		`CreateCount_Description`
		`CreateCount_LastSampleTime`
		`CreateCount_Name`
		`CreateCount_StartTime`
		`CreateCount_Unit`
		`MethodReadyCount_Current`
		`MethodReadyCount_Description`
		`MethodReadyCount_HighWaterMark`
		`MethodReadyCount_LastSampleTime`
		`MethodReadyCount_LowWaterMark`
		`MethodReadyCount_Name`
		`MethodReadyCount_StartTime`
		`MethodReadyCount_Unit`
		`RemoveCount_Count`
		`RemoveCount_Description`
		`RemoveCount_LastSampleTime`
		`RemoveCount_Name`
		`RemoveCount_StartTime`
		`Attribute RemoveCount_Unit`

表 21-35 應用程式 — 企業應用程式和獨立模組 (續)

指令	含點名稱	輸出
list -m	server.applications.app1.ejb-module1_jar.bean1 .bean-pool **備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。**	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
get -m	server.applications.app1.ejb-module1_jar.bean1 .bean-pool.* **備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。**	與 EJB 池屬性對應的屬性和值的清單 (如表 1-4 所示)。
list -m	server.applications.app1.ejb-module1_jar.bean1 .bean-cache **備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。**	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
get -m	server.applications.app1.ejb-module1_jar.bean1 .bean-cache.* **備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。**	與 EJB 快取屬性對應的屬性和值的清單 (如表 1-5 所示)。
list -m	server.applications.app1.ejb-module1_jar.bean1 .bean-method.method1 **備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。**	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
get -m	server.applications.app1.ejb-module1_jar.bean1 .bean-method.method1.* **備註：在獨立模組中，將不顯示包含應用程式名稱 (本範例中為 *app1*) 的節點。**	與 EJB 方法屬性對應的屬性和值的清單 (如表 1-2 所示)。
list -m	server.applications.app1.web-module1_war	顯示指定給模組的虛擬伺服器。
get -m	server.applications.app1.web-module1_war.*	僅顯示一條訊息，說明此節點上沒有屬性。
list -m	server.applications.app1.web-module1_war. virtual_server	顯示已註冊的 servlet 的清單。
get -m	server.applications.app1.web-module1_war. virtual_server.*	僅顯示一條訊息，說明此節點上沒有屬性。
list -m	server.applications.app1.web-module1_war. virtual_server.servlet1	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
get -m	server.applications.app1.web-module1_war. virtual_server.servlet1.*	與 Web 容器 (Servlet) 屬性對應的屬性和值的清單 (如表 1-7 所示)。

表 21-36 顯示了 HTTP 服務層級的指令、含點名稱及相應的輸出。

表 21-36 HTTP 服務層級

指令	含點名稱	輸出
`list -m`	`server.http-service`	虛擬伺服器清單。
`get -m`	`server.http-service.*`	僅顯示一條訊息，說明此節點上沒有屬性。
`list -m`	`server.http-service.server`	HTTP 偵聽程式清單。
`get -m`	`server.http-service.server.*`	僅顯示一條訊息，說明此節點上沒有屬性。
`list -m`	`server.http-service.server.http-listener1`	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
`get -m`	`server.http-service.server.*`	與 HTTP 服務屬性對應的屬性和值的清單 (如表 1-9 所示)。

表 21-37 顯示了執行緒池層級的指令、含點名稱及相應的輸出。

表 21-37 執行緒池層級

指令	含點名稱	輸出
`list -m`	`server.thread-pools`	執行緒池名稱清單。
`get -m`	`server.thread-pools.*`	僅顯示一條訊息，說明此節點上沒有屬性。
`list -m`	`server.thread-pools.orb\.threadpool\.thread-pool-1`	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
`get -m`	`server.thread-pools..orb\.threadpool\.thread-pool-1.*`	與執行緒池屬性對應的屬性和值的清單 (如表 1-14 所示)。

表 21-38 顯示了資源層級的指令、含點名稱及相應的輸出。

表 21-38 資源層級

指令	含點名稱	輸出
`list -m`	`server.resources`	池名稱清單。
`get -m`	`server.resources.*`	僅顯示一條訊息，說明此節點上沒有屬性。

表 21-38 資源層級 (續)

指令	含點名稱	輸出
`list -m`	`server.resources.jdbc-connection-pool -pool.connection-pool1`	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
`get -m`	`server.resources.jdbc-connection-pool -pool.connection-pool1.*`	與連線池屬性對應的屬性和值的清單 (如表 1-10 所示)。

表 21-39 顯示了作業事件服務層級的指令、含點名稱及相應的輸出。

表 21-39 作業事件服務層級

指令	含點名稱	輸出
`list -m`	`server.transaction-service`	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
`get -m`	`server.transaction-service.*`	與作業事件服務屬性對應的屬性和值的清單 (如表 1-15 所示)。

表 21-40 顯示了 ORB 層級的指令、含點名稱及相應的輸出。

表 21-40 ORB 層級

指令	含點名稱	輸出
`list -m`	`server.orb`	`server-orb.connection-managers`
`get -m`	`server.orb.*`	僅顯示一條訊息，說明此節點上沒有屬性。
`list -m`	`server.orb.connection-managers`	ORB 連線管理程式的名稱。
`get -m`	`server.orb.connection-managers.*`	僅顯示一條訊息，說明此節點上沒有屬性。
`list -m`	`server.orb.connection-managers. orb\.Connections\.Inbound\ .AcceptedConnections`	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
`get -m`	`server.orb.connection-managers. orb\.Connections\.Inbound\ .AcceptedConnections.*`	與 ORB 連線管理程式屬性對應的屬性和值的清單 (如表 1-13 所示)。

表 21-41 顯示了 JVM 層級的指令、含點名稱及相應的輸出。

表 21-41 JVM 層級

指令	含點名稱	輸出
`list -m`	`server.jvm`	沒有屬性，僅顯示一條訊息，說明「使用帶有 --monitor 選項的 get 指令可以檢視該節點的屬性和值。」
`get -m`	`server.jvm.*`	與 JVM 屬性對應的屬性和值的清單 (如表 1-16 所示)。

使用 JConsole

若要使 JConsole 能夠與 Application Server 一起使用，必須停用 JMX 連接器的安全性。依預設，目前版本的 Application Server (SE/EE 版本) 已啓用安全性。

若要停用 JMX 連接器的安全性，請使用以下技術之一：

1. 使用管理主控台停用 JMX 連接器的安全性。若要從管理主控台執行此操作，請執行以下步驟：
   a. 展開 [配置] 節點。
   b. 展開 [管理服務] 節點。
   c. 選取 [system] 節點。
   d. 在 [SSL] 區段取消選取 [SSL3] 和 [TLS]。
   e. 選取 [儲存]。
2. 使用 `asadmin` 停用 JMX 連接器的安全性。若要從終端機視窗或指令提示符號執行此操作，請執行以下步驟：
   a. 輸入以下指令：

   ```
   asadmin set
   server.admin-service.jmx-connector.system.security-enabled=false
   ```

   b. 重新啓動網域應用程式伺服器 (DAS)。

對於 PE 版本，依預設 JMX 連接器處於停用狀態，因此無需變更 PE 的任何配置。

3. 啓動 JConsole，然後在 [進階] 標籤中輸入 JMX URL、使用者名稱和密碼進行登入。JMX URL 的形式爲：

   `service:jmx:rmi:///jndi/rmi://`*< 您的機器名稱 >:< 連接埠 >*`/management/rmi-jmx-connector`

   備註：如果搜尋 `message ADM1501`，您可以從管理 `server.log` 檔案中獲得確切的 JMX URL。

第 22 章

Java 虛擬機器和進階設定

本章說明如何配置 Java 虛擬機器 (JVM™) 和其他進階設定。它包含以下小節：

- 有關 JVM™ 設定的管理主控台作業
- 有關進階設定的管理主控台作業

有關 JVM™ 設定的管理主控台作業

- 配置 JVM 一般設定
- 配置 JVM 類別路徑設定
- 配置 JVM 選項
- 停用安全性管理程式
- 配置 JVM 效能評測器設定

配置 JVM 一般設定

Java 虛擬機器 (JVM) 包括在 Java 2 Standard Edition (J2SE™) 軟體中，是 Application Server 所需的軟體。由於不正確的 JVM 設定將導致伺服器停止執行，因此您應當謹慎地變更這些設定。

若要配置 Application Server 所使用的 JVM 的一般設定，請執行以下步驟：

1. 在樹形元件中，選取 [應用程式伺服器] 節點。
2. 按一下 [JVM 設定] 標籤。
3. 依預設，標籤下的 [一般] 連結已處於選取狀態。

4. 在 [JVM 一般設定] 頁面中，您可以指定以下內容：

   a. 在 [Java 首頁] 欄位中，輸入 Java 2 Standard Edition (J2SE) 軟體的安裝目錄名稱。

      Application Server 依賴於 J2SE 軟體運行。若要驗證此版本是否支援您所指定的 J2SE 版本，請參閱「版本說明」。(請參閱更多資訊一節中的連結。)

      備註：如果輸入不存在的目錄名稱或輸入不受支援的 J2SE 軟體版本的安裝目錄名稱，則 Application Server 將無法啓動。

   b. 在 [Javac] 欄位中，鍵入 Java 程式設計語言編譯器的指令行選項。

      部署 EJB 元件後，Application Server 將執行編譯器。

   c. 若要通過 JPDA (Java 平台除錯程式架構) 來設定除錯，請選取 [啓用除錯] 核取方塊並在 [除錯選項] 欄位中指定選項。

      JPDA 由應用程式開發者使用。如需更多資訊，請參閱「Application Server Developer's Guide」的「Debugging J2EE Applications」一章。(有關指向此指南的連結，請參閱更多資訊。)

   d. 在 [RMI 編譯選項] 欄位中，鍵入 RMIC 編譯器的指令行選項。

      部署 EJB 元件後，Application Server 將執行 RMIC 編譯器。

   e. 在 [位元碼預處理程式] 欄位中，鍵入以逗號分隔的類別名稱清單。

      每個類別都必須實作 `com.sun.appserv.BytecodePreprocessor` 介面。將按指定次序呼叫這些類別。

      效能評測器等工具也許需要 [位元碼預處理程式] 欄位中的項目。效能評測器產生用於分析伺服器效能的資訊。如需有關效能評測的更多資訊，請參閱「Application Server Developerís Guide」中的「Debugging J2EE Applications」一章。

5. 按一下 [儲存]。

6. 重新啓動伺服器。

配置 JVM 類別路徑設定

類別路徑就是 JAR 檔案的清單，Java 執行階段環境將在此清單中搜尋類別和其他資源檔案。

若要配置 Application Server 的 JVM 類別路徑，請執行以下步驟：

1. 在樹形元件中，選取 [應用程式伺服器] 節點。

2. 按一下 [JVM 設定] 標籤。
3. 選取標籤下面的 [路徑設定] 連結。
4. 在 [JVM 類別路徑設定] 頁面中，您可以指定以下內容：
   a. 在 [環境類別路徑] 核取方塊中，保留預設選取以忽略 CLASSPATH 環境變數。

      對於程式設計中的指導而言，CLASSPATH 環境變數使用起來很方便，但是不建議將此變數用於企業環境。

   b. 若要檢視 Application Server 的類別路徑，請檢查 [伺服器類別路徑] 欄位中的唯讀內容。
   c. 若要將 JAR 檔案插入伺服器類別路徑的開頭，請在 [類別路徑字首] 欄位中輸入此檔案的完整路徑名稱。
   d. 若要將 JAR 檔案增加至伺服器類別路徑的結尾，請在 [類別路徑字尾] 欄位中輸入此檔案的完整路徑名稱。

      例如，假設您要指定資料庫驅動程式的 JAR 檔案。請參閱 整合 JDBC 驅動程式。

   e. 在 [原生庫路徑字首] 和 [原生庫路徑字尾] 欄位中，您可以將項目附加到原生庫路徑的開頭或結尾。

      原生程式庫路徑是伺服器的相對路徑 (用於其原生共用程式庫)、標準 JRE 原生程式庫路徑、Shell 環境設定 (UNIX 上的 LD_LIBRARY_PATH) 以及 [JVM 效能評測器設定] 頁面上指定的任何路徑的鏈結。
5. 按一下 [儲存]。
6. 重新啓動伺服器。

配置 JVM 選項

在 [JVM 選項] 頁面中，您可以爲執行 Application Server 的 Java 應用程式啓動程式 (java 工具) 指定選項。-D 選項用於指定 Application Server 的專用特性。

若要配置 JVM 選項，請執行以下步驟：

1. 在樹形元件中，選取 [應用程式伺服器] 節點。
2. 按一下 [JVM 設定] 標籤。
3. 選取標籤下面的 [JVM 選項] 連結。
4. 在 [JVM 選項] 頁面中，通過編輯 [值] 欄位來修改選項。

5. 若要增加選項，請執行以下步驟：
   a. 按一下 [增加 JVM 選項]。
   b. 在顯示的空白列中，在 [值] 欄位中鍵入資訊。
6. 若要移除選項，請執行以下步驟：
   a. 選取該選項旁邊的核取方塊。
   b. 按一下 [刪除]。
7. 按一下 [儲存]。
8. 重新啓動伺服器。

如需有關 JVM 選項的更多資訊，請參閱：

- http://java.sun.com/j2se/1.4.2/docs/tooldocs/tools.html
- http://java.sun.com/docs/hotspot/VMOptions.html

停用安全性管理程式

停用 Application Server 的安全性管理程式可以改善某些類型的應用程式的效能。即使安全性管理程式已停用，J2EE 授權和驗證功能仍可以起作用。您可以在開發環境中停用安全性管理程式，但不應在生產環境中停用安全性管理程式。

若要停用安全性管理程式，請執行以下步驟：

1. 移至管理主控台的 [JVM 選項] 頁面。

   若需說明，請參閱「配置 JVM 選項」。
2. 在 [JVM 選項] 頁面中，移除以下選項：

   ```
   -Djava.security.policy
   ```
3. 按一下 [儲存]。
4. 重新啓動伺服器。

配置 JVM 效能評測器設定

效能評測器工具可產生用於分析效能和識別潛在瓶頸的資料。

若要爲 Application Server 配置效能評測器設定，請執行以下步驟：

1. 在樹形元件中，選取 [應用程式伺服器] 節點。
2. 按一下 [JVM 設定] 標籤。
3. 選取標籤下面的 [效能評測器] 連結。
4. 您在 [JVM 效能評測器設定] 頁面中指定的資訊取決於您所使用的效能評測器產品。

   如需範例和說明，請參閱 「Application Server Developerís Guide」中的「Debugging J2EE Applications」一章。(有關指向此指南的連結，請參閱更多資訊。)
5. 按一下 [儲存]。
6. 重新啓動伺服器。

有關進階設定的管理主控台作業

- 設定進階網域屬性

設定進階網域屬性

1. 在樹形元件中，選取 [應用程式伺服器] 節點。
2. 選取 [進階] 標籤。
3. 在 [網域屬性] 頁面中，您可以執行以下操作：
   a. 在 [應用程式根] 欄位中，識別將在其中部署應用程式的完整目錄路徑。
   b. 在 [記錄根] 欄位中，指定保存伺服器實例記錄檔的目錄。
   c. 通常，您需要將 [語言環境] 欄位保留爲空，以使用主機的預設語言環境。

      語言環境是一個識別碼，用於指定特定的語言和區域組合。例如，美國英語的語言環境是 en_US，日語的語言環境是 ja_JP。爲使用非英語語言環境，Application Server 必須經過本地化，即必須將英語翻譯成其他語言。
4. 按一下 [儲存]。
5. 重新啓動伺服器。

編譯和配置 Apache Web Server

本附錄介紹如何編譯 Apache 源代碼和配置 Apache Web Server 的安裝以使用 Sun Java System Application Server 負載平衡器外掛程式。

請下載相應的 Apache 源代碼。有關 Sun Java System Application Server 支援的 Apache Web Server 的版本和平台的資訊，請參閱「Sun Java System Application Server 版本說明」。

本附錄包含下列主題：

- 最低需求
- 安裝 SSL 可識別 Apache

最低需求

本小節描述要成功編譯 Apache Web Server 以執行負載平衡器外掛程式應滿足的最低需求。必須編譯和建置 Apache 源代碼以使用 SSL 執行。

本小節包含下列主題：

- 適用於 Apache 1.3 的最低需求
- 適用於 Apache 2 的最低需求

適用於 Apache 1.3 的最低需求

有關 Microsoft Windows 平台的需求，請參閱：
http://httpd.apache.org/docs/windows.html#req
http://httpd.apache.org/docs/win_compiling.html

適用於其他平台的需求：

- openssl-0.9.7d (源代碼)
- mod_ssl-2.8.16-1.3.29 (源代碼)
- apache_1.3.29 (源代碼)
- gcc-3.3-sol9-sparc-local 套裝軟體 (用於 Solaris 9 SPARC/x86)。
- flex-2.5.4a-sol9-sparc-local 套裝軟體 (用於 Solaris 9 SPARC)
- flex-2.5.4a-sol9-intel-local 套裝軟體 (用於 Solaris 9 x86)

此外，在編譯 Apache 之前，請執行以下步驟：

- 在 Linux 上，在同一台機器上安裝 Sun Java System Application Server。
- 在 Solaris 8 上，確定 gcc 和 make 位於 PATH 中。
- 在 Solaris 9 上，確定 gcc 版本 3.3 和 make 位於 PATH 中，並且已安裝 flex。
- 如果您在 Red Hat Enterprise Linux Advanced Server 2.1 上使用 gcc，則 gcc 的版本必須在 3.0 以上。

備註

- 要使用其他 C 編譯器，請設定 C 編譯器的路徑並使公用程式位於 PATH 環境變數中。例如：

  exportLD_LIBRARY_PATH=$LD_LIBRARY_PATH:appserver_installdir/lib

- 這些軟體源可以從 http://www.sunfreeware.com 獲得

適用於 Apache 2 的最低需求

有關 Microsoft Windows 平台的需求，請參閱：

http://httpd.apache.org/docs-2.0/platform/windows.html

適用於其他平台的需求：

- openssl-0.9.7d (源代碼)
- httpd-2.0.49 (源代碼)
- gcc-3.3-sol9-sparc-local 套裝軟體 (用於 Solaris 9 SPARC/x86)。
- flex-2.5.4a-sol9-sparc-local 套裝軟體 (用於 Solaris 9 SPARC)
- flex-2.5.4a-sol9-intel-local 套裝軟體 (用於 Solaris 9 x86)

此外，在編譯 Apache 之前，請執行以下步驟：

- 在 Linux 上，在同一台機器上安裝 Sun Java System Application Server。
- 在 Solaris 8 上，確定 gcc 和 make 位於 PATH 中。
- 在 Solaris 9 上，確定 gcc 版本 3.3 和 make 位於 PATH 中，並且已安裝 flex。
- 如果您在 Red Hat Enterprise Linux Advanced Server 2.1 上使用 gcc，則 gcc 的版本必須在 3.0 以上。

備註

- 要使用其他 C 編譯器，請設定 C 編譯器的路徑並使公用程式位於 PATH 環境變數中。例如：

  ```
  export LD_LIBRARY_PATH=app_server_install_dir/lib:$LD_LIBRARY_PATH
  ```

  該範例是針對 sh 的。
- 這些軟體源可以從 http://www.sunfreeware.com 獲得

安裝 SSL 可識別 Apache

有關在 Microsoft Windows 平台上編譯和安裝 Apache 的說明，請參閱以下網站：

Apache 1.3：

http://httpd.apache.org/docs/win_compiling.html

Apache 2：

http://httpd.apache.org/docs-2.0/platform/win_compiling.html

請按照以下步驟在其他平台上編譯、配置和安裝 SSL 可識別 Apache Web Server。儘管範例顯示的是編譯和建置 Apache 1.3.29，但相同的程序也適用於 Apache 2。

備註 在相同的目錄層級下對 mod_ssl、OpenSSL 和 Apache 解除磁帶存檔。

- 編譯和建置 OpenSSL
- 使用 mod_ssl 配置 Apache
- 編譯和建置 Apache
- 啓動和停止 Apache

編譯和建置 OpenSSL

如果隨 Linux 安裝的 OpenSSL 的版本爲 0.9.7d，則在 Linux 上不需要執行此步驟。

如需有關 OpenSSL 的更多資訊，請參閱：

http://www.openssl.org/

解壓縮 openssl-0.9.7d 源代碼並按照以下步驟進行操作。

1. cd openssl-0.9.7d
2. ./config
3. make
4. make test
5. make install

有關從源代碼建置 OpenSSL 的更多資訊，請參閱 openssl 目錄中的 INSTALL 檔案。

使用 mod_ssl 配置 Apache

本小節僅適用於 Apache 1.3。有關 Apache 2.0 的安裝，請跳至第 365 頁的「編譯和建置 Apache」。

如需有關 mod_ssl 的更多資訊，請參閱：

http://www.modssl.org/

1. 下載 apache_1.3.29 原始碼分發檔。

   解壓縮原始碼分發檔。原始碼分發檔以壓縮的歸檔檔案形式提供。對於 apache_1.3.29，原始碼分發檔歸檔檔案將讀取 apache_1.3.29.tar.gz。

2. 使用以下指令解壓縮歸檔檔案：

   ```
   tar -zxvf apache_1.3.29.tar.gz
   ```

   該指令將在目前的工作目錄下建立名爲 apache_1.3.29 的目錄。

3. 解壓縮 mod_ssl-2.8.14-1.3.29 源代碼。
4. cd mod_ssl-2.8.14-1.3.29
5. 執行 ./configure --with-apache=../apache_1.3.29 --with-ssl=../openssl-0.9.7d --prefix=*install path* --enable-module=ssl --enable-shared=ssl --enable-rule=SHARED_CORE --enable-module=so

以上指令範例中指定的目錄爲一個變數。*prefix* 引數指示要安裝 Apache 的位置。此指令將在螢幕上輸出多行。

根據您的系統配置，此指令將爲建置建立 `make` 檔案。`configure` 中的錯誤會導致某些標頭檔或公用程式遺漏。請先安裝它們，然後再繼續進行操作。

編譯和建置 Apache

Apache 的版本不同，編譯和建置 Apache 的說明也將不同。

- 編譯和建置 Apache 1.3
- 編譯和建置 Apache 2

編譯和建置 Apache 1.3

此程序將在第 364 頁的「使用 mod_ssl 配置 Apache」中介紹的 `--prefix` 屬性提供的位置安裝 Apache。

1. 在 Linux 上，在 `src/MakeFile` 的 `End of automatically generated section` 之後包含以下行：

   ```
   LIBS+= -licuuc -licui18n -lnspr4 -lpthread -lxerces-c -lsupport
   -lnsprwrap -lns-httpd40
   ```

   ```
   LDFLAGS+= -L/appserver_installdir/lib
   ```

2. 在 Linux 上，將 Application Server 的安裝目錄放到 LD_LIBRARY_PATH 中：

   ```
   export LD_LIBRARY_PATH=/app_server_install_dir/lib:$LD_LIBRARY_PATH
   ```

3. 使用 `make` 指令編譯 Apache，如下所示：

   a. 使用 cd 指令進入 `mod_ssl` 目錄。

   b. `make`

   c. `make certificate`

   d. `make install`

備註 指令 `make certificate` 需要安全密碼。請記住此密碼，因爲啓動安全 Apache 時需要它。

指令 `make install` 將在螢幕上輸出多行，指示程序正在編譯 Apache 源代碼並正在連結 Apache。此程序通常不會出現錯誤。但如果發生錯誤，請檢查是否已正確下載 Apache 的所有程式庫檔案和公用程式。

透過在 *apache_install_path*`/conf/httpd.conf` 檔案中爲環境輸入適當的值來配置 Apache 安裝。

編譯和建置 Apache 2

1. 下載 Apache 2_0_*NN* 原始碼分發檔。

   NN 表示次要版本號碼，例如 52。

2. 解壓縮原始碼分發檔。

   原始碼分發檔以壓縮的歸檔檔案形式提供。對於 Apache 2_0_*NN*，原始碼分發檔歸檔檔案爲 `httpd-2_0_`*NN*`.tar.gz`。

3. 使用以下指令解壓縮歸檔檔案：

   ```
   tar -zxvf httpd-2_0_NN.tar.gz
   ```

   該指令將在目前的工作目錄中建立名爲 `httpd-2_0_`*NN* 的目錄。

4. `cd httpd-2_0_`*NN*。

5. 執行 `./configure --with-ssl=`*open_ssl_install_path* `--prefix=`*install_path* `--enable-ssl --enable-so`

6. 在 Linux 上，修改 *apache_src*`/build/config_vars.mk` 並新增以下行：

   ```
   EXTRA_LIBS += -licuuc -licui18n -lnspr4 -lpthread -lxerces-c
   -lsupport -lnsprwrap -lns-httpd40

   LDFLAGS+=-L<appserver install dir>/lib
   ```

7. 在 Linux 上，將 Application Server 的安裝目錄放到 LD_LIBRARY_PATH 中：

   ```
   export LD_LIBRARY_PATH=/app_server_install_dir/lib:$LD_LIBRARY_PATH
   ```

8. 使用 make 指令編譯 Apache，如下所示：

   在 `httpd-2_0_`*NN* 目錄中執行：

   a. `make`

   b. `make install`

   指令 `make install` 將在螢幕上輸出多行，指示程序正在編譯 Apache 源代碼並正在連結 Apache。此程序通常不會出現錯誤。但如果發生錯誤，請檢查是否已正確下載 Apache 的所有程式庫檔案和公用程式。

透過在 *apache_install_path*/conf/httpd.conf 檔案中爲環境輸入適當的值來配置 Apache 安裝。

備註 如果遇到錯誤，請嘗試將 Application Server 安裝目錄放入 PATH 中：

```
export LD_LIBRARY_PATH=/app_server_install_dir/lib
```

或者新增 OpenSSL 程式庫，例如：

```
export
LD_LIBRARY_PATH=/openssl_install_dir/lib:/app_server_install_dir/lib
```

備註 在 Apache 2 上，您必須手動建立並安裝證書。如需更多資訊，請參閱 Apache 文件。

啟動和停止 Apache

Apache 隨附一個標題爲 apachectl 的程序檔，該程序檔可讓啓動、停止和重新啓動 Apache 更容易。

- 執行以下指令以啓動 Apache：

  apache_install_dir/bin/apachectl start

- 執行以下指令以在 SSL 模式下啓動 Apache：

  apache_install_dir/bin/apachectl startssl

- 若要停止 apache，請執行以下指令：

  apache_install_dir/bin/apachectl stop

啓動 Apache 之後，請測試安裝。Apache 執行後，請在 Web 瀏覽器中鍵入以下位址：http://*server_name*:*port_number*/。如果安裝成功且 Apache 正在執行，將顯示測試頁面。

完成 Apache 安裝後，請參閱第 62 頁的「對 Apache Web Server 的修改」以獲得在安裝外掛程式中和安裝外掛程式後配置 Apache 的資訊。

自動重新啟動網域或節點代理程式

如果網域或節點代理程式意外停止 (例如，您需要重新啓動機器)，您可以將系統配置爲自動重新啓動網域或節點代理程式。

本附錄包含下列主題：

- 在 UNIX 平台上自動重新啓動
- 在 Microsoft Windows 平台上自動重新啓動
- 自動重新啓動的安全性

在 UNIX 平台上自動重新啟動

若要在 UNIX 平台上重新啓動網域，請在 /etc/inittab 檔案中新增一行文字。

例如，要重新啓動安裝在 opt/SUNWappserver 目錄中的 Application Server 的 domain1，請使用名爲 password.txt 的密碼檔案。

```
das:3:respawn:/opt/SUNWappserver/bin/asadmin start-domain --user admin
--passwordfile /opt/SUNWappserver/password.txt domain1
```

將這些文字放在一行上。前三個字母是程序的唯一指示符，可以進行更改。

重新啓動節點代理程式的語法與此相似。例如，要重新啓動安裝目錄爲 opt/SUNWappserverApplication Server 的 agent1，並使用名爲 password.txt 的密碼檔案。

```
das:3:respawn:/opt/SUNWappserver/bin/asadmin start-node-agent --user admin
--passwordfile /opt/SUNWappserver/password.txt agent1
```

在 Microsoft Windows 平台上自動重新啟動

若要在 Microsoft Windows 上自動重新啓動，請建立一個 Windows 服務。結合使用 Sun Java System Application Server 隨附的 appservService.exe 和 appserverAgentService.exe 可執行檔以及 Microsoft 提供的服務控制指令 (sc.exe)。

Windows XP 隨附了 sc.exe 指令，該指令位於 C:\windows\system32 目錄或 C:\winnt\system32 目錄。編寫本文檔時，已經可以從以下位址下載 Windows 2000 sc.exe：ftp://ftp.microsoft.com/reskit/win2000/sc.zip°C 如需有關使用 sc.exe 的更多資訊，請參閱 http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dndllpro/html/msdn_scmslite.asp。

使用 appservService.exe 和 appservAgentService.exe，如下所示：

```
C:\winnt\system32\sc.exe create service_name binPath=
\"fully_qualified_path_to_appservService.exe  \"fully_qualified_path_to_asadmin.bat start_command\"
\"fully_qualified_path_to_asadmin.bat stop_command\"" start= auto DisplayName=
"display_name"
```

例如，若要建立用於啓動和停止網域 domain1 的名爲 SunJavaSystemAppServer DOMAIN1 的服務，請使用密碼檔案 C:\Sun\AppServer\password.txt：

```
C:\windows\system32\sc.exe create domain1 binPath=
"C:\Sun\AppServer\lib\appservService.exe \"C:\Sun\AppServer\bin\asadmin.bat
start-domain --user admin --passwordfile C:\Sun\AppServer\password.txt
domain1\" \"C:\Sun\AppServer\bin\asadmin.bat stop-domain domain1\"" start=
auto DisplayName= "SunJavaSystemAppServer DOMAIN1"
```

若要建立用於啓動和停止節點代理程式 agent1 的服務，請使用：

```
C:\windows\system32\sc.exe create agent1 binPath=
"C:\Sun\AppServer\lib\appservAgentService.exe
\"C:\Sun\AppServer\bin\asadmin.bat start-node-agent --user admin
--passwordfile C:\Sun\AppServer\password.txt agent1\"
\"C:\Sun\AppServer\bin\asadmin.bat stop-node-agent agent1\"" start= auto
DisplayName= "SJESAS_SE8.1 AGENT1"
```

備註 作爲 binPath= 參數的一部分而輸入的啓動和停止指令必須具有正確的語法。請在指令提示符號下執行這些指令以進行測試。如果這些指令不能正確地啓動或停止網域或節點代理程式，則說明該服務不能正常工作。

備註 請勿將服務與 asadmin 啓動和停止指令混合使用來進行啓動和停止。否則，將導致伺服器狀態不同步。例如，即使元件沒有執行，該服務可能也會顯示已啓動元件。爲避免發生這種情況，請在使用服務時始終使用 sc.exe 指令來啓動和停止元件。

自動重新啟動的安全性

當按以下方式之一啓動時，需要輸入所需的密碼和主密碼：

- 在 Microsoft Windows 上，將服務配置爲要求使用者輸入密碼。
  - **a.** 在 [服務] 控制台中，按兩下您所建立的服務。
  - **b.** 在 [特性] 視窗中，按一下 [登入] 標籤。
  - **c.** 核取 [允許服務與桌面互動]，以在啓動元件時提示輸入要求的密碼。

    必須登入才能看到提示，鍵入的項目不會在螢幕上顯示出來。如果選擇使用服務，這種方法最爲安全，但需要進行使用者互動才能啓動服務。

    如果未設定 [與桌面互動] 選項，服務將保持「啓動暫掛」狀態，並且將顯示爲停止。中止該服務程序即可從此狀態中恢復。
- 在 Windows 或 UNIX 上，使用 --savemasterpassword=true 選項建立一個網域，並建立一個儲存管理員密碼的密碼檔案。啓動元件時，使用 --passwordfile 選項來指向包含密碼的檔案。

  例如：
  - **a.** 建立具有已儲存主密碼的網域。在下面的語法中，系統將提示您輸入管理員密碼和主密碼：

    ```
    asadmin create-domain --adminport 4848 --adminuser admin
    --savemasterpassword=true --instanceport 8080 domain1
    ```
  - **b.** 在 Windows 上，建立使用密碼檔案的服務，以填入管理員密碼：

    ```
    C:\windows\system32\sc.exe create domain1 binPath=
    "C:\Sun\AppServer\lib\appservService.exe
    \"C:\Sun\AppServer\bin\asadmin.bat start-domain --user admin
    --passwordfile C:\Sun\AppServer\password.txt domain1\"
    \"C:\Sun\AppServer\bin\asadmin.bat stop-domain domain1\"" start=
    auto DisplayName= "SJESAS_PE8.1 DOMAIN1"
    ```

密碼檔案 password.txt 的路徑爲 C:\Sun\AppServer\password.txt。該檔案包含以下格式的密碼

AS_ADMIN_password=*password*

例如，密碼 adminadmin 在該檔案中的格式爲：

AS_ADMIN_password=adminadmin

c. 在 UNIX 上，請使用新增到 inittab 檔案的行中的 --passwordfile 選項：

```
das:3:respawn:/opt/SUNWappserver/bin/asadmin start-domain --user
admin --passwordfile /opt/SUNWappserver/password.txt domain1
```

密碼檔案 password.txt 的路徑爲 /opt/SUNWappserver/password.txt。該檔案包含以下格式的密碼

AS_ADMIN_password=*password*

例如，密碼 adminadmin 在該檔案中的格式爲：

AS_ADMIN_password=adminadmin

domain.xml 的含點名稱屬性

本附錄介紹可用於描述 Mbean 及其屬性的含點名稱屬性。domain.xml 檔案中的每個元素都有對應的 MBean。由於使用這些名稱的語法是利用句點來分隔名稱，所以這些名稱稱爲「含點名稱」。

本附錄包含下列主題：

- 頂層元素
- 不能別名化的元素

頂層元素

domain.xml 檔案中的所有頂層元素都必須遵循以下條件：

1. 每個伺服器、配置、叢集或節點代理程式的名稱都必須是唯一的。
2. 伺服器、配置、叢集或節點代理程式無法命名爲「網域」。
3. 伺服器實例可命名爲「代理程式」。

下表列出了頂層元素及其對應的含點名稱字首。

表 C-1 頂層元素

元素名稱	含點名稱字首
應用程式	domain.applications
資源	domain.resources
配置	domain.configs
伺服器	domain.servers 該元素包含的所有伺服器都可以作為 *server-name* 進行存取。其中，*server-name* 是伺服器子元素的名稱屬性值。

表 C-1 頂層元素

元素名稱	含點名稱字首
叢集	domain.clusters 該元素包含的所有叢集都可以作為 *cluster-name* 進行存取。其中，*cluster-name* 是叢集子元素的名稱屬性值。
node-agents	domain.note-agents
lb-configs	domain.lb-configs
system-property	domain.system-property

別名包括兩個層級：

1. 利用第一層級的別名可以存取伺服器實例或叢集的屬性而不必通過 domain.servers 或 domain.clusters 字首。因此，舉例來說，形式爲「server1」的含點名稱將與含點名稱 domain.servers.server1 (其中的 server1 爲伺服器實例) 相對映。
2. 第二層級的別名用於表示叢集或獨立伺服器實例 (目標) 的配置、應用程式和資源。

下表列出了以伺服器名稱或叢集名稱開頭的含點名稱，這些含點名稱被別名化爲網域下的頂層名稱：

表 C-2 網域下的含點名稱伺服器名稱

含點名稱	別名化為	注釋
target.applications.*	domain.applications.*	該別名解析為僅由目標參考的應用程式。
target.resources.*	domain.resources.*	該別名解析為由目標參考的所有 `jdbc-connection-pool`、`connector-connection-pool`、`resource-adapter-config` 和所有其他資源。

下表列出了以伺服器名稱或叢集名稱開頭的含點名稱，這些含點名稱在伺服器或叢集所參考的配置中被別名化爲頂層名稱。

表 C-3 伺服器或叢集所參考配置的含點名稱

含點名稱	別名化為
target.http-service	*config-name*.http-service
target.iiop-service	*config-name*.iiop-service

表 C-3 伺服器或叢集所參考配置的含點名稱

含點名稱	別名化為
target.admin-service	*config-name*.admin-service
target.web-container	*config-name*.web-container
target.ejb-container	*config-name*.ejb-container
target.mdb-container	*config-name*.mdb-container
target.jms-service	*config-name*.jms-service
target.log-service	*config-name*.log-service
target.security-service	*config-name*.security-service
target.transaction-service	*config-name*.transaction-service
target.monitoring-service	*config-name*.monitoring-service
target.java-config	*config-name*.java-config
target.availability-service	*config-name*.availability-service
target.thread-pools	*config-name*.thread-pools

不能別名化的元素

不應對叢集實例進行別名化。若要獲得叢集實例的系統特性，應按以下方法使用含點名稱屬性：domain.servers.clustered-instance-name.system-property，而非 clustered-instance-name.system-property。

索引

英文字母

ACC
    請參閱容器：應用程式用戶端
accesslog 特性
    虛擬伺服器 284
active-healthcheck-enabled 69
AddressList 特性 145
AddressListBehavior 特性 146
AddressListIterations 特性 146
allowLinking 特性
    虛擬伺服器 285
Apache
    SSL 可識別 , 安裝 363
    負載平衡器外掛程式所作的修改 62
    適用於 1.3 的最低需求 361
    適用於 2 的最低需求 362
append-version 特性 155
applets 187
Application Server
    重新啓動 194
    關閉 194
Application Server 網域 28
asadmin 公用程式 27
asadmin 指令 298, 300
    create-threadpool 298
    delete-threadpool 300
bean-cache
    監視屬性名稱 317
cache-hits 317
cache-misses 317
chunkedRequestBufferSize 特性 278
chunkedRequestTimeoutSeconds 特性 278
classpath
    在生命週期模組中 110
ClientID 特性 145
configure-ha-cluster 指令 134
configure-ha-persistence 指令 134
CORBA 291
    執行緒
create-domain 指令 29
create-http-lb-config 指令 66
create-http-lb-ref 指令 67
create-jndi-resource 指令 169
create-node-agent 指令 96
default-config 配置 182
delete-domain 指令 29
delete-http-lb-ref 指令 67
delete-node-agent 指令 98
Description 特性
    JMS 目標資源 148
disable-http-lb-application 指令 72

disable-http-lb-server 指令 71
dnsCacheEnabled 特性 278
docroot 特性
  虛擬伺服器 284
EAR 檔案 101
EJB JAR 檔案 101
EJB 計時器
  遷移 53
EJB 容器
  可用性 139
EJB 模組
  部署 107
enable-http-lb-application 指令 68
enable-http-lb-server 指令 68
execution-time-millis 315
export-http-lb-config 指令 70
get 指令
  監視資料 343
HADB
  和階段作業持續性 134
HTTP
  HTTPS 路由 73
  階段作業防故障備用 72
HTTP 服務
  chunkedRequestBufferSize 特性 278
  chunkedRequestTimeoutSeconds 特性 278
  dnsCacheEnabled 特性 278
  HTTP 偵聽程式 275
  HTTP 通訊協定 281
  HTTP 檔案快取 282
  keepAliveQueryMaxSleepTime 特性 278
  keepAliveQueryMeanTime 特性 278
  monitoringCacheEnabled 特性 278
  monitoringCacheRefreshInMillis 特性 278
  ssl3SessionTimeout 特性 278
  sslCacheEntries 特性 278
  sslClientAuthDataLimit 特性 278
  sslClientAuthTimeout 特性 278
  sslSessionTimeout 特性 278
  stackSize 特性 278
  statsProfilingEnabled 特性 278
  traceEnabled 特性 277
  存取記錄 279
  持續作用子系統 276, 280
  配置 277
  連線池 281
  虛擬伺服器 274
  請求處理執行緒 276, 280
  總覽 273
HTTP 偵聽程式
  刪除 289
  建立 287
  接收器執行緒 276
  預設虛擬伺服器 275
  編輯 289
  總覽 275
HTTP 通訊協定
  HTTP 服務 281
HTTP 連接埠 , 變更 45
HTTP 階段作業 188
  階段作業持續性 133
HTTP 檔案快取
  HTTP 服務 282
HTTP_LISTENER_PORT 特性 184
HTTP_SSL_LISTENER_PORT 特性 184
HTTPS
  階段作業防故障備用 72
  路由 66, 73
IIOP 偵聽程式 292
  刪除 295
  建立 293
  編輯 294
IIOP 連接埠 , 變更 45
IIOP_LISTENER_PORT 特性 184
IIOP_SSL_MUTUALAUTH_PORT 特性 184
instance-name 特性 155
instance-name-suffix 特性 155
IOP_SSL_LISTENER_PORT 特性 184
J2EE 群組 223
J2SE 軟體 47
Java 命名和目錄服務

請參閱 JNDI
Java 訊息服務 (JMS)
請參閱 JMS 資源 141
JavaMail 26
JavaMail API
總覽 159
JavaMail 階段作業
刪除 162
建立 160
編輯 161
JavaServer Page 187
JCE 提供者
配置 254
JDBC 25
resources 195
驅動程式 268
JMS 主機
刪除 157
建立 156
編輯 156
JMS 提供者 141
append-version 特性 155
instance-name 特性 155
instance-name-suffix 特性 155
JMS 主機 156, 157
配置 152
JMS 資源
主題 142
目標資源 142, 147, 148, 149
佇列 142
連線工廠資源 142, 143, 146, 147
實體目標 142, 150, 151
總覽 142
jms-max-messages-load 317
jmsra 系統資源介面 143
JMX 偵聽程式
節點代理程式 95
JMX_SYSTEM_CONNECTOR_PORT 特性 184
JNDI 188
EJB 的查詢名稱 102
外部資源 , 刪除 170
外部資源 , 建立 168
外部資源 , 編輯 169
外部儲存庫 168
名稱 164, 195
自訂資源 , 刪除 167
自訂資源 , 使用 165
自訂資源 , 建立 166
查找及關聯的參考 165
JSP
請參閱 JavaServer Page
keepAliveQueryMaxSleepTime 特性 278
keepAliveQueryMeanTime 特性 278
kestore.jks 檔案 242
list 指令
監視 342
list-custom-resources 指令 167
list-domains 指令 29
list-jndi-resource 指令 170
loadbalancer.xml 檔案 70
magnus.conf 檔案 , Web 伺服器 61
maxNumActiveConsumers 特性
JMS 實體目標 151
MessageServiceAddressList 特性 145
Microsoft 網際網路資訊服務 (IIS) , 用於負載平衡的修改 64
monitoringCacheEnabled 特性 278
monitoringCacheRefreshInMillis 特性 278
Name 特性
JMS 目標資源 148
num-beans-in-pool 317
number-healthcheck-retries 69
num-expired-sessions-removed 317
num-passivation-errors 317
num-passivations 317
num-passivation-success 317
num-threads-waiting 317
Oasis Web 服務安全性
請參閱 WSS
obj.conf 檔案 , Web 伺服器 61
Oracle 195

ORB
    請參閱物件請求代理程式
Password 特性 145
PointBase 195
RAR 檔案 101
ReconnectAttempts 特性 145
ReconnectEnabled 特性 145
ReconnectInterval 特性 145
resources
    爲叢集配置 55
RMI-IIOP 負載平衡和防故障備用 82
RSA 加密 254
Servlet 187
Solaris
    支援 21
    修補程式 21
ssl3SessionTimeout 特性 278
sslCacheEntries 特性 278
sslClientAuthDataLimit 特性 278
sslClientAuthTimeout 特性 278
sslSessionTimeout 特性 278
sso-enabled 特性
    虛擬伺服器 284
sso-max-inactive-seconds 特性
    虛擬伺服器 284
sso-reap-interval-seconds 特性
    虛擬伺服器 284
stackSize 特性 278
start-domain 指令 30, 167, 170
start-node-agent 指令 97
statsProfilingEnabled 特性 278
stop-domain 指令 30
stop-node-agent 指令 98
Sun Java System Message Queue 141
Sun Web 伺服器
    負載平衡器所作的修改 61
sun-passthrough.properties 檔案 , 和記錄層級 77
sun-web.xml 檔案 137
total-beans-created 317
total-beans-destroyed 317
total-num-errors 315
total-num-success 315
traceEnabled 特性 277
truststore.jks 檔案 242
UserName 特性 145
WAR 檔案 101
Web 伺服器
    用於負載平衡的修改 61
    多個實例和負載平衡 65
Web 服務 25
web 容器
    可用性 137
Web 階段作業
    請參閱 HTTP 階段作業
Web 應用程式 101
    可分散 134
    啓動 106
    部署 104

三劃

工作佇列
    請參閱執行緒池
已命名的配置
    default-config 182
    目標 185
    共用 182
    刪除 185
    建立 183
    特性 184
    連接埠號和 182
    預設名稱 182
    編輯 184
    獨立 182
    關於 181

四劃

中央儲存庫
  部署的應用程式 100
  節點代理程式同步 90
支援
  Solaris 21
記錄記錄 301
記錄層級
  配置 305
目標 , JMS
  Description 特性 148
  maxNumActiveConsumers 特性 151
  Name 特性 148
  刪除目標資源 149
  刪除實體目標 151
  建立目標資源 147
  建立實體目標 150
  編輯目標資源 148
  總覽 142
目錄部署 119

五劃

主終點 , RMI-IIOP 防故障備用 83
主題 , JMS 142
可分散的 Web 應用程式 134
可用性 133
  EJB 容器層級 139
  Web 容器層級 137
  伺服器實例層級 137
  啓用和停用 135
  資料庫 , 請參閱 HADB
  層級 135
外部資源
  刪除 170
  建立 168
  編輯 169
外部儲存庫 , 存取 168
未指定的請求 58
生命週期模組
  classpath 110
  建立 110
  載入次序 110
用戶端存取 25
用於應用程式的服務 25
目標
  已命名的配置 185
  已部署的應用程式 100
  負載平衡器配置 67
  管理應用程式 115

六劃

企業 Java Bean
  執行緒
企業 JavaBean
  有狀態階段作業 192, 194
  快取 188, 191, 192
  使用中 192
  建立 188
  持續性 188
  計時器服務 194
  訊息導引 188, 193
  從快取中移除 193
  授權 188
  啓動 188
  移除閒置 194
  無狀態階段作業 191
  鈍化 188, 191, 192
  閒置 191, 192
  階段作業 188
  匯集 191, 193
  實體 188, 191, 192
企業應用程式 101
  部署 103
存取記錄
  HTTP 服務 279
安全性 25
有狀態階段作業 Bean

階段作業持續性 133, 136
請參閱企業 JavaBean
有狀態階段作業 Bean 狀態的檢查點操作 134
自訂資源
列示 167
刪除 167
使用 165
建立 166
自動部署應用程式 117

七劃

佇列
工作
請參閱執行緒池
佇列 , JMS 142
伺服器記錄
檢視 306
伺服器實例
爲叢集建立 53
啓用以進行負載平衡 68
遷移 EJB 計時器 53
靜止 71
伺服器管理 26
作業事件 267
JMS 連線工廠 144
分散式 268
分隔 268
企業 JavaBean 191
完成 268
恢復 268, 269
記錄 270
逾時 270
管理員 268
確定 268
轉返 268
關聯 268
屬性 268
作業事件服務
監視 324
作業事件管理 25
作業事件管理員
請參閱作業事件：管理員
快取
企業 JavaBean 191
停用 191
清除 193
逾時 193
防故障備用
RMI-IIOP 需求 82
和階段作業持續性 133
關於 HTTP 58

八劃

取消部署應用程式 114
命名
JNDI 與資源參考 164
命名和目錄服務 25
命名服務 25
命名慣例 , 針對應用程式 102
定義叢集 34
服務
計時器
物件請求代理程式
執行緒
物件請求代理程式 (ORB) 291
IIOP 偵聽程式 292
配置 292
總覽 292
物件請求仲裁介面 (ORB)
服務 , 監視 323

九劃

持續作用子系統
HTTP 服務 276, 280
指定的請求 58

是一個指令行 27
計時器
  請參閱企業 JavaBean：計時器服務
計時器 , EJB
  遷移 53
計時器服務
  請參閱企業 JavaBean：計時器服務
負載平衡
  HTTP 算法 59
  HTTP 需求 58
  HTTP, 關於 58
  RMI-IIOP 需求 82
  記錄訊息 75
  多個 Web 伺服器實例 65
  建立負載平衡器配置 66
  建立參考 67
  指定的請求 58
  動態重新配置 71
  捲動升級 79
  啓用伺服器實例 68
  啓用應用程式 68
  設定 60
  等冪 URL 74
  階段作業防故障備用 72
  匯出配置檔案 70
  運作狀態檢查程式 68
  靜止伺服器實例或叢集 71
  靜止應用程式 72
  變更配置 70
  粘性循環 59
重新啓動伺服器 30
重新部署應用程式 100, 116

十劃

容器 25
  applet 187
  J2EE 187
  servlet
    請參閱容器：web
  web 187
  企業 JavaBean 187, 188, 191
    配置 191
  應用程式用戶端 187
效能
  問題 191
  執行緒池
  提高 191
特性
  已命名的配置 184
記錄
  作業事件 270
  負載平衡器 75
  記錄程式名稱空間 302
  配置一般設定 304
  配置層級 305
  概況 301
  檢視伺服器記錄 306
  檢視節點代理程式記錄 91
訊息傳送 25
配置。請參閱已命名的配置
高可用性 24
  請參閱可用性
高可用性資料庫
  請參閱 HADB

十一劃

停用應用程式 114
動態重新配置 , 負載平衡器 71
基於 Cookie 的階段作業粘性 59
執行緒
  移除 298, 299
  請參閱執行緒池
執行緒池
  工作佇列 299
  刪除 299
  命名 298
  建立 298
  效能
  執行緒不足 297

閒置 298, 299
逾時 298, 299
編輯 299
捲動升級 79
接收器執行緒 , 在 HTTP 偵聽程式中 276
啓用應用程式 114
清除間隔 189, 190
異常的伺服器實例 68
終點 , RMI-IIOP 防故障備用 83
連接埠偵聽程式 44
連接埠號
和配置 182
連接埠號 , 檢視 44
連接埠號 , 變更 44
連接器 26
模組
連接器連線池
JMS 資源和 143
連接器資源
JMS 資源和 143
連接器模組
部署 108
連線工廠 , JMS
AddressList 特性 145
AddressListBehavior 特性 146
AddressListIterations 特性 146
ClientID 特性 145
MessageServiceAddressList 特性 145
Password 特性 145
ReconnectAttempts 特性 145
ReconnectEnabled 特性 145
ReconnectInterval 特性 145
UserName 特性 145
作業事件支援 144
刪除 147
建立 143
編輯 146
總覽 142
連線池
HTTP 服務 281
部署
設定可用性 134
部署規劃 119
粘性循環負載平衡 59

十二劃

單次登入
和階段作業持續性 135
虛擬伺服器特性 284
循環負載平衡 , 粘性 59
替代終點 , RMI-IIOP 防故障備用 83
無狀態階段作業 Bean
請參閱企業 JavaBean
等冪 URL 74
虛擬伺服器
accesslog 特性 284
allowLinking 特性 285
docroot 特性 284
sso-enabled 特性 284
sso-max-inactive-seconds 特性 284
sso-reap-interval-seconds 特性 284
刪除 286
建立 283
將應用程式部署到其他 116
編輯 285
總覽 274
階段作業
HTTP 188, 191
ID 190
自訂 ID 190
刪除 190
刪除資料 189
非作用中的 189, 190
配置 188
逾時 189
管理 189
儲存 191
儲存資料 189
檔案名稱 189
階段作業防故障備用

HTTP 和 HTTPS 72
階段作業持續性 133
HTTP 階段作業 133
有狀態階段作業 Bean 133, 136
和單次登入 135
配置步驟 134
階段作業管理員 189
階段作業儲存
HTTP 階段作業 133, 137
有狀態階段作業 Bean 136, 139

十三劃

匯集
企業 JavaBean 191, 193
節點代理程式
JMX 偵聽程式 95
記錄 91
安裝 86, 89
自動建立 86
刪除 94, 98
附加 87
建立 96
停止 98
啓動 97
部署 87
萬用字元 87, 93
與網域管理伺服器同步 90
認證範圍 95
編輯 94
離線部署 89
關於 85
資料庫
JNDI 名稱 164
Oracle 195
PointBase 195
另請參閱 HADB
資源參考 164
資源 RAR 檔案 101
資源介面 268
jmsra 143
部署 108
資源參考 164
資源管理員 268
路由 cookie 67
載入次序 , 在生命週期模組中。110
運作狀態檢查程式 68
逾時 192, 193, 194
執行緒池 298, 299

十四劃

實例 34
實體 Bean
請參閱企業 JavaBean：實體
監視
bean-cache 屬性 317
ORB 服務 323
作業事件服務 324
使用 get 指令 343
使用 list 指令 342
容器子系統 311
管理主控台 26
算法
HTTP 負載平衡 59
RMI-IIOP 防故障備用 83
網域 28
建立 29
將應用程式部署到 100
網域管理伺服器
伺服器實例同步 90
節點代理程式同步 90
認證範圍
節點代理程式 95
說明文件
總覽 19

十五劃

模組描述元
  檢視 114
範圍
  節點代理程式認證 95
  證書 208
線上說明手冊 27
線上輔助說明 47
請求處理執行緒
  HTTP 服務 276, 280

十六劃

獨立 , 伺服器實例或叢集 182
錯誤頁面 , HTML 74
靜止
  伺服器實例或叢集 71
  應用程式 72

十七劃

應用程式
  目錄部署 119
  列示子元件 113
  列示已部署的 113
  在虛擬伺服器上進行部署 116
  自動部署 117
  取消部署 114
  命名慣例 102
  爲叢集配置 54
  重新部署 100, 116
  效能 191
  停用 114
  捲動升級 79
  啓用 114
  啓用以進行負載平衡 68
  部署規劃 119
  模組描述元 114
  靜止 72
應用程式用戶端 JAR 檔案 101
應用程式用戶端模組
  部署 111
應用程式的子元件 , 列示 113

十八劃

叢集 24
  用於線上升級 56
  共用 50
  伺服器實例和 50
  刪除 55
  建立 51
  建立伺服器實例 53
  負載平衡器和 50
  配置 52
  配置資源 55
  配置應用程式 54
  配置叢集伺服器實例 54
  階段作業和 50
  獨立 50
  靜止 71
  應用程式捲動升級 79
  總覽 49
叢集 , 定義 34
叢集伺服器實例
  配置 54, 182
轉返
  請參閱作業事件：轉返
離線部署節點代理程式 89

十九劃

關鍵點作業 271
關鍵點間隔 271