
前言

什麼是 Enterprise JavaBeans？

早期，當 Java™ 剛誕生的時候，整個資訊工業界都把注意力放在 Java 在 GUI 上的優異表現，以及其分散式和跨平台的優點上。時至今日，Java 更是被大家當作發展企業應用解決方案最佳的分散式平台，特別是用來開發分散式伺服端的應用系統。這樣的成長，正代表著 Java 逐漸成為業界開發的抽象介面 (abstract interface)，是和實作無關 (implementation independence) 的企業應用共通語言。JDBC™ API 是最為大家熟知的例子：JDBC 提供了與資料庫業者無關的 Java 抽象介面，來存取各種 SQL 關連性資料庫；由於這個抽象介面設計得相當成功，幾乎所有的資料庫廠商都支援 JDBC。近年來，使用 Java 開發的抽象介面領域顯著地增加，其中包括提供抽象化目錄服務的 JNDI™ (Java Naming and Directory Interface)，提供抽象化遠端週邊服務的 JMX (Java Management Extensions)，以及提供抽象化訊息導向中間層服務的 JMS™ (Java Messaging Service) 等等。

Enterprise JavaBeans™ (EJB) 是 Java 家族中最新的抽象介面技術，同時也是最具企圖心的一個。值得注意的是，EJB 提供一套將元件交易監控伺服器 (component transaction monitor, CTM) 抽象化的機制。CTM 可視為兩種技術的整合：其一是傳統的交易處理監控伺服器 (transaction processing monitor, TP Monitor)，如 CICS、TUXEDO 與 Encina；另一則是分散式物件服務，如 CORBA (Common Object Request Broker Architecture)、DCOM 以及 Java RMI。將這兩種技術做最佳整合後所誕生的 CTM 產品，提供了一個以元件為基礎的健全開發環境；同時由於它會自動管理許多和企業應用相關的複雜工作，如物件仲介 (object broker)、安全性 (security)、同時共用 (concurrency)、永續保存 (persistence) 以及交易管理 (transaction management) 等等，因此可以簡化開發分散式系統的工作。

透過 Enterprise JavaBeans 的伺服器端元件模型，企業物件 (business object) 可以在不同的 CTM 上開發、使用。由於元件 (bean) 是簡單的程式設計模型，因此程式員可以專注在它的企業應用目的 (business purpose) 上。另一方面，EJB 伺服器 (遵循 Enterprise JavaBeans 規格的 CTM) 負責將元件以分散式物件的形式來呈現，並且負責管理安全性、同時共用、永續保存以及交易行為。除了定義 bean 的企業邏輯 (business logic) 之外，程式員還必須指定 bean 在執行期間的屬性，這個過程類似於設定視覺化元件 (如按鈕) 外觀屬性的過程。藉由屬性表單，程式員可以為元件指定安全管理、永續保存以及交易的相關屬性。結論是：透過 Enterprise JavaBeans，分散式元件系統的交易行為可以獲得妥善的管理。程式員固然可以使用複雜的 CORBA、DCOM 或是 Java RMI 來開發關鍵任務導向 (mission-critical) 的分散式系統，同時可以擁有不錯的整體效能；不過相較之下，使用 Enterprise JavaBeans 的平台不但可以簡化開發系統的工作，同時可以提升設計程式的效率。

在很短的時間內，Enterprise JavaBeans 就已經成為業界的標準。許多業者都已發表相容於 EJB 的產品 (部分產品甚至在 EJB 規格尚未定案前便已發表)。另外，在資訊工業界處處可見遵循 EJB 標準的 CTM，包括 TP Monitor、CORBA、ORB、應用伺服器 (Application Server)、關連式資料庫、物件導向式資料庫以及 Web 伺服器等領域。部分的產品有專屬的元件模型，但仍相容於 EJB 規格；甚至有些產品完全遵循 EJB 元件模型。

簡言之，透過 Enterprise JavaBeans 的分散式伺服器端元件模型，不但可以簡化開發企業應用系統的工作，同時 bean 可以在不同的 EJB 伺服器上開發、配置。另一方面，本書將提供足夠的基本知識，讓你能開發出與 EJB 實作平台無關之 EJB 應用系統。

誰應該看這本書？

本書的內容是用來說明 Enterprise JavaBeans 的基本架構。雖然 EJB 可以簡化分散式計算的工作，但是它仍然是一項複雜的技術；也就是說，你需要花費相當多的功夫來學習如何駕馭它。本書採用直接而有條理的方式，來說明 EJB 背後的技術如元件模型以及相關的 Java 類別和 Java 介面；同時，本書也會對 Enterprise JavaBeans 在執行期間的行為進行說明。

雖然本書著重在基本的概念，但是它絕不是一本初淺的書籍。Enterprise JavaBeans 結合許多企業應用系統的相關技術，複雜的程度自然不在話下。在閱讀本書之前，你必須能夠使用 Java 來設計程式，並擁有開發企業應用系統的經驗。另一方面，你不需要具備開發分散式系統的經驗；不過如果想要使用本書的範例，你必須要會使用 JDBC 的基本功能。對 Java 不熟悉的讀者，可以參考由 Patrick Niemeyer 以及 Jonathan Knudsen 所著的《Learning Java™》，本書的前身就是著名的《Exploring Java™》一書；對 JDBC 不熟悉的讀者，可以參考由 George Reese 所著的《Database Programming with JDBC™ and Java™, 2/e》；最後，對分散式計算不熟悉的讀者，可以參考由 Jim Farley 所著的《Java™ Distributed Computing》（以上三本皆由 O'Reilly 出版）。

本書的架構

在這裡，我們對本書的架構做個介紹。本書前三章的題材主要是基本的知識，我們透過相關的技術以及抽象的方式來解釋 EJB 的運作方式，同時指出組成 enterprise bean 的基本元素。在四到七章中，我們詳盡而深入地探討如何開發各種不同類型的 enterprise bean。第八及第九章則是進階的課題：第八章介紹的交易是許多企業應用系統不可或缺的部分，第九章介紹一些設計的策略，可以幫助你在面對現實考量時，如何設計實用而優良的 bean。第十章則介紹了 EJB 1.1 所使用的 XML 配置描述子的細節。最後，第十一章簡介了 Java™ 2 企業版（Java 2, Enterprise Edition, J2EE）與 EJB 1.1 之間的關係。

第一章 簡介

我們在这一章定義 CTM，同時說明組成 Enterprise JavaBeans 元件模型的基本元素。

第二章 架構概要

我們在这一章中定義 Enterprise JavaBeans 元件模型的基本架構，同時介紹兩種不同類型的 enterprise bean，即 entity bean 與 session bean。

第三章 系統資源管理以及主要服務

我們在这一章中說明 EJB 伺服器如何在執行期間管理 enterprise bean。

第四章 開發你的 Enterprise Bean

在这一章中，我們將帶領讀者開發兩個簡單的 enterprise bean。

第五章 用戶端的觀點

我們在这一章中講解，遠端的用戶端程式如何使用到 enterprise bean。

第六章 Entity Bean

我們在这一章中深入介紹如何開發由 container 代理永續保存的 entity bean，以及如何開發由 bean 自行永續保存的 entity bean，並說明它們在執行期間的行為。

第七章 Session Bean

我們在这一章中深入介紹如何開發 stateless session bean 以及 stateful session bean，並說明它們在執行期間的行為。

第八章 交易

这一章對交易行為進行深入的介紹，並說明 Enterprise JavaBeans 如何支援交易行為的相關服務。

第九章 設計上的策略

这一章提供一些基本的設計策略，用來簡化開發 EJB 應用的工作，並使得 EJB 系統具備更高的效能。

第十章 XML 配置描述子

這一章將對 EJB 1.1 所使用的 XML 配置描述子進行深入的介紹。

第十一章 Java 2 企業版

這一章將對 Java 2 企業版進行一個概略性的介紹，並說明 EJB 1.1 如何與這個新平台整合在一起。

附錄 A Enterprise JavaBeans API

這個附錄提供了 Enterprise JavaBeans API 的快速參考指南，範圍涵蓋 `javax.ejb` package 及 `javax.ejb.deployment` package 的類別和介面。

附錄 B 狀態與流程示意圖

這個附錄提供 enterprise bean 生命週期的相關圖表。

附錄 C EJB 業者

這個附錄列出了一份清單，包含了目前所有發表 EJB 伺服器的業者資訊。

附錄 D EJB 1.1 新增的特色

這個附錄整理了從 EJB 1.0 改版至 EJB 1.1 過程中所作的改變。

軟體來源與版本說明

本書的內容涵蓋版本 1.1 以及 1.0 的 Enterprise JavaBeans 以及其選擇性的支援功能；另一方面，我們使用版本 1.1 的 Java 平台以及 JDBC。此外，由於本書著重在開發與業者無關的 Enterprise JavaBeans 元件，所以我們會盡量避免使用業者特定的擴充功能，以及業者自訂的規格。本書的介紹適用於所有和 EJB 相容的伺服器：你必須熟悉如何在伺服器上安裝、配置以及管理執行中的 bean，才可以使用本書的範例。至於在特定伺服器上配置、執行以及使用 bean 的相關細節，你可以參閱 EJB 業者提供的說明文件，這些細節並不會包含在 EJB 規格中。

本書同時涵蓋了 EJB 1.1 以及 EJB 1.0，這兩個版本大部分是相同的，但是我們會在遇到兩者處理方式相異的地方，分別加以詳述；因此，可以放心地跳過不屬於你所使用版本的小節。此外，除非有特別說明，本書所有程式碼都是以 EJB 1.1 的方式撰寫而成，若想要修改成 EJB 1.0 的版本，我們都已經將改寫時所需的資訊註解在程式碼中。

另一方面，你可以從 <ftp://ftp.oreilly.com/pub/examples/java/ejb> 或 <http://www.oreilly.com.tw/Chinese/Java/ejb2.html> 下載本書的範例，這些範例都依照本書的章節架構存放著。

印刷體裁

以下是本書所使用的印刷體裁。

Courier

用於程式碼範例、類別、變數、method 名稱、內文裡的 Java 關鍵字，以及 SQL 命令、資料表名稱、欄位名稱，此外還包括 XML 的元素 (element) 以及 tag 名稱。

Courier 粗體

用來強調一些程式碼範例。

粗明體 (Bold Ming-Font)

在內文中所提到的所有的專業術語都會使用這種字型。

Courier 斜體

用來表示該段文字的內容可以替換。例如，*BeanNamePK* 你可以用其它的 bean 名稱取代 *BeanName*。

Enterprise JavaBeans 是由許多部分所組成的；它並不是一個單獨的物件，而是一組物件與介面的組合。因此，在本書中我們以其遠端介面的名稱來表示該 bean，例如 BaggageHandler bean；如果所指的是該 bean 的遠端介面時，我們會以 Courier 字體來表示它，例如 BaggageHandler 是指 BaggageHandler bean 的遠端介面，其定義了該 bean 提供的所有 business method。

讀者迴響

我們雖然盡全力測試、核對書中的相關資訊。不過隨著時間的流逝，有些資料可能會有變化。若讀者真的發現類似情況，請通知我們。當然也歡迎各位來函提供意見，供作日後改版的參考。請透過電子郵件與我們聯絡：

mail@oreilly.com.tw

如果想詢問技術問題，請發電子郵件到：

bookquestions@oreilly.com (英文)

bookquestions@oreilly.com.tw (台灣地區，請用繁體中文)

最後，您可以在 WWW 上找到我們：

<http://www.oreilly.com/> (美國)

<http://www.oreilly.com.tw/> (台灣)

譯序

這是本談論 Enterprise JavaBeans 的技術書籍。

隨著電子商務的興起，以及企業界一片如火如荼「E」化的推波助瀾下，建構完善而功能強大的企業系統，並將企業交易模式擴展到 Internet 的舞台，一直是企業內部 MIS 人員的主要工作；然而，隨著日新月異的技術不斷地出現（尤其是 Microsoft 所提出的），常使得開發人員淹沒在龐大的資訊洪流之中，幾乎沒有喘息的餘地。在各種企業系統解決方案中，一個常見的組合，是以 UML 進行分析，採用 Microsoft 流派的 ASP + DCOM + MTS + 資料庫的方式建構整個企業系統；此外還有最近相當熱門的技術：適用於 Linux 平台下的 PHP + MySQL 之組合等等。而本書的主角 - Enterprise JavaBeans，簡稱 EJB，正是在 Java 平台上的解決方案。

也許有人看到本書名有「JavaBeans」一詞，心中可能會浮現「哈，又是一本教人寫 GUI 的書」的念頭。如果因此而決定將這本書留在書架上，那麼就辜負本書作者的苦心了。事實上，EJB 與 JavaBeans 的差別，除了兩者都是一種 Java 元件模型標準之外，幾乎沒有任何的關連；有關 EJB 的內涵以及定義，本書將花費許多篇幅加以闡述，相信一定能滿足你的需求。

本書並不是一本單純的 Java 技術書籍。在進行本書翻譯工作的同時，我一直想替本書寫一篇導讀；然而，在開始撰寫數頁之後，我決定放棄這個念頭，因為其中牽涉到的概念（包括了 OOAD、N 層式架構、交易管理的概念等等），非三言兩語可以說明清楚，我擔心這段導讀將膨脹成為一章的分量；此外，本書的目的也不在此。有興趣的讀者，國內 MISOO 所出版的一系列叢書以及物件導向雜誌，將是不錯的中文參考書籍。

本書所用到的專有名詞相當的多，並且橫跨許多不同的領域，因此，原文術語與中文譯名的取捨，是翻譯過程中最令我頭疼的一項工作。大致上，本書專有名詞的翻譯原則如下：

企業系統領域的專有名詞：基本上採用業界常用的中文譯名，包括了「企業物件」、「企業規則」等等，而無適當譯名的名詞則保持原文，例如 stateless session bean。其餘有些則視情況賦予譯名，例如 instance pooling 翻為「instance 輪調」等等。

Java 語言的專有名詞：雖然業界通常以原文術語來稱呼這些 Java 語言的術語，例如 class、object、interface 等等，然而由於本書並非著重在 Java 語言本身的技術，因此我採用了中文的譯名，像是「類別」、「物件」以及「介面」等等。其中，method 一字因為翻成「方法」容易與本文混淆，因此保留原文。

資料庫以及交易的相關名詞：幾乎都採用業界最常使用的原文術語，例如「primary key」、「commit」、「rollback」等等。

其餘無法分類的專有名詞：多採保留原文的方式處理，例如「loopback」、「narrow」等等。

最後，要感謝蔡寶進先生在技術以及潤稿上的大力幫助，以及林子揚先生對於本書許多關鍵技術的寶貴意見，使得本書翻譯工作能夠順利地完成。此外，原書第一版的譯者張晏誠先生也是功不可沒。當然更要感謝的是，發行人蘇秉豐先生對稿件品質的高度要求，以及對我的包容與支持。本書中當然還是有些錯誤不足之處，如果讀者發現任何錯誤，請來信指教，以作下一刷更正。

黃奕勤
simon@oreilly.com.tw
2000.10