

IBM DB2 Information Integrator
OmniFind Edition



企業搜尋程式設計指南與 API 參考手冊

8.2 版

IBM DB2 Information Integrator
OmniFind Edition



企業搜尋程式設計指南與 API 參考手冊

8.2 版

使用此資訊及其支援的產品之前，請先閱讀「注意事項」中的一般資訊。

本文包含 IBM 的所有權資訊，乃依據授權合約提供並受著作權法保護。本書中的資訊不包括任何產品保證，且其陳述也不得延伸解釋。

您可以線上訂購 IBM 出版品，或可以透過當地的 IBM 業務代表來訂購：

- 若要線上訂購出版品，請造訪「IBM 出版品中心 (IBM Publication Center)」：www.ibm.com/shop/publications/order
- 若要尋找當地的 IBM 業務代表，請造訪「IBM 全球聯絡站名錄 (IBM Directory of Worldwide Contacts)」：www.ibm.com/planetwide

當您傳送資訊給 IBM 時，即授權予 IBM，IBM 得以其認為適當的方式來使用或分送資訊，而無需對您負任何責任。

© Copyright International Business Machines Corporation 2004. All rights reserved.

目錄

第 1 章 關於本指南	1
本指南適用對象	1
哪裡可以找到更多資訊	1
第 2 章 企業搜尋 API	3
機密保護	3
第 3 章 搜尋及索引 API (SI-API)	5
SI-API 應用程式結構	5
控制查詢行為	7
查詢語法	8
建置 Java 原始碼	12
SearchExample 類別	12
AdvancedSearchExample 類別	14
BrowseExample 類別	17
SI-API 呼叫及方法	18
SiapiVersion 類別	18
SiapiSearchImpl 類別	19
SearchFactory 介面	19
CategoryInfo 介面	20
CollectionInfo 介面	20
FieldInfo 介面	21
Searchable 介面	22
SearchService 介面	27
ApplicationInfo 介面	27
Query 介面	29
NameValuePair 介面	40
Result 介面	41
ResultCategory 介面	43
ResultSet 介面	44
SpellCorrection 介面	46
BrowseFactory 介面	47
BrowseService 介面	48
Category 介面	48
TaxonomyBrowser 介面	49
SiapiException 類別	50
TaxonomyInfo 介面	56
範例 SI-API 應用程式	56
至少必要的應用程式	56
瀏覽及導覽應用程式	59
擷取所有搜尋結果	61
第 4 章 資料接聽器 API	63
資料接聽器 API 屬性	64
資料控制	64
使用資料接聽器 API 移除資料	65

使用資料接聽器 API 新增資料	66
將資料推播到索引器	67
指定中間資料	67
建立中間資料	67
資料接聽器 API 呼叫及方法	69
DLDataPusher 類別	69
範例資料接聽器 API 應用程式	71
基本資料接聽器 API 應用程式	72

DB2 Information Integrator 文件 75

DB2 Universal Database on z/OS 事件發佈功能的相關文件	75
IMS 及 VSAM on z/OS 事件發佈功能的相關文件	75
Linux、UNIX 及 Windows 事件發佈與抄寫功能的相關文件	76
Linux、UNIX 及 Windows 聯合功能的相關文件	77
z/OS 聯合功能相關文件	78
z/OS 抄寫功能相關文件	79
Linux、UNIX 及 Windows 企業搜尋功能的相關文件	80
版本注意事項及安裝需求	80
檢視版本注意事項及安裝需求	81
檢視及列印 PDF 文件	82
存取 DB2 Information Integrator 文件	82

協助工具 85

鍵盤輸入與導覽	85
鍵盤焦點	85
鍵盤輸入	85
鍵盤導覽	85
容易使用的顯示畫面	85
字型設定	86
不依靠顏色	86
與輔助技術的相容性	86
方便存取的文件	86

注意事項 87

商標	88
--------------	----

連絡 IBM 91

取得產品資訊 93

提供關於文件的意見 95

索引 97

第 1 章 關於本指南

本指南說明如何使用企業搜尋隨附的 Java™ 應用程式設計介面 (API)。API 即提供建立搜尋應用程式所需的工具。

本指南包括：

- 企業搜尋應用程式簡介
- API 元件文件
- 程式碼範例

DB2® Information Integrator OmniFind Edition 提供一種稱為企業搜尋的技術。當安裝 IBM® DB2 Information Integrator OmniFind Edition (DB2 II OmniFind) 時，就會同時安裝企業搜尋元件。除非光碟標籤或特定的產品元件另有指示，否則企業搜尋一詞僅用於 DB2 II OmniFind 文件。

本指南適用對象

本指南適用於想要建立自訂企業搜尋應用程式的應用程式設計人員。

您應具備下列技能：

- 熟悉應用程式設計
- 撰寫 Java 應用程式的經驗

哪裡可以找到更多資訊

請參閱下列資源，以取得更多資訊。

產品資訊

DB2 Information Integrator OmniFind Edition 的相關資訊可透過電話或從 Web 取得。

如果您住在美國，可以撥打下列號碼之一：

- 若要訂購產品或取得一般資訊，請撥 1-800-IBM-CALL (1-800-426-2255)
- 若要訂購出版品，請撥 1-800-879-2755

若要透過 **Web** 取得資訊，則請造訪 www.ibm.com/software/data/integration/db2ii/support.html。此網站提供下列最新資訊：

- 技術叢書
- 訂購書籍
- 用戶端下載
- 新聞群組
- 修正套件
- 新聞
- Web 資源鏈結

第 2 章 企業搜尋 API

企業搜尋 API 是一組 Java API，用來新增文件、移除文件，以及在企業搜尋集合中進行搜尋。

API 說明

IBM 搜尋及索引 API

使用 IBM 搜尋及索引 API (SIAPI) 可建置自訂搜尋應用程式。SIAPI 的企業搜尋實作容許遠端存取搜尋伺服器。這些 API 可讓您提交搜尋要求、處理搜尋結果，以及瀏覽分類架構樹狀結構。

資料接聽器 API

使用資料接聽器 API 可新增或移除企業搜尋集合中的文件。

資料接聽器是企業搜尋的元件，用來接收外部資料來源搜索器的資料。資料來源搜索器會連線到資料接聽器，然後將資料推播到目標集合。

在將資料推播到目標集合前，資料接聽器元件還可以接收新增到企業搜尋佇列並在內部進行搜索的文件。

機密保護

企業搜尋及 Browse API 是以遠端方式與 ESSearchServer Enterprise Application 進行通訊，因為 ESSearchServer Enterprise Application 安裝在每個搜尋節點的 WebSphere® Application Server 上。

在這個安全模式中，WebSphere Application Server 會盤查所有 HTTP 要求，來提供有效的使用者名稱及密碼。輸入的使用者名稱及密碼必須在作用中的使用者登錄是有效的，該登錄是透過 WebSphere 管理主控台配置。未包含有效使用者認證的要求都會被拒絕。

企業搜尋 SIAPI 實作會自動為您設定使用者名稱及密碼值。您可以在應用程式中建立 ApplicationInfo 類別的實例時，指定使用者名稱及密碼值。請參閱第 56 頁的『至少必要的應用程式』的範例。

搜尋應用程式名稱及密碼必須儲存在 WebSphere 鑑別所用的同一個儲存庫中。

限制:

搜尋 API 限制：

- 支援 HTTP BASIC 鑑別
- 不支援 HTTPS (SSL v2 或 v3)

第 3 章 搜尋及索引 API (SIAPI)

IBM 搜尋及索引 API (SIAPI) 是一種程式設計介面，可讓您搜尋並瀏覽集合及分類架構。

SIAPI 提供整合的程式設計介面，可讓您撰寫程式來搜尋各種 IBM 後端搜尋產品。

SIAPI 支援以下作業：

- 搜尋索引
- 自訂搜尋結果集中傳回的資訊
- 搜尋並瀏覽分類架構

SIAPI 應用程式結構

SIAPI 應用程式包括下列作業：

- 取得 SIAPI 實作 factory 物件
- 取得 SearchService 物件
- 取得 Searchable 物件
- 發出查詢
- 處理查詢結果

取得 SIAPI 實作 factory 物件

SIAPI 搜尋應用程式一開始會先取得實作 factory 物件。

```
SearchFactory factory =  
SiapiSearchImpl.createSearchFactory  
("com.ibm.es.api.search.RemoteSearchFactory");
```

SIAPI 是 factory 的 Java API。搜尋應用程式所使用的所有物件，都是透過呼叫 SIAPI 物件 factory 方法建立，或呼叫 factory 產生物件方法傳回。您可以載入不同的 factory，很容易切換不同 SIAPI 實作。

企業搜尋 SIAPI 實作由 com.ibm.es.api.search.RemoteSearchFactory 類別提供。

取得 SearchService 物件

使用 factory 物件來取得 SearchService 物件。SearchService 物件可讓您存取搜尋伺服器上可搜尋的集合。

您必須使用企業搜尋搜尋伺服器的主機名稱及埠，以及用來接收錯誤訊息所需的語言環境來配置 SearchService 物件。語言環境是 4 個字元或 5 個字元的 Java 字串。例如，英文為 "enUS" 或 "en_US"。

這些配置參數是在 java.util.Properties 中設定。然後，參數會傳送到產生 SearchService 物件的 getSearchService factory 方法。

```
Properties configuration = new Properties();
configuration.setProperty("hostname", "es.mycompany.com");
configuration.setProperty("port", "80");
configuration.setProperty("locale", "en_US");
SearchService searchService =
    factory.getSearchService(configuration);
```

取得 Searchable 物件

使用 SearchService 物件來取得 Searchable 物件。Searchable 物件在搜尋伺服器上有一個關聯的可搜尋集合。Searchable 物件可讓您發出查詢並取得關聯集合的相關資訊。每個企業搜尋集合都有一個 ID。

要求 Searchable 物件時，您必須使用應用程式 ID 來識別應用程式。請連絡企業搜尋管理員，以取得適當的應用程式 ID。

如果搜尋伺服器配置為開啓廣域機密保護，就需要提供密碼。此密碼是用來鑑別您的應用程式。如需詳細資訊，請參閱第 3 頁的『機密保護』。

```
ApplicationInfo appInfo = factory.createApplicationInfo("my_application_id");
appInfo.setPassword("my_password");
Searchable searchable =
    searchService.getSearchable(appInfo, "some_collection_id");
```

呼叫 getAvailableSearchables 方法來取得應用程式可使用的所有 Searchable 物件。

```
Searchable[] searchables =
    searchService.getAvailableSearchables(appInfo);
```

如需 Searchable 物件的相關資訊，請參閱第 22 頁的『Searchable 介面』。

發出查詢

取得 Searchable 物件後，您可以對這個 Searchable 物件發出查詢。對 Searchable 物件發出查詢：

- 建立 Query 物件。
- 自訂 Query 物件。
- 將查詢物件提給 Searchable 物件。
- 取得含括在 ResultSet 物件中的查詢結果。

```
String queryString = "big apple";
Query query = factory.createQuery(queryString);
query.setRequestedResultRange(0, 10);
ResultSet resultSet = searchable.search(query);
```

如需詳細資訊，請參閱第 7 頁的『控制查詢行爲』及第 8 頁的『查詢語法』。

處理查詢結果

ResultSet 及 Result 介面可讓您存取查詢結果。

```
Result[] results = resultSet.getResults();
for ( int i = 0 ; i < results.length ; i++ ) {
    System.out.println
    ( "Result " + i + ": " + results[i].getDocumentID()
      + " - " + results[i].getTitle() );
}
```

SI-API 與第 44 頁的『ResultSet 介面』及個別第 41 頁的『Result 介面』物件的互動方法有很多種。

控制查詢行為

Query 介面的下列方法可讓您控制各種查詢行為，包括如何處理查詢以及每個結果會傳回哪些中間資料。

表 1. 查詢行為方法

方法	說明
setQueryLanguage (String language)	指定使用搜尋伺服器所用集合預設語言以外的其他語言。例如，英文的查詢語言參數是 en-US。至於中文，簡體中文是 zh-CN，繁體中文則是 zh-TW。
setRequestedResultRange (int fromResult, int numberOfResults)	控制傳回的結果範圍。 fromResult 值控制結果集從哪一個排序位置的文件開始。例如，0 值表示您要求的是查詢結果中的第一個文件（ fromResult 必須是 0 或 numberOfResults 的整倍數）。 numberOfResults 值控制要傳回多少筆結果到目前的結果頁面。最大值是 100。
setReturnedAttribute (int attributeType, boolean isReturned)	啟用或停用與每個 Result 物件一起傳回的所有預先定義結果屬性值。 依預設，除了中間資料欄位屬性 (RETURN_RESULT_FIELDS) 外，企業搜尋會傳回所有的預先定義結果屬性值。
setReturnedFields (java.lang.String[] fieldNames)	控制 Result 物件中將傳回哪些中間資料欄位。 依預設，企業搜尋並不會傳回任何中間資料欄位。
setSiteCollapsingEnabled (boolean value)	如果最符合的結果有兩筆以上來自相同網站，則指定這個方法。 例如，如果特定的查詢從 www.ibm.com 上傳回 100 筆結果並啟用網站收合，則 ResultSet 只會在最符合的結果中顯示其中的 2 筆結果。在列出其他網站結果之後，才會出現其餘的結果。 若要從相同網站擷取更多結果，您可以將網站 www.ibm.com 加到查詢字串中，然後重新發出相同的查詢。
setSortKey (String sortKey)	指定排序鍵。com.ibm.siapi.search.BaseQuery 中定義了下列預先定義的排序鍵值： <ul style="list-style-type: none">• SORT_KEY_NONE• SORT_KEY_DATE• SORT_KEY_RELEVANCE 您可以指定排序鍵為所搜尋集合的任何其他有效數字欄位名稱。預設排序鍵為 SORT_KEY_RELEVANCE。

表 1. 查詢行為方法 (繼續)

方法	說明
setSortOrder (int sortOrder)	指定排序為 SORT_ORDER_ASCENDING 或 SORT_ORDER_DESCENDING。 如果排序鍵為 SORT_KEY_RELEVANCE 或 SORT_KEY_NONE，則忽略排序順序。
setSortPoolSize (int poolSize)	控制結果集中將排序並傳回的最符合結果筆數。值的範圍介於 1 到 500 (預設排序儲存區大小是 500 筆)。任何超出這個範圍的值都會導致搜尋伺服器擲出 SiapiException。 如果 sortKey 是 SORT_KEY_RELEVANCE 或 SORT_KEY_NONE，則忽略排序儲存區大小。
setPredefinedResultsEnabled (Boolean value)	指定除了一般結果外，查詢結果是否還要包含預先定義結果。依預設會啟用預先定義鏈結。
setSpellCorrectionEnabled (Boolean value)	指定查詢結果是否要包含查詢的拼寫校正建議。依預設會停用拼寫校正。
setResultCategoriesDetailLevel (int detailLevel)	指定查詢結果的必要種類明細層級。如果啟用種類屬性 (RETURN_RESULT_CATEGORIES)，則使用這個方法。預設值是 RESULT_CATEGORIES_ALL。

查詢語法

企業搜尋使用下列語法來處理查詢。

搜尋字元

企業搜尋語法使用特殊字元來使搜尋查詢更精確。

字元	用法
+	在字詞前面放上加號 (+)，表示文件必須包含該字詞才算符合。直接在字詞前面鍵入加號。 查詢： +computer +software 結果： 這個查詢會傳回包括字詞 computer 及 software 的文件。
-	在字詞前面放上減號 (-)，表示文件必須不含該字詞才算符合。直接在字詞前面鍵入減號。 查詢： computer -hardware 結果： 這個查詢會傳回包括字詞 computer 但不包括 hardware 的文件。

字元	用法
^	<p>在字詞前面放上曲折符號 (^)，表示文件除了必須包含該字詞（在查詢中），還必須至少有一個前面未加曲折符號 (^) 的字詞。</p> <p>查詢： ^computer science software</p> <p>結果： 這個查詢會傳回包括字詞 computer 而且形式為 computer science、computer software 或 computer science software 的文件。</p>
()	<p>使用括弧 ()，表示文件必須包含括弧中一或多個字詞才算符合。</p> <p>請勿在括弧中使用加號 (+) 或減號 (-)。</p> <p>使用 OR 或垂直線 () 來隔開括弧中的字詞。</p> <p>查詢： (computer OR software) 或 (computer software)</p> <p>結果： 這個查詢會尋找包括 computer、software 或包括這兩個字詞 computer software 的文件。</p>
" "	<p>使用引號 ("")，表示文件必須包含引號中的片語全文才算符合。</p> <p>查詢： "computer software programming"</p> <p>結果： 這個查詢會尋找包括確切字詞 computer software programming 的文件。</p>
~	<p>在字詞前面加上 ~ 字元符號，表示文件必須包含語言基本詞形（也稱為詞根或字幹）與該字詞相同的字詞才算符合。</p> <p>查詢： "~apples"</p> <p>結果： 這個查詢會尋找包括字詞 apples 或 apple 的文件，因為 apple 是 apples 的基本詞形。</p>
=	<p>在字詞前面加上等號 (=)，表示文件必須包含與該字詞完全相符的字詞才算符合。</p> <p>查詢： "=apples"</p> <p>結果： 這個查詢會尋找包括字詞 apples，但不包括 apple 的文件。</p>
site:text	<p>如果您搜尋包含 Web 內容的集合，可使用 site 關鍵字來搜尋特定的網域。例如，您可以傳回特定網站的所有網頁。</p> <p>請勿將字首 http:// 包括在網站查詢中。</p> <p>查詢： +laptop site:www.ibm.com</p> <p>結果： 這個查詢會尋找 www.ibm.com 網域中包含單字 laptop 的所有文件。</p>

字元	用法
<code>url:text</code>	<p>如果您搜尋包含 Web 內容的集合，可使用 URL 關鍵字來尋找 URL 中任何位置包含特定單字的文件。</p> <p>查詢： <code>URL:support</code></p> <p>結果： 這個查詢會尋找 URL 中任何位置包含 <code>support</code> 的文件，例如 <code>http://www.ibm.com/support/fr/</code>。</p>
<code>link:text</code>	<p>如果您搜尋包含 Web 內容的集合，可使用 <code>link</code> 關鍵字來尋找至少包含一個特定網頁鏈結的文件。</p> <p>查詢： <code>link:http://www.ibm.com/us</code></p> <p>結果： 這個查詢會尋找包括一或多個 <code>http://www.ibm.com/us</code> 網頁鏈結的所有文件。</p>
<code>field:text</code>	<p>如果集合中的文件包括欄位（如關聯式資料庫中的直欄），而且集合管理員已設定可用欄位名稱來搜尋這些欄位，您就可以查詢集合中的特定欄位。</p> <p>查詢： <code>lastname:smith div:software</code></p> <p>結果： 這個查詢會傳回 Software 部門 (<code>div:software</code>) 姓 Smith (<code>lastname:smith</code>) 員工的所有相關文件。</p>
<code>docid:documentid</code>	<p>使用 <code>docid</code> 關鍵字可尋找具有特定 URL 或文件 ID 的文件。</p> <p>查詢： <code>docid:http://www.ibm.com/solutions/us/</code> 或 <code>docid:http://www.ibm.com/products/us/</code></p> <p>結果： 這個查詢會尋找 URL 為 <code>docid:http://www.ibm.com/solutions/us/</code> 或 <code>docid:http://www.ibm.com/products/us/</code> 的所有文件。</p>
<code>taxonomy_ID::category_ID</code>	<p>若要搜尋的集合包含由企業搜尋分類程式產生的種類，可使用種類字詞來尋找屬於特定分類程式所用分類架構中特定種類的文件。</p> <p>查詢： <code>scopes::research "computer science"</code></p> <p>結果： 這個查詢會尋找屬於 <code>research</code> 範圍種類並包含詞組 <code>computer science</code> 的所有文件。</p>
<code>\$source::source_type</code>	<p>使用 <code>\$source</code> 關鍵字可尋找來自特定來源類型的文件。若集合中包含多個來源的文件，來源查詢便很有用。</p> <p>查詢： <code>\$source::DB2 "computer science"</code></p> <p>結果： 這個查詢會尋找包含詞組 <code>computer science</code> 的 DB2 文件。</p>
<code>\$language::language_id</code>	<p>使用 <code>\$language</code> 關鍵字可尋找用特定語言撰寫的文件。</p> <p>查詢： <code>\$language::en "computer science"</code></p> <p>結果： 這個查詢會尋找包含詞組 <code>computer science</code> 的英文文件。</p>

字元	用法
<code>\$doctype::document_type</code>	<p>使用 <code>\$doctype</code> 關鍵字可尋找特定文件格式的文件。</p> <p>查詢： <code>\$doctype:application/pdf "computer science"</code></p> <p>結果： 這個查詢會尋找包含詞組 <code>computer science</code> 的 Adobe Acrobat 文件。</p>
<code>#field::=value</code>	<p>使用數字限制語法可尋找數字欄位的值等於指定數字的文件。</p> <p>查詢： <code>#price:=1700 laptop</code></p> <p>結果： 這個查詢會尋找包含 <code>laptop</code> 而且 <code>price</code> 欄位值等於 1700 的文件。</p>
<code>#field::>value</code>	<p>使用數字限制語法可尋找數字欄位的值大於指定數字的文件。</p> <p>查詢： <code>#price:>1700 laptop</code></p> <p>結果： 這個查詢會尋找包含 <code>laptop</code> 而且 <code>price</code> 欄位值大於 1700 的文件。</p>
<code>#field::<value</code>	<p>使用數字限制語法可尋找數字欄位的值小於指定數字的文件。</p> <p>查詢： <code>#price:<1700 laptop</code></p> <p>結果： 這個查詢會尋找包含 <code>laptop</code> 而且 <code>price</code> 欄位有個值小於 1700 的文件。</p>
<code>#field::>=value</code>	<p>使用數字限制語法可尋找數字欄位的值大於或等於指定數字的文件。</p> <p>查詢： <code>#price:>=1700 laptop</code></p> <p>結果： 這個查詢會尋找包含 <code>laptop</code> 而且 <code>price</code> 欄位值大於或等於 1700 的文件。</p>
<code>#field::<=value</code>	<p>使用數字限制語法可尋找數字欄位的值小於或等於指定數字的文件。</p> <p>查詢： <code>#price:<=1700 laptop</code></p> <p>結果： 這個查詢會尋找包含 <code>laptop</code> 而且 <code>price</code> 欄位值小於或等於 1700 的文件。</p>
<code>#field::>value1<value2</code>	<p>使用數字限制語法可尋找特定範圍內的文件。</p> <p>查詢： <code>#price:>1700<3900 laptop</code></p> <p>結果： 這個查詢會尋找包含 <code>laptop</code> 而且 <code>price</code> 欄位值大於 1700 但小於 3900 的文件。</p>
<code>#field::>=value1<=value2</code>	<p>使用數字限制語法可尋找特定範圍內的文件。</p> <p>查詢： <code>#price:>=1700<=3900 laptop</code></p> <p>結果： 這個查詢會尋找包含 <code>laptop</code> 而且 <code>price</code> 欄位值大於或等於 1700 但小於或等於 3900 的文件。</p>

字元	用法
#field::>value1<=value2	<p>使用數字限制語法可尋找特定範圍內的文件。</p> <p>查詢： #price:>1700<=3900 laptop</p> <p>結果： 這個查詢會尋找包含 laptop 而且 price 欄位值大於 1700 但小於或等於 3900 的文件。</p>
#field::>=value1<value2	<p>使用數字限制語法可尋找特定範圍內的文件。</p> <p>查詢： #price:>=1700<3900 laptop</p> <p>結果： 這個查詢會尋找包含 laptop 而且 price 欄位值大於或等於 1700 但小於 3900 的文件。</p>
ACL constraints: (ACL tokens)	<p>基於安全考量，存取控制限制不能像其他限制一樣指定在查詢字串中。請使用 Query 介面的 setACLConstraints(String aclConstraints) 方法來指定查詢的存取控制限制。指定 ACL 限制時，可以使用括弧、加號 (+)、減號 (-) 及曲折符號 (^)，而且這些符號的用法與一般查詢語法相同。</p> <p>查詢： thinkpad ACL constraints: (john_m dev_group)</p> <p>結果： 這個查詢會尋找包括單字 thinkpad 及 ACL 符記 john_m or dev_group 的文件。</p>

建置 Java 原始碼

建置 Java 原始碼之前，您必須安裝並配置 Apache ANT（一種 Java 建置工具）。如需安裝及配置 Apache ANT 的更多相關資訊，請造訪 <http://ant.apache.org/>。

若要建置 Java 原始碼：

- 在命令行中，導覽到下列任一目錄：
 - \$install_root/samples/siapi（例如：/opt/IBM/es/samples/siapi）
 - \$install_root/samples//datalistener（例如：/opt/IBM/es/samples//datalistener）

這些目錄都包括 build.xml 檔案，以便 ANT 用來建置檔案。

- 鍵入 ant 並按 Enter 鍵。

編譯 Java 原始碼後，您會收到 ADD TEXT HERE 訊息。

SearchExample 類別

SearchExample 類別提供一個簡單的範例，說明將搜尋提給企業搜尋伺服器所需要的最低需求。

範例

```
public class SearchExample {
    private String hostname    = "localhost";
    private String portNumber  = "80";
    private String queryString;
    private String collectionId;
    private String applicationName;
```

```

private String applicationPassword;

public void execute() throws Exception {
    // obtain the OmniFind specific SI-API Search factory implementation
    SearchFactory factory =
        SiapiSearchImpl.createSearchFactory("com.ibm.es.api.search.
RemoteSearchFactory");

    // create a valid Application ID that will be used
    // by the Search Node to authorize this
    // access to the collection
    ApplicationInfo appinfo = factory.createApplicationInfo(applicationName);
    appinfo.setPassword(applicationPassword);

    // create a new Properties object.
    Properties config = new Properties();
    // set the hostname of the OmniFind Search Node. The hostname
    // should be the hostname that is assigned to the
    // Search Node's WebSphere installation
    config.setProperty("hostname", hostname);
    // set the port number - the
    // default value is port 80 (the web server port).
    config.setProperty("port", portNumber);
    // set the locale of the "client". This value is
    // used by the underlying SI-API implementation to
    // return translated error messages to the client
    // for the appropriate language.
    // NOTE: this should be the 2 letter ISO-639
    // language code followed by an underscore
    // followed by the ISO-639 2 letter country code
    config.setProperty("locale", "en_US");

    // obtain the Search Service implementation
    SearchService searchService =
        factory.getSearchService(config);

    // obtain a Searchable object to the specified
    // collection ID
    Searchable searchable = searchService.getSearchable(appinfo, collectionId);
    if(searchable == null) {
        System.out.println("Failed to get a searchable for collection:
" + collectionId);
        return;
    }

    // create a new Query object using the specified
    // query string
    Query q = factory.createQuery(queryString);
    q.setQueryLanguage("en_US");

    // execute the search by calling the Searchable's
    // search method. A SI-API ResultSet object will
    // be returned
    ResultSet rset = searchable.search(q);
    if(rset != null) {
        // get the array of results from the ResultSet
        Result r[] = rset.getResults();
        if(r != null) {
            // walk the results list and print out the
            // document identifier
            for(int k = 0; k < r.length; k++) {
                System.out.println("Result " + k + ": " + r[k].getDocumentID());
            }
        }
    }
}

```

```

public static void main(String[] args) throws Exception {
    SearchExample sc = new SearchExample();
    if (args.length < 5) {
        System.out.println("Usage: SearchExample <queryString> <collection id>
<application name>" +
            "<application password> <hostname> <port>");
        System.out.println("Example Usage: SearchExample lotus coll appl
password localhost 80");
        return;
    }
    sc.queryString = args[0];
    sc.collectionId = args[1];
    sc.applicationName = args[2];
    sc.applicationPassword = args[3];
    sc.hostname = args[4];
    sc.portNumber = args[5];
    sc.execute();
}
}

```

AdvancedSearchExample 類別

AdvancedSearchExample 類別是使用進階查詢設定及結果處理選項的範例。

範例

```

public class AdvancedSearchExample {
    private String hostname = "localhost";
    private String portNumber = "80";
    private String queryString;
    private String collectionId;
    private String applicationName;
    private String applicationPassword;

    public void execute() throws Exception {
        // obtain the OmniFind specific SIAPI Search factory implementation
        SearchFactory factory =
            SiapiSearchImpl.createSearchFactory("com.ibm.es.api.search.
RemoteSearchFactory");

        // create a valid Application ID that will be used
        // by the Search Node to authorize this
        // access to the collection
        ApplicationInfo appinfo = factory.createApplicationInfo(applicationName);
        appinfo.setPassword(applicationPassword);

        // create a new Properties object.
        Properties config = new Properties();
        // set the hostname of the OmniFind Search Node. The hostname
        // should be the hostname that is assigned to the
        // Search Node's WebSphere installation
        config.setProperty("hostname", hostname);
        // set the port number - the
        // default value is port 80 (the web server port).
        config.setProperty("port", portNumber);
        // set the locale of the "client". This value is
        // used by the underlying SIAPI implementation to
        // return translated error messages to the client
        // for the appropriate language.
        // NOTE: this should be the 2 letter ISO-639
        // language code followed by an underscore
        // followed by the ISO-639 2 letter country code
        config.setProperty("locale", "en_US");

        // obtain the Search Service implementation
        SearchService searchService =

```

```

        factory.getSearchService(config);

        // obtain a Searchable object to the specified
        // collection ID
        Searchable searchable = searchService.getSearchable(appinfo, collectionId);
        if(searchable == null) {
            System.out.println("Failed to get a searchable for collection:
" + collectionId);
            return;
        }

        // create a new Query object using the specified
        // query string
        Query q = factory.createQuery(queryString);

        // set the result range we want to access on this query
        q.setRequestedResultRange(0, 20);
        q.setQueryLanguage("en_US");
        // designate that we do not want the Description
        // field returned for each result
        q.setReturnedAttribute(Query.RETURN_RESULT_DESCRIPTION, false);

// execute the search by calling the Searchable's
// search method. A SI-API ResultSet object will
// be returned
ResultSet rset = searchable.search(q);
if(rset != null) {
    System.out.println("");
    System.out.println("Estimated results:
" + rset.getEstimatedNumberOfResults());
    System.out.println("Available results:
" + rset.getAvailableNumberOfResults());
    System.out.println("Evaluation time: " + rset.getQueryEvaluationTime());
    System.out.println("");
    Result pr[] = rset.getPredefinedResults();
    if(pr != null) {
        // walk the predefinedResults list and print out the
        // document identifier and document title
        for(int k = 0; k < pr.length; k++) {
            System.out.println("PredefinedResult " + k + ":
" + pr[k].getDocumentID());
            System.out.println("\tTitle: " + pr[k].getTitle());
            System.out.println("\tDescription: " + pr[k].getDescription());
        }
    }
    System.out.println("");
    SpellCorrection sc[] = rset.getSpellCorrections();
    if(sc != null) {
        // walk the list of returned spelling corrections
        // and print out the query sub string and the
        // spelling suggestions
        for(int k = 0; k < sc.length; k++) {
            System.out.println("SpellCorrection " + k + ": " + sc[k].
getQuerySubstring());
            String[] corrections = sc[k].getSuggestions();
            if(corrections != null) {
                for(int s = 0; s < corrections.length; s++) {
                    System.out.println("Suggestion " + s + ":
" + corrections[s]);
                }
            }
        }
    }
    System.out.println("");
    // get the array of results from the ResultSet
    Result r[] = rset.getResults();
    if(r != null) {

```

```

        // walk the results list
        for(int k = 0; k < r.length; k++) {
            // print out all predefined field values.
            // NOTE: Description will be "null"
            // since we modified the Query above
            // to NOT return the Description field.
            System.out.println("Result " + k + ": " + r[k].getDocumentID());
System.out.println("\tScore: " + r[k].getScore() + "%");
            System.out.println("\tTitle: " + r[k].getTitle());
System.out.println("\tLanguage: " + r[k].getLanguage());
System.out.println("\tType: " + r[k].getDocumentType());
            System.out.println("\tSource: " + r[k].getDocumentSource());
            System.out.println("\tDate: " + r[k].getDate());
System.out.println("\tDescription: " + r[k].getDescription());
            // walk any categories that this result belongs to
            ResultCategory[] cats = r[k].getCategories();
            if(cats != null) {
                for(int s = 0; s < cats.length; s++) {
                    CategoryInfo info = cats[s].getInfo();
                    // print out the taxonomy id, the category id, and the
category label
                    // for each ResultCategory returned
                    System.out.println("\tTaxonomy : " + cats[s].
getTaxonomyID());
                    if(info != null) {
                        System.out.println("\tCategory " + s + ":
" + info.getID());
                        System.out.println("\t\tLabel " + s + ":
" + info.getLabel());
                    }
                }
            }
            // print out any additional field names and their
// values (document metadata)
            NameValuePair[] fields = r[k].getFields();
            if(fields != null) {
                for(int s = 0; s < fields.length; s++) {
                    NameValuePair nvp = fields[s];
                    if(nvp != null) {
                        System.out.print("\tField " + nvp.getName() + ":
" + nvp.getValue());
                    }
                }
            }
System.out.println("");
        }
    }
}

public static void main(String[] args) throws Exception {
    AdvancedSearchExample sc = new AdvancedSearchExample();
    if (args.length < 5) {
        System.out.println("Usage: AdvancedSearchExample <queryString>
<collection id> <application name> " +
            "<application password><hostname><port>");
        System.out.println("Example Usage: AdvancedSearchExample lotus coll app1
password localhost 80");
        return;
    }
    sc.queryString = args[0];
    sc.collectionId = args[1];
    sc.applicationName = args[2];
    sc.applicationPassword = args[3];
    sc.hostname = args[4];
}
}

```

```

        sc.portNumber = args[5];
        sc.execute();
    }
}

```

BrowseExample 類別

BrowseExample 類別提供範例，來存取集合的分類架構樹狀結構及列印部份基本導覽屬性。

範例

```

public class BrowseExample {
    private String hostname    = "localhost";
    private String portNumber  = "80";
    private String collectionId;
    private String taxonomyId;
    private String applicationName;
    private String applicationPassword;

    public void execute() throws Exception {
        // obtain the OmniFind specific SIAPI Browse factory implementation
        BrowseFactory factory =
            SiapiBrowseImpl.createBrowseFactory("com.ibm.es.api.browse.RemoteBrowseFactory");

        // create a valid Application ID that will be used
        // by the Search Node to authorize this
        // access to the collection
        ApplicationInfo appinfo = factory.createApplicationInfo(applicationName);
        appinfo.setPassword(applicationPassword);

        // create a new Properties object.
        Properties config = new Properties();
        // set the hostname of the OmniFind Search Node. The hostname
        // should be the hostname that is assigned to the
        // Search Node's WebSphere installation
        config.setProperty("hostname", hostname);
        // set the port number - the
        // default value is port 80 (the web server port).
        config.setProperty("port", portNumber);
        // set the locale of the "client". This value is
        // used by the underlying SIAPI implementation to
        // return translated error messages to the client
        // for the appropriate language.
        // NOTE: this should be the 2 letter ISO-639
        // language code followed by an underscore
        // followed by the ISO-639 2 letter country code
        config.setProperty("locale", "en_US");

        // obtain the Browse service implementation
        BrowseService browseService = factory.getBrowseService(config);

        // get a TaxonomyBrowser for the specified taxonomy id and collection id
        TaxonomyBrowser browser = browseService.getTaxonomyBrowser(appinfo,
            collectionId,
            taxonomyId);
        if(browser == null) {
            System.out.println("Failed to get a taxonomy for taxonomy id:
" + taxonomyId +
                                " from collection: " + collectionId);
            return;
        }

        //Display the Taxonomy Info
        TaxonomyInfo taxonomyInfo = browser.getTaxonomyInfo();
    }
}

```

```

        System.out.println("Taxonomy label: " + taxonomyInfo.getLabel());

        // get the root category
        Category rootCategory = browser.getRootCategory();

        // display the root category
        System.out.println("Root category label: " + rootCategory.getInfo().
getLabel());
        System.out.println("child categories:");
        CategoryInfo[] childrenInfo = rootCategory.getChildren();
        for (int i = 0; i < childrenInfo.length; i++) {
            System.out.println("\t" + childrenInfo[i].getLabel());
        }

        // Now get the root's first child category
        Category childCategory = browser.getCategory(rootCategory.getChildren()
[0].getID());

        // Display the child category and it's path from root
        System.out.println("Root's first child's label: " + childCategory.getInfo()
.getLabel());
        System.out.println("It's path from root is : ");
        CategoryInfo[] pathFromRoot = childCategory.getPathFromRoot();
        for (int i = 0; i < pathFromRoot.length; i++) {
            System.out.println("-->" + pathFromRoot[i].getLabel());
        }
    }

    public static void main(String[] args) throws Exception {
        BrowseExample sc = new BrowseExample();
        if (args.length < 5) {
            System.out.println("Usage: BrowseExample <taxonomy id> <collection id>
<applicationname>");
            System.out.println("Example Usage: BrowseExample tax1 col1 Default password
localhost 80");
            return;
        }
        sc.taxonomyId = args[0];
        sc.collectionId = args[1];
        sc.applicationName = args[2];
        sc.applicationPassword = args[3];
        sc.hostname = args[4];
        sc.portNumber = args[5];
        sc.execute();
    }
}

```

SI-API 呼叫及方法

SI-API 使用下列類別及介面。

SiapiVersion 類別

SiapiVersion 類別用來管理 SI-API 版本。

getSiapiVersion 方法

傳回 SI-API 的版本。

語法

```
public String getSiapiVersion()
```


SiapiSearchImpl 類別

SiapiSearchImpl 類別可取得搜尋 factory，以實作企業搜尋 SI-API。

createSearchFactory 方法

以實作類別名稱建立 SI-API 實作所需的搜尋 factory。

使用 `com.ibm.es.api.search.RemoteSearchFactory` 呼叫這個方法。

語法

```
public static SearchFactory createSearchFactory(String factoryImplClassName)
    throws SiapiException
```

參數

factoryImplClassName

實作 factory 類別的完整名稱。

傳回

SI-API 實作所需的搜尋 factory。

SearchFactory 介面

SearchFactory 介面是 SI-API 搜尋實作的進入點。

getSearchService 方法

取得某特定應用程式及特定搜尋伺服器的搜尋服務物件。

語法

```
public SearchService getSearchService(Properties config) throws SiapiException
```

參數

config

企業搜尋支援三種配置屬性：

- `hostname`（必要） - 企業搜尋搜尋伺服器的主機名稱。
- `port`（選用） - 企業搜尋搜尋伺服器的埠號（預設埠號是 80）。
- `locale`（選用） - 用來傳回轉譯錯誤訊息的語言環境設定（預設值是預設 java 語言環境）。

傳回

搜尋服務。

擲出

如果無法取得搜尋服務，就會擲出 `SiapiException`。

createQuery 方法

使用指定查詢文字建立 Query 物件。

查詢文字必須符合企業搜尋所支援的查詢語法。請參閱第 8 頁的『查詢語法』，以取得建立查詢的相關資訊。

語法

```
public Query createQuery(String queryText) throws SiapiException
```

參數

queryText

查詢文字。

傳回

新的查詢物件。

createApplicationInfo 方法

建立指定應用程式 ID 的應用程式資訊物件。取得 `ApplicationInfo` 物件後，您就可以使用 `ApplicationInfo.setPassword(String)` 或 `ApplicationInfo.setToken(String)` 來設定應用程式的密碼或安全性符記。

語法

```
public ApplicationInfo createApplicationInfo(java.lang.String applicationID)
                                     throws SiapiException
```

擲出

如果 `applicationID` 無效，就會擲出 `SiapiException`。

getVersion 方法

傳回特定 SI-API 實作的版本。每個 SI-API 實作的版本資料都可進行不同的管理。在此傳回的值可能與 SI-API 的版本不同。

語法

```
public java.lang.String getVersion()
```

CategoryInfo 介面

`CategoryInfo` 介面提供種類的描述資訊。

getID 方法

傳回種類 ID。

語法

```
public java.lang.String getID()
```

getLabel 方法

傳回種類標籤。

語法

```
public java.lang.String getLabel()
```

CollectionInfo 介面

`CollectionInfo` 介面提供集合的描述資訊。

getCollectionInfo 方法

傳回 Searchable 介面的 ID 及標籤。

語法

```
public CollectionInfo getCollectionInfo()
```

傳回

有集合 ID 及標籤的 CollectionInfo 介面。

getID 方法

傳回集合 ID。

語法

```
public java.lang.String getID()
```

getLabel 方法

傳回集合標籤。

語法

```
public java.lang.String getLabel()
```

FieldInfo 介面

FieldInfo 介面代表欄位定義。

欄位就是文件內容或中間資料的特定文字部份。

getID 方法

傳回欄位名稱。

語法

```
public java.lang.String getID()
```

getAvailableFields 方法

傳回這個集合文件所定義的中間資料欄位資訊。

語法

```
public FieldInfo[] getAvailableFields()
```

傳回

欄位資訊物件陣列。

isContentSearchable 方法

如果這種類型的欄位可搜尋內容，會傳回 True。

可搜尋內容的欄位指的是有可以搜尋的文字而且使用非欄位搜尋查詢的欄位。例如，查詢單字 apple，就會傳回可搜尋內容欄位中含有單字 apple 的所有文件。

語法

```
public boolean isContentSearchable()
```

傳回

如果這種類型的欄位可搜尋內容，會傳回 `True`；反之，則傳回 `False`。

isFieldSearchable 方法

如果這種類型的欄位屬於可搜尋的欄位，會傳回 `True`。

語法

可搜尋欄位指的是欄位搜尋查詢可以搜尋到其文字的欄位。例如，查詢 `keywords:apple`，就會傳回可搜尋欄位關鍵字含有單字 `apple` 的所有文件。

```
public boolean isFieldSearchable()
```

傳回

如果這種類型的欄位屬於可搜尋的欄位，會傳回 `True`；反之，則傳回 `False`。

isReturnable 方法

如果這種類型的欄位內容可以與搜尋結果一起傳回，則傳回 `True`。

例如，如果某文件有可傳回的 `author` 中間資料欄位，而該文件作為查詢結果傳回，則 `author` 欄位的內容將與結果一起傳回。

使用 `Query` 介面的 `setReturnedFields` 方法來指定要與查詢結果一起傳回的欄位名稱。使用 `getFields` 方法來取得 `Result` 物件中傳回欄位的內容。

語法

```
public boolean isReturnable()
```

傳回

如果這種類型的欄位內容可以與搜尋結果一起傳回，會傳回 `True`；反之，則傳回 `False`。

isParametric 方法

如果這種類型的欄位屬於參數式欄位，會傳回 `True`。

參數式欄位指的是其值可以轉換為數值的欄位。若要建立查詢限制，將查詢結果限制在指定參數式欄位值範圍，請參閱第 8 頁的『查詢語法』。

語法

```
public boolean isParametric()
```

傳回

如果這種類型的欄位屬於參數式欄位，會傳回 `True`；反之，則傳回 `False`。

Searchable 介面

`Searchable` 介面可讓您搜尋集合的索引，並提供集合的相關資訊。

ATTRIBUTE_LANGUAGE 常數

將此常數傳給 `getAvailableAttributeValues` 方法，便可擷取文件語言清單，將查詢限制在此 `Searchable` 介面

語法

```
public static final int ATTRIBUTE_LANGUAGE
```

ATTRIBUTE_SOURCE 常數

將此常數傳給 `getAvailableAttributeValues` 方法，便可擷取文件來源清單，將查詢限制在此 `Searchable` 介面。

語法

```
public static final int ATTRIBUTE_SOURCE
```

ATTRIBUTE_DOCTYPE 常數

將此常數傳給 `getAvailableAttributeValues` 方法，便可擷取文件來源清單，將查詢限制在此 `Searchable` 介面

語法

```
public static final int ATTRIBUTE_DOCTYPE
```

search 方法

執行查詢並傳回結果集。

語法

```
public ResultSet search(Query query) throws SiapiException
```

參數

query

您要搜尋的查詢。

傳回

包含搜尋結果的 `ResultSet` 物件。

擲出

如果無法執行搜尋作業，就會擲出 `SiapiException`。

count 方法

傳回所有符合指定查詢的文件精確計數。計數所包括的文件內含所有必要的查詢字詞，完全不含禁止的字詞。

語法

```
public int count(Query query) throws SiapiException
```

參數

query

要搜尋的查詢。

傳回

符合指定查詢的文件數目。

擲出

如果無法執行計數作業，就會擲出 `SiapiException`。

getSpellCorrections 方法

傳回指定查詢字串的拼寫校正建議清單；如果沒有可用的建議，則傳回 `NULL`。

如果拼寫校正已啟用，請參閱 `Query` 介面中的方法 - 第 36 頁的『`setSpellCorrectionEnabled` 方法』。

語法

```
public SpellCorrection[] getSpellCorrections()
```

參數

queryText

要檢查的查詢字串。

傳回

傳回指定查詢字串的拼寫校正建議清單。

getDefaultLanguage 方法

擷取這個 `Searchable` 介面用來處理查詢的預設語言。

語法

```
public String getDefaultLanguage()
```

getAvailableAttributeValues 方法

傳回值清單，以用來建立指定類型的屬性查詢限制。

語法

```
public java.lang.String[] getAvailableAttributeValues(int attributeType)
```

參數

attributeType

屬性類型。

傳回

傳回指定屬性類型的可用值清單。

註： 部份屬性類型所傳回的清單可能不完整或不存在。在這種情況下，會傳回空值。

有效的屬性類型是：

- `ATTRIBUTE_DOCTYPE`
- `ATTRIBUTE_LANGUAGE`
- `ATTRIBUTE_SOURCE`

例如，呼叫 `getAvailableValues(Searchable.ATTRIBUTE_SOURCE)` 會傳回這個集合的文件來源清單。如需其他資訊，請參閱第 8 頁的『查詢語法』。

getAvailableFields 方法

傳回這個集合文件所定義的中間資料欄位資訊。

語法

```
public FieldInfo[] getAvailableFields()
```

傳回

欄位資訊物件陣列。

setProperty 方法

設定可搜尋屬性的值。

使用這些屬性，您可以修改 `Searchable` 介面的行爲。若要瞭解支援哪些屬性，請呼叫『`getProperties` 方法』。

限制: 企業搜尋並不支援所有屬性，且呼叫這個方法時會擲出異常。

語法

```
public void setProperty(String name,  
                        String value) throws SiapiException
```

參數

name

可搜尋屬性的名稱。

value

可搜尋屬性的值。

擲出

如果這時無法修改屬性，或值不合法，就會擲出 `SiapiException`。

getProperty 方法

傳回可搜尋屬性的值。

語法

```
public String getProperty(String name)
```

參數

name

要擷取的可搜尋屬性名稱。

getProperties 方法

傳回所有可搜尋的屬性；如果沒有，則傳回 `NULL`。

語法

```
public Properties getProperties()
```

getCollectionInfo 方法

傳回 Searchable 介面的 ID 及標籤。

語法

```
public CollectionInfo getCollectionInfo()
```

傳回

有集合 ID 及標籤的 CollectionInfo 介面。

getAvailableSearchables 方法

取得集合的所有參照。

語法

```
public Searchable[] getAvailableSearchables(ApplicationInfo appInfo)  
    throws SiapiException
```

參數

appInfo

要求端應用程式的鑑別及授權資訊。

傳回

集合的 Searchable 介面。

擲出

如果無法執行動作，就會擲出 SiapiException。

getSearchable 方法

依集合 ID 取得集合參照。

語法

```
public Searchable getSearchable(ApplicationInfo appInfo,  
    String collectionID) throws SiapiException
```

參數

appInfo

要求端應用程式的鑑別及授權資訊。

collectionID

集合 ID。

傳回

Searchable。

擲出

如果無法取得 Searchable 介面參照，就會擲出 SiapiException。

SearchService 介面

使用這個介面來取得可搜尋的物件。

getAvailableSearchables 方法

取得集合的所有參照。

語法

```
public Searchable[] getAvailableSearchables(ApplicationInfo appInfo)  
    throws SiapiException
```

參數

appInfo

要求端應用程式的鑑別及授權資訊。

傳回

集合的 Searchable 介面。

擲出

如果無法執行動作，就會擲出 SiapiException。

getSearchable 方法

依集合 ID 取得集合參照。

語法

```
public Searchable getSearchable(ApplicationInfo appInfo,  
                                String collectionID) throws SiapiException
```

參數

appInfo

要求端應用程式的鑑別及授權資訊。

collectionID

集合 ID。

傳回

Searchable。

擲出

如果無法取得 Searchable 介面參照，就會擲出 SiapiException。

ApplicationInfo 介面

ApplicationInfo 介面控制用來授權存取集合的資訊。

getID 方法

傳回應用程式 ID。

語法

```
public java.lang.String getId()
```

getPassword 方法

傳回與應用程式相關聯的密碼。

語法

```
public String getPassword()
```

getToken 方法

傳回與應用程式相關聯的符記值。

語法

```
public String getToken()
```

setPassword 方法

設定與應用程式相關聯的密碼。請在需要密碼進行鑑別的機密保護環境中使用這個方法。

語法

```
public void setPassword(String password) throws SiapiException
```

參數

password

您要設定的密碼。

擲出

如果密碼無效，就會擲出 `SiapiException`。

setToken 方法

設定與應用程式相關聯的符記。

請在需要傳遞符記的機密保護環境（如單一登入環境）中使用這個方法。

語法

```
public void setToken(String token) throws SiapiException
```

參數

token

要設定的符記

擲出

如果符記無效，就會擲出 `SiapiException`。

createApplicationInfo 方法

建立指定應用程式 ID 的應用程式資訊物件。取得 `ApplicationInfo` 物件後，您就可以使用 `ApplicationInfo.setPassword(String)` 或 `ApplicationInfo.setToken(String)` 來設定應用程式的密碼或安全性符記。

語法

```
public ApplicationInfo createApplicationInfo(java.lang.String applicationID)
                                     throws SiapiException
```

擲出

如果 applicationID 無效，就會擲出 SiapiException。

createApplicationInfo 方法

建立指定應用程式 ID 的應用程式資訊物件。

取得 ApplicationInfo 物件後，您就可以使用 setPassword 或 setToken 來設定應用程式的密碼或安全性符記。

語法

```
public ApplicationInfo createApplicationInfo(String applicationID)
                                     throws SiapiException
```

參數

ID 應用程式的 ID。

Query 介面

Query 介面控制查詢行為。

createQuery 方法

使用指定查詢文字建立 Query 物件。

查詢文字必須符合企業搜尋所支援的查詢語法。請參閱第 8 頁的『查詢語法』，以取得建立查詢的相關資訊。

語法

```
public Query createQuery(String queryText) throws SiapiException
```

參數

queryText

查詢文字。

傳回

新的查詢物件。

SORT_KEY_RELEVANCE 常數

將此常數傳給 setSortKey 方法，按關聯性排序結果。

語法

```
public static final java.lang.String SORT_KEY_RELEVANCE
```

SORT_KEY_DATE 常數

將此常數傳給 setSortKey 方法，按日期排序結果。

語法

```
public static final java.lang.String SORT_KEY_DATE
```

SORT_KEY_NONE 常數

將此常數傳給 `setSortKey` 方法，以取得未排序的結果。

語法

```
public static final java.lang.String SORT_KEY_NONE
```

SORT_ORDER_ASCENDING 常數

將此常數傳給 `setSortOrder` 方法，按遞增次序來排序結果。

語法

```
public static final int SORT_ORDER_ASCENDING
```

SORT_ORDER_DESCENDING 常數

將此常數傳給 `setSortOrder` 方法，按遞減次序來排序結果。

語法

```
public static final int SORT_ORDER_DESCENDING
```

RESULT_CATEGORIES_ALL 常數

將此常數傳給 `setResultCategoriesDetailLevel` 方法，以設定完整的結果種類明細層級。

語法

```
public static final int RESULT_CATEGORIES_ALL
```

RESULT_CATEGORIES_NO_PATH_TO_ROOT 常數

將此常數傳給 `setResultCategoriesDetailLevel` 方法，以設定部份的結果種類明細層級。

語法

```
public static final int RESULT_CATEGORIES_NO_PATH_TO_ROOT
```

RETURN_RESULT_DESCRIPTION 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回結果說明。

語法

```
public static final int RETURN_RESULT_DESCRIPTION
```

RETURN_RESULT_TITLE 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回結果標題。

語法

```
public static final int RETURN_RESULT_TITLE
```

RETURN_RESULT_FIELDS 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回儲存結果的欄位。

語法

```
public static final int RETURN_RESULT_FIELDS
```

RETURN_RESULT_CATEGORIES 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回結果種類。

語法

```
public static final int RETURN_RESULT_CATEGORIES
```

RETURN_RESULT_TYPE 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回結果文件類型。

語法

```
public static final int RETURN_RESULT_TYPE
```

RETURN_RESULT_SOURCE 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回結果文件來源。

語法

```
public static final int RETURN_RESULT_SOURCE
```

RETURN_RESULT_LANGUAGE 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回結果語言。

語法

```
public static final int RETURN_RESULT_LANGUAGE
```

RETURN_RESULT_DATE 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回結果日期。

語法

```
public static final int RETURN_RESULT_DATE
```

RETURN_RESULT_SCORE 常數

將此常數傳給 `setReturnedAttribute` 方法，以啟用或停用傳回結果分數。

語法

```
public static final int RETURN_RESULT_SCORE
```

getText 方法

傳回這個查詢的文字。

請參閱第 8 頁的『查詢語法』，以取得詳細資訊。

語法

```
public String getText()
```

setText 方法

設定這個查詢的文字。

請參閱第 8 頁的『查詢語法』，以取得詳細資訊。

語法

```
public void setText(String text) throws SiapiException
```

參數

text

這個查詢的新文字。

getQueryLanguage 方法

傳回之前透過呼叫 `setQueryLanguage` 方法為這個查詢所指定的語言；如果未指定語言，則傳回 `NULL`。

語法

```
public String getQueryLanguage()
```

getReturnedFields 方法

傳回要傳給這個查詢結果的欄位名稱（之前已透過呼叫 `setReturnedFields` 方法指定）。如果沒有指定欄位，則傳回 `NULL`。

每份文件的某些欄位元素可以標記為可傳回，表示這些欄位元素會儲存在該文件的索引中。當文件出現在搜尋結果中時，使用者便可還原這些欄位元素。

如果沒有指定傳回的欄位，這個方法會傳回 `NULL`，而且查詢結果將不會傳回欄位。

您可以透過呼叫『`resetReturnedFields` 方法』，明確指定要擷取哪些欄位。

語法

```
public String[] getReturnedFields()
```

setReturnedFields 方法

設定傳回的中間資料欄位名稱。依預設會傳回所有欄位。

在呼叫這個方法之前，請使用 `setReturnedAttribute` 方法來啟用擷取中間資料欄位（使用 `RETURN_RESULT_FIELDS` 常數）。

語法

```
public void setReturnedFields(String[] fieldNames) throws SiapiException
```

參數

fieldNames

必要欄位名稱

resetReturnedFields 方法

重設這個查詢傳回的中間資料欄位資料。

語法

```
public void resetReturnedFields()
```

getNumRequestedResults 方法

傳回這個查詢所要求的結果筆數。

語法

```
public int getNumRequestedResults()
```

getFirstRequestedResult 方法

傳回這個查詢所要求的第一個結果排序（序數位置）。

例如，假設第一個結果的排序位置是 20，而要求結果筆數是 10，就會擷取排序為 20 到 30 的 10 個結果。排序從 0 開始。

語法

```
public int getFirstRequestedResult()
```

setRequestedResultRange 方法

設定這個查詢所要求的第一個結果排序（序數位置）及要求結果筆數。第一個結果的排序必須是 0，或是要求結果筆數的整倍數。

例如，如果第一個結果的排序位置是 17，而要求結果筆數是 4，就會擷取排序為 17 到 20 的 4 個結果。排序從 0 開始。

語法

```
public void setRequestedResultRange(int fromResult,  
                                     int numberOfResult) throws SiapiException
```

參數

fromResult

第一個要求結果的排序。

numberOfResults

要求結果的筆數（這個參數所容許的最大值是 100）。

getProperty 方法

傳回可搜尋屬性的值。

語法

```
public String getProperty(String name)
```

參數

name

要擷取的可搜尋屬性名稱。

setProperty 方法

設定可搜尋屬性的值。

使用這些屬性，您可以修改 `Searchable` 介面的行為。若要瞭解支援哪些屬性，請呼叫第 25 頁的『`getProperties` 方法』。

限制: 企業搜尋並不支援所有屬性，且呼叫這個方法時會擲出異常。

語法

```
public void setProperty(String name,  
                        String value) throws SiapiException
```

參數

name

可搜尋屬性的名稱。

value

可搜尋屬性的值。

擲出

如果這時無法修改屬性，或值不合法，就會擲出 `SiapiException`。

getProperties 方法

傳回所有可搜尋的屬性；如果沒有，則傳回 `NULL`。

語法

```
public Properties getProperties()
```

getSortKey 方法

傳回排序這個查詢結果所用的鍵值。

預先定義的特殊鍵值如下：

`SORT_KEY_RELEVANCE`

依關聯性排序結果（預設）。

`SORT_KEY_DATE`

依日期排序結果。

`SORT_KEY_NONE`

不排序。

其他有效鍵值包括搜尋引擎支援用來排序的任何欄位名稱。

語法

```
public String getSortKey()
```

getSortPoolSize 方法

傳回搜尋引擎依排序鍵排序的最相關結果筆數。排序儲存區大小的預設值是 500 筆。

語法

```
public int getSortPoolSize()
```


傳回

排序儲存區大小。

setSortPoolSize 方法

設定搜尋引擎依排序鍵排序的最相關結果筆數。

這個方法的預設值視實作而定，而各個 SI-API 實作的設定也可能不盡相同。

語法

```
public void setSortPoolSize(int sortPoolSize) throws SiapiException
```

參數

sortPoolSize

排序儲存區大小

擲出

如果排序儲存區大小是負數或太大，就會擲出 `SiapiException`。

getSortOrder 方法

傳回排序查詢結果的次序。

有效值包括：

SORT_ORDER_ASCENDING

遞增排序。

SORT_ORDER_DESCENDING

遞減排序。

語法

```
public int getSortOrder()
```

傳回

排序。

setSortKey 方法

設定用來排序這個查詢結果的鍵值。

預先定義的特殊鍵值如下：

SORT_KEY_RELEVANCE

依關聯性設定排序結果（預設）。

SORT_KEY_DOCUMENT_ID

依文件 ID 設定排序結果。

SORT_KEY_DATE

依日期設定排序結果。

SORT_KEY_NONE

指定不排序。

其他有效值包括搜尋引擎支援用來排序的任何欄位名稱。

語法

```
public void setSortKey(String sortKey) throws SiapiException
```

參數

sortKey

用來排序的欄位名稱或排序方法

isSiteCollapsingEnabled 方法

如果計算這個查詢結果時應該套用網站收合，即傳回 `True`。

如果最符合的結果有兩筆以上來自相同網站，即會收合結果

語法

```
public boolean isSiteCollapsingEnabled()
```

setSiteCollapsingEnabled 方法

設定這個查詢的網站收合行爲。

語法

```
public void setSiteCollapsingEnabled(boolean value) throws SiapiException
```

參數

value

如果計算這個查詢結果時應套用網站收合，即爲 `True`；反之，則爲 `False`。

isPredefinedResultsEnabled 方法

如果這個查詢傳回預先定義鏈結，則傳回 `True`。

語法

```
public boolean isPredefinedResultsEnabled()
```

setPredefinedResultsEnabled 方法

設定這個查詢的預先定義鏈結行爲。

語法

```
public void setPredefinedResultsEnabled(boolean value) throws SiapiException
```

參數

value

如果這個查詢應傳回預先定義鏈結，即爲 `True`；反之，則爲 `False`

setSpellCorrectionEnabled 方法

設定是否在這個查詢中啓用拼寫校正。

語法

```
public void setSpellCorrectionEnabled(boolean enable) throws SiapiException
```

參數

enable

如果應啓用拼寫校正，即爲 `True`；反之，則爲 `False`

擲出

如果無法設定拼寫校正。

isSpellCorrectionEnabled 方法

如果這個查詢啓用拼寫校正，會傳回 `True`。否則傳回 `False`。

語法

```
public boolean isSpellCorrectionEnabled()
```

setResultCategoriesDetailLevel 方法

設定查詢結果必要的種類明細層級。

語法

使用 `setReturnedAttribute` 方法搭配 `RETURN_RESULT_CATEGORIES` 常數來啓用或停用擷取結果種類。

有效的 `detailLevel` 值包括：

RESULT_CATEGORIES_ALL

傳回每個結果種類及其完整路徑（從根路徑開始）資訊（`RESULT_CATEGORIES_ALL` 是預設值）。

RESULT_CATEGORIES_NO_PATH_TO_ROOT

傳回每個結果種類，但沒有完整路徑（從根路徑開始）資訊（即 `getPathFromRoot` 傳回 `NULL`）。

```
public void setResultCategoriesDetailLevel(int detailLevel)
```

setReturnedAttribute 方法

依預設，會傳回欄位以外的所有結果屬性。

有效的 `attributeType` 值是：

RETURN_RESULT_TITLE

如果 `isReturned` 設定爲 `False`，則 `getTitle` 傳回 `NULL`。

RETURN_RESULT_DESCRIPTION

如果 `isReturned` 設定爲 `False`，則 `getDescription` 傳回 `NULL`。

RETURN_RESULT_FIELDS

如果 `isReturned` 設定爲 `False`，則 `getFields` 傳回 `NULL`。

RETURN_RESULT_CATEGORIES

如果 `isReturned` 設定爲 `False`，則 `getCategories` 傳回 `NULL`。

RETURN_RESULT_TYPE

如果 `isReturned` 設定爲 `False`，則 `getDocumentType` 傳回 `NULL`。

RETURN_RESULT_SOURCE

如果 `isReturned` 設定爲 `False`，則 `getDocumentSource` 傳回 `NULL`。

RETURN_RESULT_LANGUAGE

如果 isReturned 設定為 False，則 getLanguage 傳回 NULL。

RETURN_RESULT_DATE

如果 isReturned 設定為 False，則 getDate 傳回 NULL。

RETURN_RESULT_SCORE

如果 isReturned 設定為 False，則 getScore 傳回 0.0。

語法

```
public void setReturnedAttribute(int attributeType,  
                                boolean isReturned)
```

參數

attributeType

結果屬性類型。

isReturned

如果應傳回指定類型的屬性，即為 True；反之，則為 False。

isAttributeReturned 方法

如果查詢結果物件包含指定的屬性類型，將傳回 True。依預設會傳回所有結果屬性。

有效的 attributeType 值是：

RETURN_RESULT_TITLE

如果 isReturned 設定為 False，則 getTitle 將傳回 NULL（依預設，除了欄位外，會傳回所有結果屬性）。

RETURN_RESULT_DESCRIPTION

如果 isReturned 設定為 False，則 getDescription 將傳回 NULL。

RETURN_RESULT_FIELDS

如果 isReturned 設定為 False，則 getFields 將傳回 NULL。

RETURN_RESULT_CATEGORIES

如果 isReturned 設定為 False，則 getCategories 將傳回 NULL。

RETURN_RESULT_TYPE

如果 isReturned 設定為 False，則 getDocumentType 將傳回 NULL。

RETURN_RESULT_SOURCE

如果 isReturned 設定為 False，則 getDocumentSource 將傳回 NULL。

RETURN_RESULT_LANGUAGE

如果 isReturned 設定為 False，則 getLanguage 將傳回 NULL。

RETURN_RESULT_DATE

如果 isReturned 設定為 False，則 getDate 將傳回 NULL。

RETURN_RESULT_SCORE

如果 isReturned 設定為 False，則 getScore 將傳回 0.0。

語法

```
public boolean isAttributeReturned(int attributeType)
```

參數

attributeType

結果屬性類型

傳回

如果傳回指定的屬性，即為 `True`；反之，則為 `False`。

setQueryID 方法

指定這個查詢的應用程式定義 ID。搜尋引擎可以使用這個 ID 來記載此查詢的執行資訊。

語法

```
public void setQueryID(java.lang.String queryID)
```

參數

queryID

應用程式定義的查詢 ID。

getQueryID 方法

傳回對這個查詢所設定的應用程式定義查詢 ID；如果沒有設定 ID，則傳回 `NULL`。

語法

```
public java.lang.String getQueryID()
```

傳回

傳回應用程式定義的查詢 ID。

setACLConstraints 方法

設定這個查詢的存取控制限制。

請參閱第 8 頁的『查詢語法』，以取得 ACL 限制語法的相關資訊。

語法

```
public void setACLConstraints(java.lang.String aclConstraints)
```

參數

aclConstraints

指定存取控制限制的字串。

getACLConstraints 方法

傳回對這個查詢所設定的存取控制限制字串。如果沒有設定存取控制限制，則傳回 `NULL`。

請參閱第 8 頁的『查詢語法』，以取得 ACL 限制語法的相關資訊。

傳回

傳回對這個查詢所設定的存取控制限制字串。

setQueryLanguage 方法

指定搜尋伺服器所用集合預設語言以外的其他語言。

語法

```
public void setQueryLanguage(String lang) throws SiapiException
```

參數

lang

這個查詢的語言。您可以使用兩個字母的代碼（如：en、ja），也可以使用四個字母的代碼（如：en-US、fr-FR）來指定查詢語言。

getResultCategoriesDetailLevel 方法

傳回查詢結果的種類明細層級。

查詢結果的種類明細層級。可能的回覆值是：

RESULT_CATEGORIES_ALL

傳回每個結果種類及其完整路徑（從根路徑開始）資訊。

RESULT_CATEGORIES_NO_PATH_TO_ROOT

傳回每個結果種類，但沒有完整路徑（從根路徑開始）資訊。

語法

```
public int getResultCategoriesDetailLevel()
```

傳回

setText 方法

設定這個查詢的文字。

請參閱第 8 頁的『查詢語法』，以取得詳細資訊。

語法

```
public void setText(String text) throws SiapiException
```

參數

text

這個查詢的新文字。

NameValuePair 介面

NameValuePair 介面包含文件的名稱及相關值。

每個 NameValuePair 代表一個文件欄位或中間標記。Result 實作會儲存 NameValuePair 物件陣列，來代表特定文件的完整欄位清單。

getName 方法

傳回與這個值相關聯的名稱。

語法

```
public String getName()
```

傳回

與這個值相關聯的名稱。

getValue 方法

傳回值。

語法

```
public String getValue()
```

傳回

傳回值。

getFields 方法

傳回針對這個結果所傳回的所有欄位名稱及值。如果沒有欄位，或使用者要求不傳回欄位，則傳回 NULL。

語法

```
public NameValuePair[] getFields()
```

Result 介面

Result 介面包含單一搜尋結果。

每個 result 都會存放相對應的文件唯一 ID 資訊。視您所要求擷取的屬性而定，結果也可以包含文件標題、說明（若有的話）、分數、文件類型、來源及語言。

每個 result 也可以包含一組顯示成 NameValuePairs 的動態欄位屬性，以及文件目前指定所屬的種類清單。

getCategories 方法

傳回這個結果文件所屬的種類。如果這個結果不屬於任何種類，或如果使用者要求不傳回種類，這個方法就會傳回 NULL。

語法

```
public ResultCategory[] getCategories()
```

getDate 方法

如果沒有日期，或使用者用 Query 物件的 setReturnedAttribute 方法指定不傳回日期，則傳回 NULL。

語法

```
public Date getDate()
```

getDescription 方法

傳回這個結果文件的說明（也稱為摘要）。如果沒有欄位，或使用者用 Query 物件的 setReturnedAttribute 方法指定不傳回欄位，則傳回 NULL

語法

```
public String getDescription()
```

getDocumentID 方法

傳回這個結果文件的唯一名稱。

語法

```
public String getDocumentID()
```

getDocumentSource 方法

傳回這個結果文件的來源。如果沒有欄位，或使用者要求不傳回欄位，則傳回 NULL 或不明。

語法

```
public String getDocumentSource()
```

getDocumentType 方法

傳回這個結果文件的類型。如果沒有欄位，或使用者要求不傳回欄位，則傳回 NULL 或不明。

語法

```
public String getDocumentType()
```

getFields 方法

傳回針對這個結果所傳回的所有欄位名稱及值。如果沒有欄位，或使用者要求不傳回欄位，則傳回 NULL。

語法

```
public NameValuePair[] getFields()
```

getFields 方法

傳回這個結果所傳回名為 `fieldName` 的欄位值。如果沒有可傳回的欄位，則傳回 NULL。

語法

```
public String[] getFields(String fieldName)
```

getLanguage 方法

傳回這個結果文件的語言。如果語言不明，或使用者要求不傳回語言，則傳回 NULL。

語法

```
public String getLanguage()
```

getProperties 方法

傳回所有可搜尋的屬性；如果沒有，則傳回 NULL。

語法

```
public Properties getProperties()
```

getProperty 方法

傳回可搜尋屬性的值。

語法

```
public String getProperty(String name)
```

參數

name

要擷取的可搜尋屬性名稱。

getScore 方法

傳回索引指定給這個結果的分數。如果使用者要求不傳回分數，則傳回 NULL。

語法

```
public double getScore()
```

getTitle 方法

傳回這個結果文件的標題。如果沒有可用的標題，或使用者要求不傳回標題，則傳回 NULL。

語法

```
public String getTitle()
```

isFirstOfASite 方法

如果這個結果是相同網站一連串結果中的第一個，會傳回 True。

語法

```
public boolean isFirstOfASite()
```

ResultCategory 介面

結果文件的種類。

語法

```
public interface ResultSet extends Serializable
```

getConfidence 方法

傳回文件屬於這個種類的信賴水準。信賴水準介於 0 到 100 之間。

語法

```
public double getConfidence()
```

getInfo 方法

傳回這個種類的 ID 及標籤。

語法

```
public CategoryInfo getInfo()
```

getPathFromRoot 方法

傳回所有上層種類的 ID 及標籤，從根種類開始，最後是這個種類的直接母種類。

語法

```
public CategoryInfo[] getPathFromRoot()
```

傳回

上層種類資訊。

getTaxonomyID 方法

傳回這個種類的分類架構 ID。

語法

```
public String getTaxonomyID()
```

傳回

分類架構 ID。

MAX_CONFIDENCE 常數

最大信賴水準（MAX_CONFIDENCE 的值為 100.0）。

語法

```
public static final double MAX_CONFIDENCE
```

MIN_CONFIDENCE 常數

最小信賴水準（MIN_CONFIDENCE 的值為 0.0）。

語法

```
public static final double MIN_CONFIDENCE
```

ResultSet 介面

ResultSet 介面包含已執行查詢的搜尋結果。

除了查詢結果之外，ResultSet 還包含管理員指派的預先定義鏈結、此要求的預估結果總數、可用結果數目、總執行時間，以及拼寫校正建議。透過 Searchable 介面提供的 search 方法會在查詢完成時，將 ResultSet 實例傳給呼叫者。

getAvailableNumberOfResults 方法

傳回這個查詢可以擷取的結果筆數。

這個數目可能與預估的結果筆數不同。例如，預估的結果筆數超出排序的最大值 500，getAvailableNumberOfResults 就會傳回 500。當查詢結果未排序（排序鍵設定為 SORT_KEY_NONE）時，如果這是最後的結果頁面，這個方法會傳回 0；如果還有更多結果，則傳回 1。

語法

```
public int getAvailableNumberOfResults()
```

getEstimatedNumberOfResults 方法

傳回符合這個查詢的預估結果總數，這個值可能與 getAvailableNumberOfResults 傳回的值不同。如果這個數目不明，則傳回 -1。

這個預估值所指的文件數目，就是內含所有必要查詢字詞但完全不含禁止字詞的文件數目。企業搜尋不提供未排序結果的預估結果筆數。因此，排序鍵為 `SORT_KEY_NONE` 時，會傳回 0。

語法

```
public int getEstimatedNumberOfResults()
```

getPredefinedResults 方法

傳回預先定義查詢結果陣列。如果沒有可用的預先定義結果，或您要求不傳回預先定義鏈結，則傳回 `NULL`（請參閱 `Query.setPredefinedResultsEnabled()`）。

這個方法容許搜尋引擎傳回一組強調顯示的特殊查詢結果。

語法

```
public Result[] getPredefinedResults()
```

getProperties 方法

傳回所有可搜尋的屬性；如果沒有，則傳回 `NULL`。

語法

```
public Properties getProperties()
```

getProperty 方法

傳回可搜尋屬性的值。

語法

```
public String getProperty(String name)
```

參數

name

要擷取的可搜尋屬性名稱。

getQueryEvaluationTime 方法

傳回處理查詢及取得這個結果集所需要的時間（毫秒）。

語法

```
public long getQueryEvaluationTime()
```

getResults 方法

傳回取得的結果陣列。

語法

```
public Result[] getResults()
```

getSpellCorrections 方法

傳回指定查詢字串的拼寫校正建議清單；如果沒有可用的建議，則傳回 `NULL`。

如果拼寫校正已啟用，請參閱 `Query` 介面中的方法 - 第 36 頁的『`setSpellCorrectionEnabled` 方法』。

語法

```
public SpellCorrection[] getSpellCorrections()
```

參數

queryText

要檢查的查詢字串。

傳回

傳回指定查詢字串的拼寫校正建議清單。

hasUnconstrainedResults 方法

傳回指示值，指出搜尋期間所套用的限制是否導致這個查詢的結果集沒有任何結果。如果移除查詢限制，就會傳回無條件限制的結果。

相關限制包括：

- ACL
- 種類
- 範圍限制條件
- 結果語言
- 文件類型限制
- 文件來源限制

語法

```
public int hasUnconstrainedResults()
```

傳回

正值表示存在無條件限制結果，0（零）表示沒有無條件限制結果，負值則表示不知道是否有無條件限制結果。

isEvaluationTruncated 方法

在搜尋程序期間，如果評估搜尋來取得這個搜尋結果集的程序至少會截斷一次，就會傳回 True。

語法

```
public boolean isEvaluationTruncated()
```

SpellCorrection 介面

SpellCorrection 介面包含已更正的原始查詢子字串及該子字串的校正建議（依可能性排序）。

getQuerySubstring 方法

傳回有拼寫校正建議的查詢子字串。

語法

```
public String getQuerySubstring()
```

getSuggestions 方法

傳回依可能性排序的拼寫校正建議；如果沒有建議，則傳回 NULL。

語法

```
public String[] getSuggestions()
```

BrowseFactory 介面

BrowseFactory 介面是 SI-API 分類架構瀏覽功能的企業搜尋實作進入點。

getBrowseService 方法

取得瀏覽服務物件。

語法

```
public BrowseService getBrowseService(java.util.Properties config)
    throws SiapiException
```

參數

config

搜尋伺服器的配置資訊

企業搜尋支援 3 種配置屬性（如需取得 SearchService 的相關資訊，請參閱第 5 頁的『SI-API 應用程式結構』）：

- **hostname**（必要） - 企業搜尋搜尋伺服器的主機名稱。
- **port**（選用） - 企業搜尋搜尋伺服器的埠號（預設埠號是 80）。
- **locale**（選用） - 用來傳回轉譯錯誤訊息的語言環境設定（預設值是預設 java 語言環境）。

createApplicationInfo 方法

建立指定應用程式 ID 的應用程式資訊物件。

取得 ApplicationInfo 物件後，您就可以使用 setPassword 或 setToken 來設定應用程式的密碼或安全性符記。

語法

```
public ApplicationInfo createApplicationInfo(String applicationID)
    throws SiapiException
```

參數

ID 應用程式的 ID。

getVersion 方法

傳回特定 SI-API 實作的版本。

每個 SI-API 實作的版本化資料都可進行不同的管理。在此傳回的值可能與 SI-API API 的版本不同。

語法

```
public String getVersion()
```

BrowseService 介面

BrowseService 介面提供集合分類架構瀏覽服務。

getTaxonomyBrowser 方法

傳回介面來瀏覽指定集合的分類架構。

語法

```
public TaxonomyBrowser getTaxonomyBrowser(ApplicationInfo appInfo,  
                                           java.lang.String collectionID,  
                                           java.lang.String taxonomyID)  
    throws SiapiException
```

參數

appInfo

要求端應用程式的鑑別及授權資訊。

collectionID

集合 ID。

taxonomyID

必要分類架構的 ID。

傳回

指定集合的分類架構瀏覽器 SiapiException。

getAvailableTaxonomyBrowsers 方法

傳回指定集合的所有可用 TaxonomyBrowsers 陣列。

語法

```
public TaxonomyBrowser[] getAvailableTaxonomyBrowsers(ApplicationInfo appInfo,  
                                                       java.lang.String collectionID)  
    throws SiapiException
```

參數

appInfo

要求端應用程式的鑑別及授權資訊。

collectionID

集合 ID。

傳回

可用的 TaxonomyBrowsers SiapiException 陣列。

Category 介面

Category 介面含括種類的分類架構資料。

getInfo 方法

傳回這個種類的 ID 及標籤。

語法

```
public CategoryInfo getInfo()
```

getChildren 方法

傳回所有子種類的 ID 及標籤。

語法

```
public CategoryInfo[] getChildren()
```

傳回

子種類 ID。

getPathFromRoot 方法

傳回所有上層種類的 ID 及標籤，從根種類開始，最後是這個種類的直接母種類。

語法

```
public CategoryInfo[] getPathFromRoot()
```

傳回

上層種類資訊。

getRootCategory 方法

傳回樹狀結構的根種類。

語法

```
public Category getRootCategory()
```

傳回

根種類。

getCategory 方法

傳回指定種類 ID 的種類。

語法

```
public Category getCategory(java.lang.String categoryID)
```

參數

categoryID
要傳回的種類 ID。

傳回

指定的種類。

TaxonomyBrowser 介面

TaxonomyBrowser 介面提供分類架構瀏覽及導覽介面。這個介面是用於唯讀、非管理分類架構瀏覽作業。

getRootCategory 方法

傳回樹狀結構的根種類。

語法

```
public Category getRootCategory()
```

傳回

根種類。

getCategory 方法

傳回指定種類 ID 的種類。

語法

```
public Category getCategory(java.lang.String categoryID)
```

參數

categoryID
要傳回の種類 ID。

傳回

指定的種類。

getTaxonomyInfo 方法

傳回分類架構資訊 (ID 及標籤)。

語法

```
public TaxonomyInfo getTaxonomyInfo()
```

SiapiException 類別

通用 SI-API 異常。

搜尋及檢索 API 方法會導致這個異常發生。這是唯一的 SI-API 異常，藉由設定適當的嚴重性及適當的類型，便可用來代表多種錯誤。

每個 SiapiException 都有相關聯的嚴重性、錯誤碼及選擇性的內含異常。每個錯誤碼都有相關聯的參數化訊息，而 SI-API 實作還可以新增引數來解析錯誤碼。

這個異常的實作方式，是將異常的原始原因（即原始最低異常）作為內含異常保留在 SiapiException 中。因此，當列印堆疊追蹤時，就會列印原始異常的呼叫堆疊。這個實作有助於釐清問題的原因。

SEVERITY_ERROR 常數

非嚴重錯誤的嚴重性。

語法

```
public static final int SEVERITY_ERROR
```


SEVERITY_FATAL_ERROR 常數

嚴重錯誤的嚴重性。

語法

```
public static final int SEVERITY_FATAL_ERROR
```

TYPE_UNKNOWN_ERROR 常數

非預期的內部錯誤。

訊息：內部錯誤：%1

訊息參數：

%1：一般訊息

語法

```
public static final int TYPE_UNKNOWN_ERROR
```

TYPE_DOC_EXIST_ERROR 常數

嘗試建立已經存在的文件。

訊息：「文件 %1 已經存在」

訊息參數：

%1：文件 ID

語法

```
public static final int TYPE_DOC_EXIST_ERROR
```

TYPE_DOC_NOT_FOUND_ERROR 常數

嘗試存取或移除不存在的文件。

訊息：「文件 %1 不存在」

訊息參數：

%1：文件 ID

語法

```
public static final int TYPE_DOC_NOT_FOUND_ERROR
```

TYPE_SEARCH_ENGINE_STATE_ERROR 常數

嘗試與搜尋引擎的互動不合法。搜尋引擎狀態並不符合要求的作業。

訊息：「搜尋引擎狀態錯誤：%1」

訊息參數：

%1：一般訊息

語法

```
public static final int TYPE_SEARCH_ENGINE_STATE_ERROR
```

TYPE_ILLEGAL_VALUE_ERROR 常數

嘗試設定不合法的值（例如，將屬性設定為不合法的值）。

訊息：「不合法值錯誤：%1」

訊息參數：

%1：不合法的名稱及值

語法

```
public static final int TYPE_ILLEGAL_VALUE_ERROR
```

TYPE_IO_ERROR 常數

由於 I/O 錯誤，導致與搜尋引擎的互動失敗。例如，因為磁碟已滿而無法建立檔案。

訊息：「IO 錯誤」

訊息參數：

無

語法

```
public static final int TYPE_IO_ERROR
```

TYPE_IMPL_FACTORY_ERROR 常數

嘗試建立 SI-API 實作的 factory 物件失敗。

可能原因：

- 無法找到並載入 factory 實作類別。
- 無法立即使用 factory 實作物件。

訊息：「factory 實作錯誤」

訊息參數：

無

語法

```
public static final int TYPE_IMPL_FACTORY_ERROR
```

TYPE_UNSUPPORTED_OPERATION 常數

指出 SI-API 實作缺少部份功能。

訊息：「未支援的作業：%1」

訊息參數：

%1 未支援作業的名稱

語法

```
public static final int TYPE_UNSUPPORTED_OPERATION
```

TYPE_INDEX_DOES_NOT_EXIST 常數

指定名稱的索引不存在。

訊息：「集合 %1 不存在」

訊息參數：

%1：集合 ID

語法

```
public static final int TYPE_INDEX_DOES_NOT_EXIST
```

TYPE_TOO_MANY_VALUES 常數

多值變數所指定的值多於容許的數目。

訊息：「引數 %1 有太多值」

訊息參數：

%1：引數名稱

語法

```
public static final int TYPE_TOO_MANY_VALUES
```

TYPE_TOO_FEW_VALUES 常數

多值變數所指定的值少於要求的數目。

訊息：「引數 %1 有太少值」

訊息參數：

%1：引數名稱

語法

```
public static final int TYPE_TOO_FEW_VALUES
```

TYPE_ILLEGAL_RESULTS_RANGE 常數

要求的結果範圍超出搜尋引擎所容許的值。

訊息：「指定的結果範圍不合法：%1」

訊息參數：

%1：指定的結果範圍

語法

```
public static final int TYPE_ILLEGAL_RESULTS_RANGE
```

TYPE_INDEX_CORRUPTED 常數

索引已毀損。

訊息：「集合 %1 有一個索引毀損」

訊息參數：

%1：集合 ID

語法

```
public static final int TYPE_INDEX_CORRUPTED
```

TYPE_UNKNOWN_ENCODING 常數

由於字元編碼不明，導致某些資料無法使用。

訊息：「由於編碼 %1 不明，以致資料無法使用」

訊息參數：

%1：編碼名稱

語法

```
public static final int TYPE_UNKNOWN_ENCODING
```

TYPE_QUERY_SYNTAX_ERROR 常數

發生查詢語法錯誤。

訊息：「查詢語法錯誤：%1」

訊息參數：

%1：不合法的查詢字串

語法

```
public static final int TYPE_QUERY_SYNTAX_ERROR
```

getSeverity 方法

傳回這個異常的嚴重性（可能是 `SEVERITY_ERROR` 或 `SEVERITY_FATAL_ERROR`）。

語法

```
public int getSeverity()
```

傳回

傳回這個異常的嚴重性。

getSeverityDescription 方法

傳回這個異常的嚴重性名稱（人員可讀的）。

語法

```
public java.lang.String getSeverityDescription()
```

傳回

傳回這個異常的嚴重性名稱（人員可讀的）。

getType 方法

傳回這個異常的類型。

可能是 `SEVERITY_ERROR` 或 `SEVERITY_FATAL_ERROR`

可能是 `SEVERITY_ERROR` 或 `SEVERITY_FATAL_ERROR`

語法

```
public int getType()
```

傳回

傳回這個異常的類型。

getTypeDescription 方法

傳回這個異常的類型名稱（人員可讀的）。

語法

```
public java.lang.String getTypeDescription()
```

傳回

傳回這個異常的類型名稱（人員可讀的）。

printStackTrace 方法

列印內含異常的堆疊追蹤。否則，如果堆疊追蹤不存在，則列印這個異常的堆疊。

語法

```
public void printStackTrace()
```

置換

置換 `java.lang.throwable` 類別中的 `printStackTrace`。

addArgument 方法

新增額外的名稱值組作為這個異常的引數。

語法

```
public void addArgument(NameValuePair arg)
```

參數

arg

額外的名稱值組。

getArguments 方法

傳回這個異常的引數。

語法

```
public java.util.ArrayList getArguments()
```

傳回

傳回這個異常的引數及引數陣列清單。

getMessage 方法

傳回這個異常的錯誤訊息字串。

語法

```
public java.lang.String getMessage()
```

傳回

傳回這個異常的錯誤訊息字串。

置換

置換 `java.lang.throwable` 類別中的 `getMessage`。

getLocalizedMessage 方法

傳回這個異常的當地語言訊息。

語法

```
public java.lang.String getLocalizedMessage()
```

傳回

傳回這個異常的當地語言訊息。

語法

```
public java.lang.String getLocalizedMessage()
```

置換

java.lang.throwable 類別中的 `getLocalizedMessage`

TaxonomyInfo 介面

TaxonomyInfo 介面提供分類架構的 ID 及標籤。

getID 方法

傳回分類架構 ID。

語法

```
public java.lang.String getID()
```

getLabel 方法

傳回分類架構標籤。

語法

```
public java.lang.String getLabel()
```

getTaxonomyInfo 方法

傳回分類架構資訊（ID 及標籤）。

語法

```
public TaxonomyInfo getTaxonomyInfo()
```

範例 SI-API 應用程式

企業搜尋 SI-API 包括數個範例應用程式。

範例應用程式示範：

- 將搜尋查詢提給搜尋伺服器至少需要的應用程式
- 基本搜尋作業
- 程式設計所需的一般搜尋作業
- 基本分類架構瀏覽及導覽作業

至少必要的應用程式

SearchExample 類別提供一個簡單的範例，您至少需要這樣的應用程式，才能將搜尋查詢提給搜尋伺服器。

下列範例示範如何：

- 存取服務
- 指定集合
- 指定應用程式
- 提出查詢
- 處理傳回的結果

SearchExample 類別

SearchExample 類別已在 `esapi.jar` 檔案中編譯。您可以在命令行中執行下列命令，來執行這個類別。

UNIX

```
java -classpath <installroot>/lib/esapi.jar:<installroot>/lib/siapi.jar  
SearchExample
```

Windows

```
javac -classpath <installroot>\lib\esapi.jar;<installroot>\lib\siapi.jar  
SearchExample
```

如果未指定主機，預設主機會假設是本端主機。如果未指定埠，則預設埠會假設是 80。

取得 factory

任何根據 SI-API 所撰寫的程式第一行，都應建立 Factory 類別的實例。企業搜尋中的 SearchFactory 類別實作是：`com.ibm.es.api.search.RemoteSearchFactory`。

取得企業搜尋特定 SI-API 實作的 factory。

```
SearchFactory factory =  
    SiapiSearchImpl.createSearchFactory  
    ("com.ibm.es.api.search.RemoteSearchFactory");
```

建立有效的應用程式 ID

搜尋伺服器使用應用程式 ID 來授權存取集合。

```
ApplicationInfo appinfo = factory.createApplicationInfo("app1");  
appinfo.setPassword("password");
```

建立新的 Properties 物件

```
Properties config = new Properties();
```

設定主機名稱

設定 SI-API 搜尋伺服器的主機名稱。在此例中即為企業搜尋搜尋伺服器。主機名稱應是安裝企業搜尋 WebSphere 時所指定的主機名稱。

```
config.setProperty("hostname", hostname);
```

設定埠號

設定 SI-API 搜尋伺服器的埠號。預設值是埠 80（Web 伺服器埠）。

```
config.setProperty("port", portNumber);
```

設定語言環境

設定用戶端的語言環境。基礎 SIAPI 實作使用此值，將錯誤訊息轉譯成適當的語言，然後傳回用戶端。

```
config.setProperty("locale", "en_US");
```

取得 SearchService 實作

```
SearchService searchService =  
    factory.getSearchService(config);
```

指定的集合 ID 取得 Searchable 物件

```
Searchable searchable =  
    searchService.getSearchable(appinfo, collectionId);  
if (searchable == null) {  
    System.out.println("Failed to get a searchable  
    for collection: " + collectionId);  
    return;  
}
```

使用指定查詢字串建立新的 Query 物件

```
Query q = factory.createQuery(queryString);
```

設定結果範圍

在這個查詢上設定您要存取的结果範圍。

```
q.setRequestedResultRange(0, 10);
```

執行搜尋

呼叫 Searchable 的搜尋方法來執行搜尋。接著會傳回一個 SIAPI ResultSet 物件。

```
ResultSet rset = searchable.search(q);  
System.out.println("returned from search call");  
if(rset != null) {
```

從 ResultSet 取得結果

```
Result r[] = rset.getResults();  
if(r != null) {
```

列印結果

列出結果清單並列印文件 ID 及文件標題。

```
for (int k = 0; k < r.length; k++) {  
    System.out.println  
        ("Result " + k + ": " + r[k].getDocumentID() + " - " + r[k].getTitle());  
    }  
} else {  
    System.out.println("result set was null");  
}  
}
```

僅供測試之用

```
public static void main(String[] args) throws Exception {  
    SearchExample sc = new SearchExample();  
    if (args.length < 2) {  
        System.out.println
```



```

("Usage: SearchExample<queryString><collection ID>optionally:<hostname><port>");
System.out.println("Example Usage: SearchExample lotus coll");
System.out.println("\tdefault host and port is localhost:80");
System.out.println("Example Usage: SearchExample lotus coll localhost 80");
    return;
    }
    sc.queryString = args[0];
    sc.collectionId = args[1];
    if (args.length > 2) {
        sc.hostname = args[2];
    } else {
        sc.hostname = "localhost";
    }
    if (args.length > 3) {
        sc.portNumber = args[3];
    } else {
        sc.portNumber = "80";
    }
    sc.execute();
}
}

```

瀏覽及導覽應用程式

這個範例程式碼是基本分類架構瀏覽及導覽作業的 SI-API 範例。

這個範例示範如何：

- 取得瀏覽 factory
- 取得瀏覽服務
- 取得瀏覽器參考資訊
- 取得並顯示根種類
- 取得根種類的第一個子種類
- 顯示子種類及其從根路徑開始的路徑

瀏覽及導覽範例

```

*/
import java.util.Locale;
import java.util.Properties;

import com.ibm.siapi.browse.BrowseFactory;
import com.ibm.siapi.browse.BrowseService;
import com.ibm.siapi.browse.Category;
import com.ibm.siapi.browse.SiapiBrowseImpl;
import com.ibm.siapi.browse.TaxonomyBrowser;
import com.ibm.siapi.common.ApplicationInfo;
import com.ibm.siapi.common.CategoryInfo;
import com.ibm.siapi.common.TaxonomyInfo;

/*
 * The BrowseExample class gives an example of accessing a collection's
 * taxonomy tree and printing out some of the basic navigation properties.
 *
 */
public class BrowseExample {
    private String hostname    = "localhost";
    private String portNumber  = "80";
    private String collectionId;
    private String taxonomyId;
    private String applicationName;
    private String applicationPassword;

```

```

public void execute() throws Exception {
    // obtain the OmniFind specific SI-API Browse factory implementation
BrowseFactory factory =
    SiapiBrowseImpl.createBrowseFactory("com.ibm.es.api.browse.RemoteBrowseFactory");

    // create a valid Application ID that will be used
    // by the Search Node to authorize this
    // access to the collection
    ApplicationInfo appinfo = factory.createApplicationInfo(applicationName);
    appinfo.setPassword(applicationPassword);

    // create a new Properties object.
    Properties config = new Properties();
    // set the hostname of the OmniFind Search Node. The hostname
    // should be the hostname that is assigned to the
    // Search Node's WebSphere installation
    config.setProperty("hostname", hostname);
    // set the port number - the
    // default value is port 80 (the web server port).
    config.setProperty("port", portNumber);
    // set the locale of the "client". This value is
    // used by the underlying SI-API implementation to
    // return translated error messages to the client
    // for the appropriate language.
    // NOTE: this should be the 2 letter ISO-639
    // language code followed by an underscore
    // followed by the ISO-639 2 letter country code
    config.setProperty("locale", "en_US");

    // obtain the Browse service implementation
BrowseService browseService = factory.getBrowseService(config);

    // get a TaxonomyBrowser for the specified taxonomy id and collection id
    TaxonomyBrowser browser = browseService.getTaxonomyBrowser(appinfo,
collectionId, taxonomyId);
    if(browser == null) {
        System.out.println("Failed to get a taxonomy for taxonomy id: "
+ taxonomyId +
            " from collection: " + collectionId);
        return;
    }

    //Display the Taxonomy Info
    TaxonomyInfo taxonomyInfo = browser.getTaxonomyInfo();
    System.out.println("Taxonomy label: " + taxonomyInfo.getLabel());

    // get the root category
    Category rootCategory = browser.getRootCategory();

    // display the root category
    System.out.println("Root category label: " + rootCategory.getInfo()
.getLabel());
    System.out.println("child categories:");
    CategoryInfo[] childrenInfo = rootCategory.getChildren();
    for (int i = 0; i < childrenInfo.length; i++) {
        System.out.println("\t" + childrenInfo[i].getLabel());
    }

    // Now get the root's first child category
    Category childCategory = browser.getCategory(rootCategory.getChildren()[0]
.getID());

    // Display the child category and its path from root
    System.out.println("Root's first child's label: " + childCategory.getInfo()
.getLabel());
    System.out.println("It's path from root is : ");
    CategoryInfo[] pathFromRoot = childCategory.getPathFromRoot();

```

```

        for (int i = 0; i < pathFromRoot.length; i++) {
            System.out.println("-->" + pathFromRoot[i].getLabel());
        }
    }

    public static void main(String[] args) throws Exception {
        BrowseExample sc = new BrowseExample();
        if (args.length < 5) {
            System.out.println("Usage: BrowseExample <taxonomy id> <collection id>
<application name>");
            System.out.println("Example Usage: BrowseExample tax1 col1 Default password
localhost 80")
            return;
        }
        sc.taxonomyId = args[0];
        sc.collectionId = args[1];
        sc.applicationName = args[2];
        sc.applicationPassword = args[3];
        sc.hostname = args[4];
        sc.portNumber = args[5];
        sc.execute();
    }
}

```

擷取所有搜尋結果

下列範例示範如何設定查詢來傳回未排序的結果及重複查詢結果。企業搜尋容許查詢取得最多 500 筆排序結果，不過，您可以取得所有未排序的結果。

取得 SearchFactory 及 Searchable

取得 SearchFactory 及 Searchable 物件，如第 56 頁的『至少必要的應用程式』範例所示。

```

SearchFactory factory;
Searchable searchable;

... // obtain a SearchFactory and Searchable object

```

建立新的 Query 物件

```
Query q = factory.createQuery("big apple");
```

設定查詢傳回未排序的結果

```
q.setSortKey(Query.SORT_KEY_NONE);
```

執行搜尋

重複執行查詢，一次取得一頁結果。企業搜尋所容許的結果頁面大小上限是 100 筆。

註：收到結果頁面時，您必須對 `getAvailableNumberOfResults` 及 `getEstimatedNumberOfResults` 做出與排序查詢結果不同的解釋：

- `getEstimatedNumberOfResults` 永遠傳回 0，因為企業搜尋不提供未排序結果的預估結果筆數。
- `getAvailableNumberOfResults` 會傳回兩個值之一：如果這是最後的結果頁面，傳回 0；如果還有更多結果，則傳回 1。
- 您可以使用 `getResults` 傳回的陣列長度，瞭解這個結果頁面實際有多少筆結果。

```
int fromResult = 0;
int pageSize = 100;
boolean moreResults = true;

// loop over query results, pageSize results at a time
while (moreResults) {

    // set the result range for the next page of results
    q.setRequestedResultRange(fromResult, pageSize);

    // execute the search
    ResultSet resultPage = s.search(q);

    // loop over the results from the ResultSet
    Result[] results = resultPage.getResults();
    for (int i=0;i<results.length;i++) {
    ... // process result
    }

    // check if there are more available results
    moreResults = (resultPage.getAvailableNumberOfResults() == 1);

    // modify the range for getting the next page of results
    fromResult += pageSize;
}
```

第 4 章 資料接聽器 API

資料接聽器是一個企業搜尋元件，接收來自已知埠的資料，並將資料傳送給集合。經由呼叫資料接聽器 API，使用者可將頁面新增到集合、將 URI 從集合中移除，或指示集合的 Web 搜索器造訪或重新造訪 URL。

企業搜尋概觀

企業搜尋系統可以收集各種來源的資料、為收集到的資料建立索引，以及提供搜尋功能。資料來源包括 Web、新聞伺服器、資料庫表格、Lotus Notes® 資料庫、內容管理系統及檔案系統。

圖 1 顯示企業搜尋系統搜尋行為的概觀。使用者對搜尋伺服器提出查詢，而索引伺服器則定期重新整理搜尋伺服器上的資料。

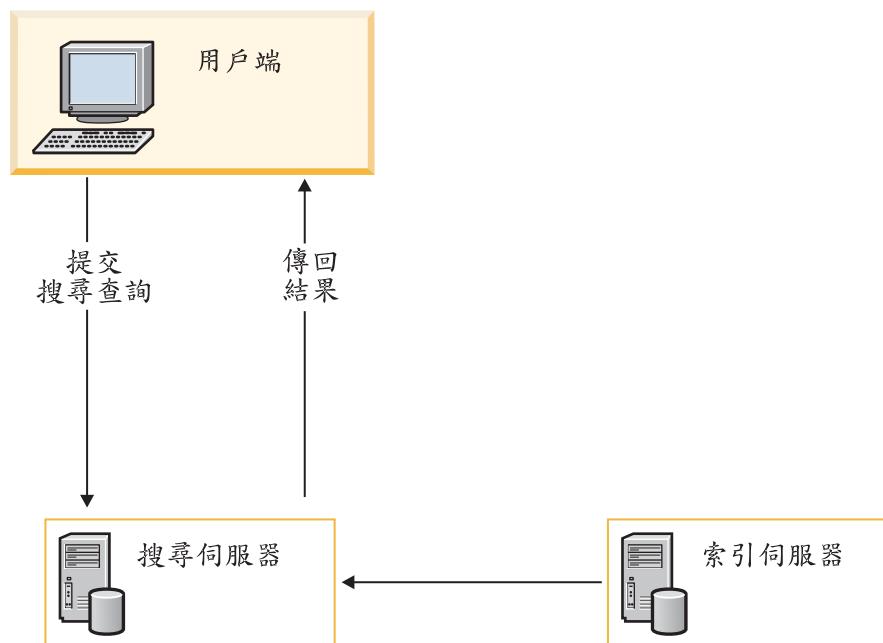


圖 1. 企業搜尋行為概觀

資料接聽器 API 概觀

資料接聽器會連繫企業搜尋系統中的下列元件：

用戶端機器

提出搜尋查詢。

搜尋伺服器

接受查詢並將結果提供給用戶端。

索引伺服器

儲存來自搜索器的剖析資料。

搜索器 從資料來源擷取文件。

資料接聽器 API 可讓您向企業搜尋系統提出資料要求。

用戶端連線到資料接聽器元件，以便將資料推播到適當的伺服器。用戶端連線到資料接聽器元件時，資料接聽器元件會驗證用戶端 ID 及密碼，並驗證用戶端是否得到授權，以便將資料推播到指定的集合。

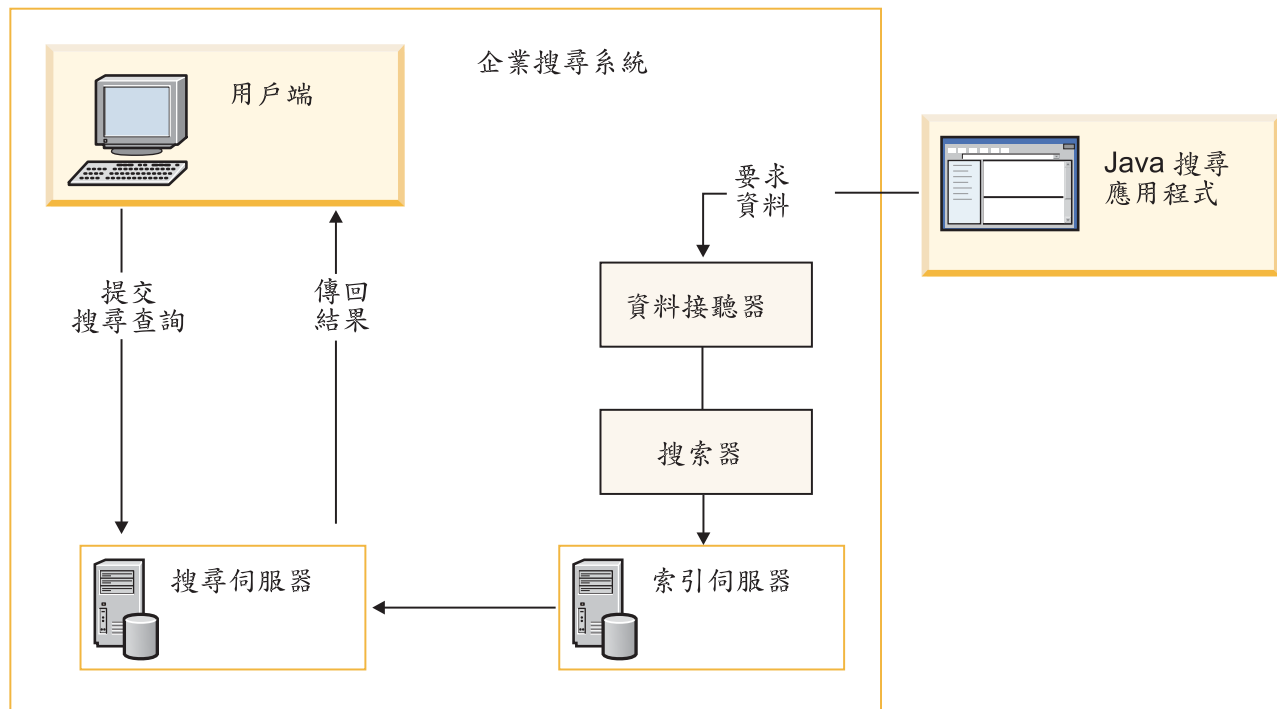


圖 2. 資料接聽器 API 搭配企業搜尋系統的運作方式

資料接聽器 API 屬性

若要將資料推播到資料接聽器元件，您必須提供下列資訊：

表 2. 資料接聽器 API 屬性

屬性	定義
Hostname	資料接聽器伺服器的主機名稱。
Port	伺服器接聽的埠號。
ID	用戶端 ID。
PW	用戶端密碼。
Col	目標集合的 ID。
URI	資料的 URI。
Metadata	資料的 Metadata 物件，屬於 DataSourceMetada 類型。
Content	資料的主體。

資料控制

資料接聽器 API 可以向集合伺服器及搜索器提出 URI 及 URL 要求，來新增及移除內容。

統一資源識別碼 (URI)

代表 Web 上任何資源（如 HTML 文件、影像、視訊短片或程式）的編碼位址。URI 超類別包括 URL。

統一資源定位器 (URL)

即網址，用來命名及尋找 Web 上的特定項目。

使用資料接聽器 API 移除資料

使用資料接聽器 API 移除搜尋集合資料的方式有兩種：

- 移除搜尋伺服器及索引中的 URI
- 移除索引中的 URI 型樣

移除搜尋伺服器及索引中的 URI

您可以使用資料接聽器 API 來移除特定的 URI。要求傳送到資料接聽器後，就會將 URI 從索引伺服器及搜尋伺服器移除，如下圖所示。

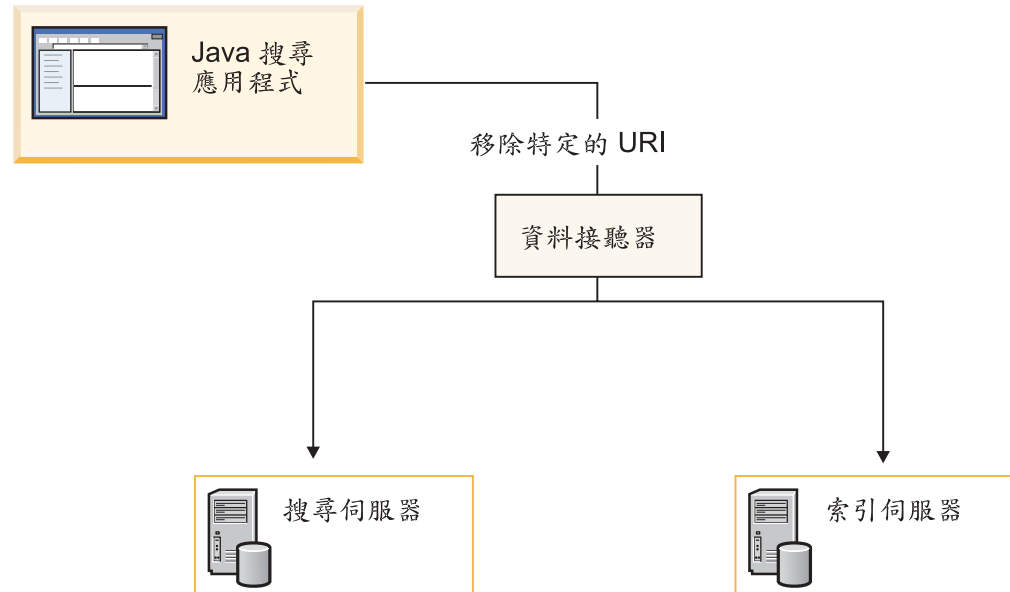


圖 3. 移除特定的 URI

使用 `removeURI` 方法可刪除指定集合的資料。

移除索引中的 URI 型樣

您可以使用資料接聽器 API 來移除 URI 型樣。例如，如果您提出移除 URI 型樣的要求，並指定 URI 為 `www.ibm.com/*.html`，索引伺服器就會移除下列 URL：

- `www.ibm.com/home.html`
- `www.ibm.com/family.html`
- `www.ibm.com/pics.html`

註：請謹慎使用移除 URI 型樣的要求。已移除的內容將無法回復。該內容必須重新新增到索引中。

移除要求傳送到資料接聽器後，就會移除索引伺服器的 URI 型樣，如上圖所示。

使用 `removeURI` 方法可刪除指定集合的資料。在索引伺服器重新整理搜尋伺服器上的資料之前，已移除的 URI 還是會出現在搜尋結果中，一起傳給使用者。

使用資料接聽器 API 新增資料

使用資料接聽器 API 將資料推播到搜尋集合的方式有兩種：

- 新增 URI 及內容
- 造訪或重新造訪 URL

新增 URI 及內容

您可以在 `pushData` 要求中加入內容參數，便能使用資料接聽器 API 來新增 URL 及其內容。如此一來，指定的資料就會傳送到資料接聽器及索引伺服器，如圖 4 所示。

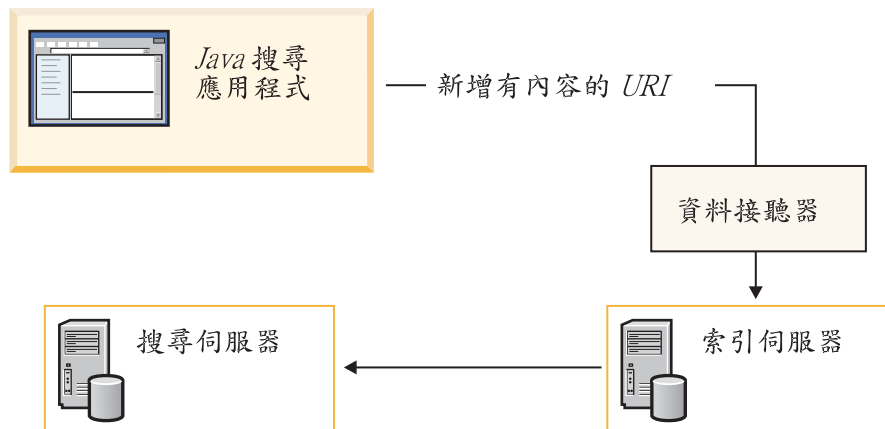


圖 4. 新增 URI 及內容

索引伺服器重新整理搜尋伺服器上的資料後，使用者便可存取新增的 URI 及其內容。

造訪或重新造訪 URL

您可以透過 `revisitURLs` 方法，使用資料接聽器 API 來新增特定的 URL 並擷取其資料。此方法僅適用於 Web 內容。所有其他資料來源則必須使用 `pushData` 呼叫。

造訪 要求搜索器從新 URL 擷取資料

重新造訪

要求搜索器重新整理目前在索引中的 URL 資料

如果您新增 URL 但沒有相關聯的資料，資料接聽器會將要求傳送到搜索器，搜索器會擷取內容並將內容傳送到索引伺服器。

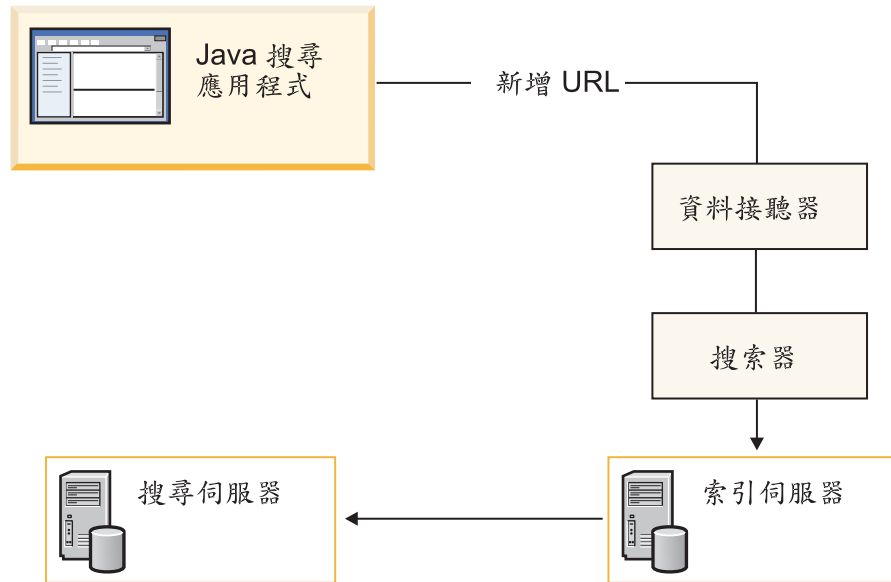


圖 5. 新增 URL

將資料推播到索引器

資料接聽器 API 會將文件推播到索引器。

文件包括三個元件：

- URI
- 內容
- 中間資料

指定中間資料

針對所有新增資料要求，您必須在發出推播資料呼叫前，指定中間資料方法。中間資料包含文件資訊，也可包括作者、修改日期或建立日期等資訊。

使用 `createDataSourceMetadata` 方法可建立中間資料。每個中間資料都有兩個區段：

- `CommonMetadata`
- `DataSourceSpecificMetadata`

建立中間資料

使用 `createDataSourceMetadata` 方法可建立中間資料。

`CommonMetadata` 區段包括所有資料來源共用的所有欄位。這些欄位包括：

Datasource

指定資料來源。這個欄位是可以搜尋的，而且會在搜尋結果中傳回。

SecurityACL

指定使用者所需的認證符記，以便在搜尋結果中擷取這個文件。若有多個符記，請以逗點區隔。如果這個欄位是空白或不存在，則預設值為 `PUBLIC`，表示所有人都可以使用。

Date 代表 EPOCH 之後經過秒數的數字字串（負數代表 EPOCH 之前的資料）。日期採用 GMT 時間，而非區域時間。日期可用來計算文件的靜態分數。這個欄位是用來支援日期排序及範圍搜尋。

StaticScoreRef

可用來計算文件靜態分數的數字字串。這個欄位將保留供以後使用。

HasSeparateContent

指示實際內容是否為空白的字串。如果內容不是空白的，屬性將指出內容類型、字集及內容的語言。

下列範例顯示共用的中間資料欄位。

```
<Metadata Language="en">
<CommonMetadata
  Datasource = "DB2",
  CrawlerId="MyCrawlerId",
  DatasourceName="My Display Name For My Data Source",
  Date=GMT,
  StaticScoreRef=8>
<SecurityACLs>Token1,Token2</SecurityACLs>

  <HasSeparateContent ContentType="application/pdf",
    Charset="iso-933",
    Language="gb">
    YES
  </HasSeparateContent>
</CommonMetadata>
</Metadata>
```

DataSourceSpecificMetadata 區段包括資料來源的特定欄位。每個欄位可能有下列屬性：

FieldName

指示欄位名稱，以便欄位搜尋來搜尋這個欄位。

Searchable

指示這個欄位是否可搜尋。預設值為是。

FieldSearchable

指示這個欄位是否需要欄位搜尋。預設值為是。

Metadata

指示這個欄位的內容是否必須包括在搜尋結果的 extraFields 資料中。預設值為是。

ParametricSearch

指示這個欄位是否需要參數搜尋。預設值為否。如果為是，則這個欄位的資料格式應該可以由剖析器轉換成數值。

ResolveConflict

指定在多個欄位具有相同欄位名稱的情況下，要如何解決衝突。例如，某份文件可能有兩個作者欄位，一個來自中間資料，另一個來自內容。可能的選項包括：MetadataPreferred、ContentPreferred 及 Coexist（連接）。

下列範例顯示 DataSourceSpecific 中間資料欄位。

```
<Metadata Language="en">
<DataSourceSpecificMetadata>
  <Field FieldName="DatabaseName",
    Searchable="YES",
```

```

        FieldSearchable="YES",
        Metadata="YES",
        ResolveConflict="MetadataPreferred">
myDatabase
</Field>
<Field FieldName="TableName",
    Searchable="YES",
    FieldSearchable="YES",
    Metadata="YES",
    ResolveConflict="MetadataPreferred">
myTable
</Field>
<Field FieldName="MYID",
    Searchable="YES",
    FieldSearchable="YES",
    Metadata="YES",
    ResolveConflict="MetadataPreferred">
10
</Field>
</DataourceSpecificMetadata>
</Metadata>

```

建立更多的中間資料

若要將更多的中間資料欄位新增到提供的欄位，請使用 `addMetaField` 方法。使用這個方法可將元素新增到現存的中間資料物件。

```

public static void addMetaField(DataSourceMetadata metadata,
                               String fieldName,
                               String fieldValue,
                               boolean searchable,
                               boolean partOfResult,
                               boolean fieldSearchable,
                               boolean parametricSearchable)

```

資料接聽器 API 呼叫及方法

資料接聽器使用下列 API。

DLDataPusher 類別

將資料推播到資料接聽器，來將 URI 從集合中移除，並重新造訪 URL。

若要呼叫這些 API，您必須指定資料接聽器的主機名稱及埠，並提供搜索器 ID 及密碼來進行鑑別。

pushData 方法

`pushData` 方法是一種將資料推播到資料接聽器的 API 函數。

語法

```

public static DLResponse pushData(String hostname,
                                  int port,
                                  String id,
                                  String pw,
                                  String uri,
                                  String col,
                                  String metadata,
                                  byte[] content)

```

參數

hostname

資料接聽器伺服器的主機名稱。

port

資料接聽器的埠號。

ID 搜索器 ID。

pw 搜索器密碼。

URI

資料的 URI。

col 目標集合的 ID。

metadata

中間資料的 XML 字串版本。

content

要推播到資料接聽器的內容。

removeURIs 方法

removeURIs 方法是一種將 URI 從指定集合中移除的 API 函數。

語法

```
public static DLResponse removeURIs(String hostname,  
                                     int port,  
                                     String id,  
                                     String pw,  
                                     String uris,  
                                     String col,
```

參數

hostname

資料接聽器伺服器的主機名稱。

port

資料接聽器的埠號。

ID 搜索器 ID。

pw 搜索器密碼。

URI

資料的 URI。

col 目標集合的 ID。

revisitURLs 方法

指示指定集合的 Web 搜索器來新增或重新造訪 URI。

語法

```
public static DLResponse removeURIs(String hostname,  
                                     int port,  
                                     String id,
```

```
String pw,  
String uris,  
String col)
```

參數

hostname

資料接聽器伺服器的主機名稱。

port

資料接聽器的埠號。

ID 搜索器 ID。

pw 搜索器密碼。

URI

URI 或 URI 型樣，以資料的空格來分隔。

col 目標集合的 ID。

addMetaField 方法

addMetaField 方法將元素新增到 metaData 物件。

語法

```
public static void addMetaField(DataSourceMetadata metadata,  
                                String fieldName,  
                                String fieldValue,  
                                boolean searchable,  
                                boolean partOfResult,  
                                boolean fieldSearchable,  
                                boolean parametricSearchable)
```

createDataSourceMetadata 方法

createDataSourceMetadata 方法可建立 metaData 物件。

語法

```
public static DataSourceMetadata createDataSourceMetadata(String ds,  
                                                         String cid,  
                                                         String dsName,  
                                                         int score,  
                                                         Date dt,  
                                                         String language,  
                                                         String securityACLs,  
                                                         String contentType,  
                                                         String charSet,  
                                                         byte[] content)
```

範例資料接聽器 API 應用程式

DLSampleClient 類別提供了一些簡單的範例，示範如何使用資料接聽器 API 來推播資料。

範例應用程式示範：

- 基本資料接聽器作業
 - 識別及鑑別
 - 推播資料

- 移除資料
- 進階資料接聽器作業
 - 準備中間資料
 - 呼叫推播方法
 - 檢查結果
 - 重複資料推播

基本資料接聽器 API 應用程式

DLSampleClient 類別提供基本資料接聽器 API 作業的範例。

基本範例顯示如何新增或重新造訪某個集合的 URL，以及將 URL 從另一個集合移除。下列範例示範如何：

- 進行識別及鑑別作業
- 推播資料
- 移除資料

資料接聽器範例

```
public class DLSampleClient {
```

進行識別及鑑別作業

若要將資料推播到資料接聽器，您必須提供伺服器的主機名稱及埠。您也需要提供搜索器 ID 及密碼來進行鑑別。

```
static void dataPushExample_1(String hostname, int port, String crawlerID,
    String passwd) {
```

建立 Pusher 物件

```
DLDataPusher pusher = new DLDataPusher(hostname, port, crawlerID, passwd);
```

指定要新增的 URI

指定您想要搜索器新增或重新造訪的特定 URI 或 URI 型樣。

```
StringBuffer sb = new StringBuffer();
sb.append("url1");
sb.append("\n");
sb.append("url*pattern1");
sb.append("\n");

sb.append("url2");
sb.append("\n");
sb.append("url*pattern2");

String collectionID = "collection2";
String urls = sb.toString();
```

導向資料

指定要重新造訪的 URL 目標集合。

```
pusher.revisitURLs(urls, collectionID);
```

移除資料

指定要移除 URI 的目標集合。

```
collectionID = "collection3";  
    pusher.removeURIs(urls, collectionID);  
}
```

DB2 Information Integrator 文件

本主題提供 DB2 Information Integrator 相關文件的資訊。

下列主題中的表格將列出每本 PDF 書籍的正式文件標題、書號及位置。若要訂購印刷版書籍，您必須知道正式書名或文件書號。下列主題也提供 DB2 Information Integrator 版本注意事項及安裝需求的標題、檔名及位置。

DB2 Universal Database on z/OS 事件發佈功能的相關文件

DB2 Universal Database on z/OS 事件發佈功能的相關文件

目的

DB2 Universal Database on z/OS 事件發佈功能的相關文件。

表 3. DB2 Universal Database on z/OS 事件發佈功能的相關 DB2 Information Integrator 文件

名稱	書號	位置
<i>ASNCLP Program Reference for Replication and Event Publishing</i>	N/A	DB2 Information Integrator 支援網站
<i>Introduction to Replication and Event Publishing</i>	GC18-7567	<ul style="list-style-type: none">• DB2 PDF 文件光碟• DB2 Information Integrator 支援網站
<i>Replication and Event Publishing Guide and Reference</i>	SC18-7568	<ul style="list-style-type: none">• DB2 PDF 文件光碟• DB2 Information Integrator 支援網站
<i>Tuning for Replication and Event Publishing Performance</i>	N/A	DB2 Information Integrator 支援網站
<i>Release Notes for IBM DB2 Information Integrator Standard Edition, Advanced Edition, and Replication for z/OS</i>	N/A	<ul style="list-style-type: none">• 在 DB2 資訊中心，產品概觀 > 資訊整合 > DB2 Information Integrator 概觀 > 問題、可行方案及文件更新• DB2 Information Integrator 安裝啓動程式• DB2 Information Integrator 支援網站• <i>DB2 Information Integrator</i> 產品光碟

IMS 及 VSAM on z/OS 事件發佈功能的相關文件

IMS 及 VSAM on z/OS 事件發佈功能的相關文件

目的

IMS 及 VSAM on z/OS 事件發佈功能的相關文件。

表 4. IMS 及 VSAM on z/OS 事件發佈功能的相關 DB2 Information Integrator 文件

名稱	書號	位置
<i>Client Guide for Classic Federation and Event Publisher for z/OS</i>	SC18-9160	DB2 Information Integrator 支援網站
<i>Data Mapper Guide for Classic Federation and Event Publisher for z/OS</i>	SC18-9163	DB2 Information Integrator 支援網站
<i>Getting Started with Event Publisher for z/OS</i>	GC18-9186	DB2 Information Integrator 支援網站
<i>Installation Guide for Classic Federation and Event Publisher for z/OS</i>	GC18-9301	DB2 Information Integrator 支援網站
<i>Operations Guide for Event Publisher for z/OS</i>	SC18-9157	DB2 Information Integrator 支援網站
<i>Planning Guide for Event Publisher for z/OS</i>	SC18-9158	DB2 Information Integrator 支援網站
<i>Reference for Classic Federation and Event Publisher for z/OS</i>	SC18-9156	DB2 Information Integrator 支援網站
<i>System Messages for Classic Federation and Event Publisher for z/OS</i>	SC18-9162	DB2 Information Integrator 支援網站
<i>Release Notes for IBM DB2 Information Integrator Event Publisher for IMS for z/OS</i>	N/A	DB2 Information Integrator 支援網站
<i>Release Notes for IBM DB2 Information Integrator Event Publisher for VSAM for z/OS</i>	N/A	DB2 Information Integrator 支援網站

Linux、UNIX 及 Windows 事件發佈與抄寫功能的相關文件

Linux、UNIX 及 Windows 事件發佈與抄寫功能的相關文件

目的

Linux、UNIX 及 Windows 事件發佈與抄寫功能的相關文件。

表 5. Linux、UNIX 及 Windows 事件發佈與抄寫功能的相關 DB2 Information Integrator 文件

名稱	書號	位置
<i>ASNCLP Program Reference for Replication and Event Publishing</i>	N/A	DB2 Information Integrator 支援網站
<i>Installation Guide for Linux, UNIX, and Windows</i>	GC18-7036	<ul style="list-style-type: none">• DB2 PDF 文件光碟• DB2 Information Integrator 支援網站
<i>Introduction to Replication and Event Publishing</i>	GC18-7567	<ul style="list-style-type: none">• DB2 PDF 文件光碟• DB2 Information Integrator 支援網站
<i>Migrating to SQL Replication</i>	N/A	DB2 Information Integrator 支援網站

表 5. Linux、UNIX 及 Windows 事件發佈與抄寫功能的相關 DB2 Information Integrator 文件 (繼續)

名稱	書號	位置
<i>Replication and Event Publishing Guide and Reference</i>	SC18-7568	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>SQL Replication Guide and Reference</i>	SC27-1121	DB2 Information Integrator 支援網站
<i>Tuning for Replication and Event Publishing Performance</i>	N/A	DB2 Information Integrator 支援網站
<i>Tuning for SQL Replication Performance</i>	N/A	DB2 Information Integrator 支援網站
<i>Release Notes for IBM DB2 Information Integrator Standard Edition, Advanced Edition, and Replication for z/OS</i>	N/A	<ul style="list-style-type: none"> • 在 DB2 資訊中心，產品概觀 > 資訊整合 > DB2 Information Integrator 概觀 > 問題、可行方案及文件更新 • DB2 Information Integrator 安裝啟動程式 • DB2 Information Integrator 支援網站 • <i>DB2 Information Integrator</i> 產品光碟

Linux、UNIX 及 Windows 聯合功能的相關文件

Linux、UNIX 及 Windows 聯合功能的相關文件

目的

Linux、UNIX 及 Windows 聯合功能的相關文件。

表 6. Linux、UNIX 及 Windows 聯合功能的相關 DB2 Information Integrator 文件

名稱	書號	位置
<i>Application Developer's Guide</i>	SC18-7359	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>C++ API Reference for Developing Wrappers</i>	SC18-9172	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>Data Source Configuration Guide</i>	N/A	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>Federated Systems Guide</i>	SC18-7364	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站

表 6. Linux、UNIX 及 Windows 聯合功能的相關 DB2 Information Integrator 文件 (繼續)

名稱	書號	位置
<i>Guide to Configuring the Content Connector for VeniceBridge</i>	N/A	DB2 Information Integrator 支援網站
<i>Installation Guide for Linux, UNIX, and Windows</i>	GC18-7036	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>Java API Reference for Developing Wrappers</i>	SC18-9173	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>Migration Guide</i>	SC18-7360	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>Wrapper Developer's Guide</i>	SC18-9174	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>Release Notes for IBM DB2 Information Integrator Standard Edition, Advanced Edition, and Replication for z/OS</i>	N/A	<ul style="list-style-type: none"> • 在 DB2 資訊中心，產品概觀 > 資訊整合 > DB2 Information Integrator 概觀 > 問題、可行方案及文件更新 • DB2 Information Integrator 安裝啟動程式 • DB2 Information Integrator 支援網站 • <i>DB2 Information Integrator</i> 產品光碟

z/OS 聯合功能相關文件

z/OS 聯合功能相關文件

目的

z/OS 聯合功能相關文件。

表 7. z/OS 聯合功能的相關 DB2 Information Integrator 文件

名稱	書號	位置
<i>Client Guide for Classic Federation and Event Publisher for z/OS</i>	SC18-9160	DB2 Information Integrator 支援網站
<i>Data Mapper Guide for Classic Federation and Event Publisher for z/OS</i>	SC18-9163	DB2 Information Integrator 支援網站
<i>Getting Started with Classic Federation for z/OS</i>	GC18-9155	DB2 Information Integrator 支援網站
<i>Installation Guide for Classic Federation and Event Publisher for z/OS</i>	GC18-9301	DB2 Information Integrator 支援網站

表 7. z/OS 聯合功能的相關 DB2 Information Integrator 文件 (繼續)

名稱	書號	位置
<i>Reference for Classic Federation and Event Publisher for z/OS</i>	SC18-9156	DB2 Information Integrator 支援網站
<i>System Messages for Classic Federation and Event Publisher for z/OS</i>	SC18-9162	DB2 Information Integrator 支援網站
<i>Transaction Services Guide for Classic Federation for z/OS</i>	SC18-9161	DB2 Information Integrator 支援網站
<i>Release Notes for IBM DB2 Information Integrator Classic Federation for z/OS</i>	N/A	DB2 Information Integrator 支援網站

z/OS 抄寫功能相關文件

z/OS 抄寫功能相關文件

目的

z/OS 抄寫功能相關文件。

表 8. z/OS 抄寫功能的相關 DB2 Information Integrator 文件

名稱	書號	位置
<i>ASNCLP Program Reference for Replication and Event Publishing</i>	N/A	DB2 Information Integrator 支援網站
<i>Introduction to Replication and Event Publishing</i>	GC18-7567	DB2 Information Integrator 支援網站
<i>Migrating to SQL Replication</i>	N/A	DB2 Information Integrator 支援網站
<i>Replication and Event Publishing Guide and Reference</i>	SC18-7568	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>Replication Installation and Customization Guide for z/OS</i>	SC18-9127	DB2 Information Integrator 支援網站
<i>SQL Replication Guide and Reference</i>	SC27-1121	<ul style="list-style-type: none"> • DB2 PDF 文件光碟 • DB2 Information Integrator 支援網站
<i>Tuning for Replication and Event Publishing Performance</i>	N/A	DB2 Information Integrator 支援網站
<i>Tuning for SQL Replication Performance</i>	N/A	DB2 Information Integrator 支援網站

表 8. z/OS 抄寫功能的相關 DB2 Information Integrator 文件 (繼續)

名稱	書號	位置
<i>Release Notes for IBM DB2 Information Integrator Standard Edition, Advanced Edition, and Replication for z/OS</i>	N/A	<ul style="list-style-type: none"> 在 DB2 Information Center 中，產品概觀 > 資訊整合 > DB2 Information Integrator 概觀 > 問題、可行方案及文件更新 DB2 Information Integrator 安裝啓動程式 DB2 Information Integrator 支援網站 <i>DB2 Information Integrator</i> 產品光碟

Linux、UNIX 及 Windows 企業搜尋功能的相關文件

Linux、UNIX 及 Windows 企業搜尋功能的相關文件

目的

Linux、UNIX 及 Windows 企業搜尋功能的相關文件。

表 9. Linux、UNIX 及 Windows 企業搜尋功能的相關 DB2 Information Integrator 文件

名稱	書號	位置
管理企業搜尋	SC40-0934	DB2 Information Integrator 支援網站
企業搜尋安裝手冊	GC40-0933	DB2 Information Integrator 支援網站
企業搜尋程式設計指南與 API 參考手冊	SC40-0935	DB2 Information Integrator 支援網站
企業搜尋版本注意事項	N/A	DB2 Information Integrator 支援網站

版本注意事項及安裝需求

版本注意事項提供產品的版本及修正套件層次資訊，並包括每個版本的最新修訂文件。安裝需求則提供產品版本的特定資訊。

表 10. DB2 Information Integrator 版本注意事項及安裝需求

名稱	檔名	位置
<i>Installation Requirements for IBM DB2 Information Integrator Event Publishing Edition, Replication Edition, Standard Edition, Advanced Edition, Advanced Edition Unlimited, Developer Edition, and Replication for z/OS</i>	Prereqs	<ul style="list-style-type: none"> <i>DB2 Information Integrator</i> 產品光碟 DB2 Information Integrator 安裝啓動程式

表 10. DB2 Information Integrator 版本注意事項及安裝需求 (繼續)

名稱	檔名	位置
<i>Release Notes for IBM DB2 Information Integrator Standard Edition, Advanced Edition, and Replication for z/OS</i>	ReleaseNotes	<ul style="list-style-type: none"> 在 DB2 資訊中心，產品概觀 > 資訊整合 > DB2 Information Integrator 概觀 > 問題、可行方案及文件更新 DB2 Information Integrator 安裝啟動程式 DB2 Information Integrator 支援網站 <i>DB2 Information Integrator</i> 產品光碟
<i>Release Notes for IBM DB2 Information Integrator Event Publisher for IMS for z/OS</i>	N/A	DB2 Information Integrator 支援網站
<i>Release Notes for IBM DB2 Information Integrator Event Publisher for VSAM for z/OS</i>	N/A	DB2 Information Integrator 支援網站
<i>Release Notes for IBM DB2 Information Integrator Classic Federation for z/OS</i>	N/A	DB2 Information Integrator 支援網站
企業搜尋版本注意事項	N/A	DB2 Information Integrator 支援網站

檢視版本注意事項及安裝需求

檢視版本注意事項及安裝需求

目的

若要在 Windows 作業系統上檢視光碟中的版本注意事項及安裝需求，請輸入：

```
x\doc\%L
```

參數

x Windows 光碟機代號

%L

您要使用的文件語言環境，如 en_US。

目的

若要在 UNIX 作業系統上檢視光碟中的版本注意事項及安裝需求，請輸入：

```
/cdrom/doc/%L
```

參數

cdrom

光碟的 UNIX 裝載點

%L

您要使用的文件語言環境，如 en_US。

檢視及列印 PDF 文件

檢視及列印 PDF 文件

若要檢視及列印 *DB2 PDF* 文件光碟的 DB2 Information Integrator PDF 書籍

1. 從 *DB2 PDF* 文件光碟的根目錄開啓 `index.htm` 檔案。
2. 按一下您要使用的語言。
3. 按一下您要檢視的文件鏈結。

存取 DB2 Information Integrator 文件

存取 DB2 Information Integrator 文件

所有的 DB2 Information Integrator 書籍與版本注意事項 (PDF 檔案) 都可以從 DB2 Information Integrator 支援網站取得，網址是：
www.ibm.com/software/data/integration/db2ii/support.html。

若要存取最新的 DB2 Information Integrator 產品文件，請在 DB2 Information Integrator 支援網站上按一下 Product Information 鏈結，如第 83 頁的圖 6 所示。

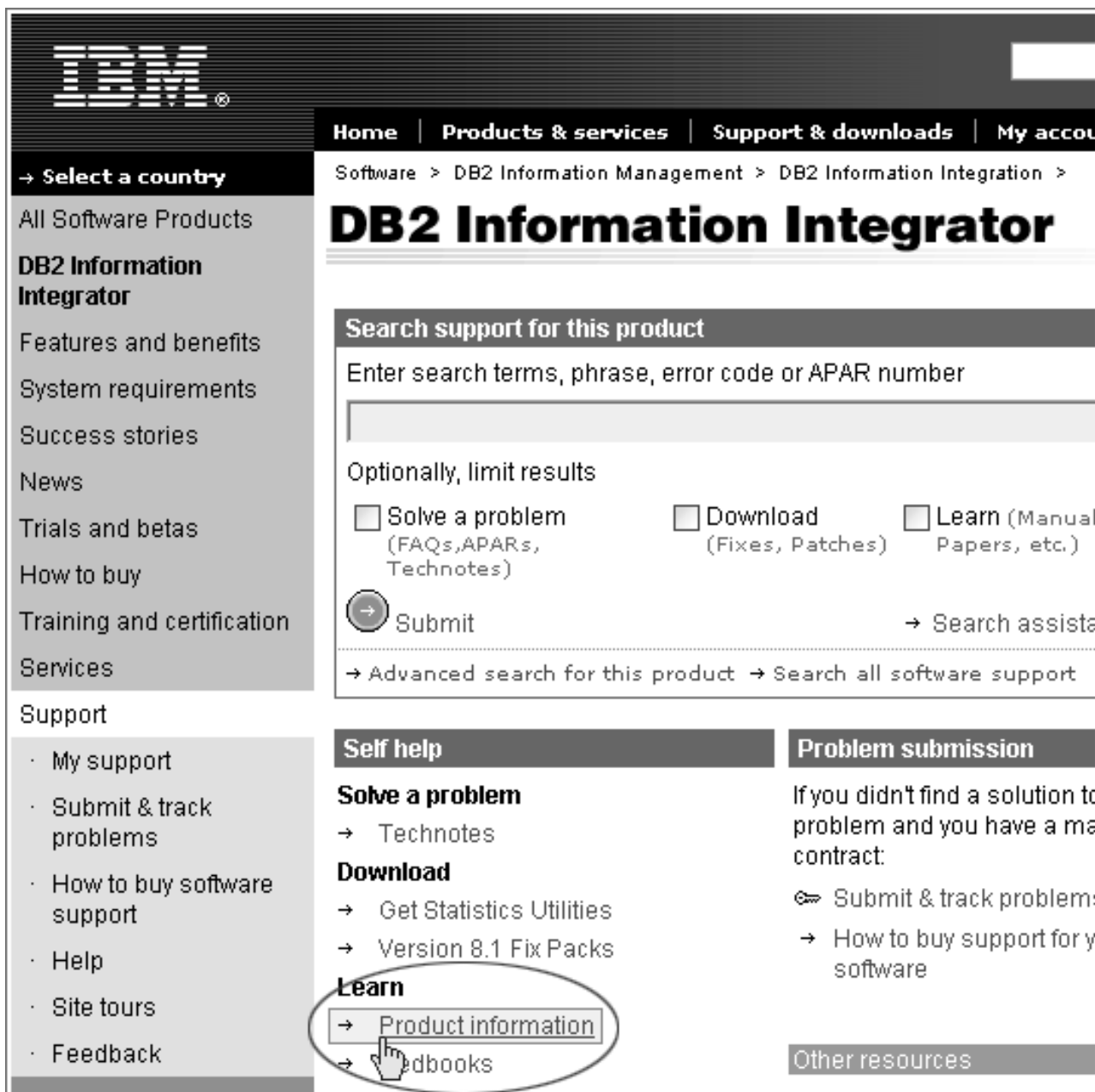


圖 6. DB2 Information Integrator 支援網站上的產品資訊鏈結

您可以從 Product Information 鏈結存取所有支援語言版本的最新 DB2 Information Integrator 文件：

- DB2 Information Integrator 產品文件 (PDF 檔案)
- 修正套件產品文件，包括版本注意事項
- 下載及安裝 DB2 Information Center for Linux、UNIX 及 Windows 的操作指示
- 線上 DB2 資訊中心的相關鏈結

DB2 Information Integrator 支援網站也提供支援文件、IBM 紅皮書、白皮書、產品下載、使用者群組鏈結，以及 DB2 Information Integrator 的相關消息。

協助工具

協助工具特性可幫助行動不便或視覺障礙等殘障使用者順利地使用軟體產品。下列清單是 DB2[®] 第 8 版產品的主要協助工具功能：

- 所有 DB2 功能都可以使用鍵盤來導覽，而無須使用滑鼠。如需詳細資訊，請參閱『鍵盤輸入與導覽』。
- 您可以自訂 DB2 使用者介面字型的大小及顏色。如需詳細資訊，請參閱『容易使用的顯示畫面』。
- DB2 產品支援使用 Java™ Accessibility API 的協助工具應用程式。如需詳細資訊，請參閱第 86 頁的『與輔助技術的相容性』。
- DB2 文件都是用方便存取的格式提供。如需詳細資訊，請參閱第 86 頁的『方便存取的文件』。

鍵盤輸入與導覽

鍵盤焦點

鍵盤焦點

在 UNIX[®] 作業系統中，會強調顯示可使用按鍵的作用中視窗區域。

鍵盤輸入

鍵盤輸入

只要使用鍵盤就可以操作 DB2 工具。您可以使用按鍵或按鍵組合來執行作業，也可以由滑鼠來執行。標準作業系統按鍵用於標準作業系統作業。

如需使用按鍵或按鍵組合來執行作業的相關資訊，請參閱鍵盤捷徑及快速鍵：一般 GUI 說明。

鍵盤導覽

鍵盤導覽

您可以使用按鍵或按鍵組合來導覽 DB2 工具使用者介面。

如需使用按鍵或按鍵組合來導覽 DB2 工具的相關資訊，請參閱鍵盤捷徑及快速鍵：一般 GUI 說明。

容易使用的顯示畫面

容易使用的顯示畫面

目的

容易使用的顯示畫面

字型設定

字型設定

您可以使用「工具設定」筆記本來選取功能表與對話視窗中文字的顏色、大小及字型。

如需指定字型設定的相關資訊，請參閱變更功能表及文字的字型：一般 GUI 說明。

不依靠顏色

不依靠顏色

您不需區分顏色，即可使用本產品的任一功能。

與輔助技術的相容性

與輔助技術的相容性

DB2 工具介面支援 Java Accessibility API，可讓您與螢幕讀取器及其他輔助技術搭配使用。

方便存取的文件

方便存取的文件

DB2 文件以 XHTML 1.0 格式提供，可在大多數 Web 瀏覽器上瀏覽。XHTML 可讓您根據瀏覽器的顯示偏好設定來檢視文件。您也可以使用螢幕讀取器及其他輔助技術。

語法圖解則以帶點十進位格式提供。只有使用螢幕讀取器存取線上文件時，才可以用這種格式。

注意事項

本資訊是針對 IBM 在美國所提供之產品與服務開發出來的。IBM 不見得會對所有國家或地區都提供本文件所提的各項產品、服務或功能。要知道在您所在地區是否可得到這些產品及服務時，請向當地的 IBM 服務代表查詢。而此處任何對於 IBM 產品、程式或服務的參考之處，並不表示或暗示只可以使用 IBM 的產品、程式或服務。任何未侵犯 IBM 的智慧財產權，任何功能相當的產品、程式或服務都可以取代 IBM 的產品、程式或服務。不過，使用者必須自行負責評估和驗證任何非 IBM 產品、程式或服務的作業。

在本文件中可能包含著 IBM 所擁有之專利或擱置專利申請的內容。本文件使用者並不享有前述專利之任何授權。您可以用書面方式來查詢授權，來函請寄到：IBM Director of Licensing IBM Corporation North Castle Drive Armonk, NY 10504-1785 U.S.A.

若要查詢二位元組 (DBCS) 資訊的授權事宜，請連絡您國家或地區的 IBM 智慧財產部門，或者用書面方式寄到：IBM World Trade Asia Corporation Licensing 2-31 Roppongi 3-chome, Minato-ku Tokyo 106-0032, Japan

下列段落不適用於英國或任何其他與當地法律相抵觸的國家或地區：IBM 公司係以『現狀』提供本出版品，且不作任何明示或默示的保證，包括但不僅限於非侵害、可售性或符合特定用途之暗示保證。有些地區不允許放棄在特定交易中的明示或默示保證，因此，這項聲明對您可能不適用。

本書中可能會有技術上的錯誤或排版印刷上的訛誤。因此，IBM 會定期修訂；並將修訂後的內容納入新版中。IBM 得隨時修改及/或變更本書中所說明的產品及/或程式，恕不另行通知。

本資訊中任何對非 IBM 網站的敘述僅供參考，為便利貴客戶之使用，而非為該網站背書。這些網站中的資料，並不包含在 IBM 產品的資料中，使用網站中的資料，須自行負擔風險。

在不造成您困擾或損及您個人權益的前提下，IBM 得以適切使用或散佈您以各種型式所提供的相關資訊。

本程式之獲授權者若希望取得本程式之相關資訊，以便達到下列目的：(i) 在獨立建立的程式與其他程式（包括本程式）之間交換資訊；以及 (ii) 相互使用已交換的資訊。則請與位於下列地址之人員連絡：

IBM Corporation J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003 U.S.A.

上述資料之取得有其條件，在某些情況下必須付費方得使用。

IBM 基於「IBM 客戶合約」、「IBM 國際程式授權合約」或雙方之間任何同等的合約等條款，提供本文件中所說的授權程式與其所有適用的授權資料。

任何此處涵蓋的執行效能資料都是在一個受控制的環境下決定出來的。因此，若在其他作業環境下，所得的結果可能會大大不同。有些測定已在開發階段系統上做過，不

過這並不保證在一般系統上會出現相同結果。再者，有些測定可能已透過推測方式評估過。但實際結果可能並非如此。本文件的使用者應依自己的特定環境，查證適用的資料。

非 IBM 產品的相關資訊，取自該產品供應商、發佈的聲明或其他公共來源。IBM 未測試這些產品，因此無法確認非 IBM 產品的效能、相容性或其他聲明。有關非 IBM 產品的功能問題，請洽該產品供應商。

有關 IBM 未來動向的任何陳述，僅代表 IBM 的目標而已，並可能於未事先聲明的情況下有所變動或撤回。

這個資訊中包含每日業務使用的報告和資料範例。為使說明盡可能完備，範例中包含個人、公司、品牌和產品的名稱。此等名稱皆屬虛構，凡有類似實際個人或企業所用之名稱及地址者，皆屬巧合。

著作權授權：

本資訊可包含原始語言的範例應用程式，用以說明各種作業平台上的程式設計技術。貴客戶得為開發、使用、行銷或散佈運用樣本程式之作業平台的應用程式程式介面所撰寫的應用程式之目的，免費複製、修改並散佈這些樣本程式。此等範例並未在所有情況下完整測試。故 IBM 不保證或默示保證這些樣本程式之可靠性、服務性或功能。貴客戶得為開發、使用、行銷或散佈符合 IBM 應用程式設計介面的應用程式之目的，免費複製、修改並散佈這些樣本程式。

這些範例程式的每個複本或任何部分，或任何衍生作品都必須包括以下版權聲明：

Outside In (®) Viewer Technology, © 1992-2004 Stellent, Chicago, IL., Inc. All Rights Reserved.

IBM XSLT Processor Licensed Materials - Property of IBM © Copyright IBM Corp., 1999-2004. All Rights Reserved.

商標

本主題列出 IBM 商標和某些非 IBM 商標。

下列術語是 IBM 公司在美國和/或其他國家或地區的商標：

IBM AIX、AIX 5L
DB2
DB2 Universal Database
Domino
Informix
Lotus
Lotus Notes
Notes
OmniFind
WebSphere
z/OS

下列術語是其他公司的商標或註冊商標：

Java 及所有以 Java 為基礎的商標和標誌是 Sun Microsystems, Inc. 在美國和/或其他國家或地區的商標或註冊商標。

Microsoft、Windows、Windows NT 和 Windows 標誌是 Microsoft Corporation 在美國和/或其他國家或地區的商標。

Intel、Intel Inside（標誌）、MMX 和 Pentium 是 Intel Corporation 在美國和/或其他國家或地區的商標。

UNIX 是 The Open Group 在美國和其他國家或地區的註冊商標。

Linux 是 Linus Torvalds 在美國和/或其他國家或地區的商標。

其他公司、產品或服務名稱，可能是其他公司的商標或服務標誌。

連絡 IBM

若要在美國或加拿大連絡 IBM 客戶服務中心，請撥 1-800-IBM-SERV (1-800-426-7378)。

若要瞭解可用的服務選項，請撥打下列號碼之一：

- 美國：1-888-426-4343
- 加拿大：1-800-465-9600

若要尋找您所在國家或地區的 IBM 辦事處，請參閱 Web 上的 IBM Directory of Worldwide Contacts，網址是：www.ibm.com/planetwide。

取得產品資訊

DB2 Information Integrator 的相關資訊可透過電話或從 Web 取得。

DB2 Information Integrator 的相關資訊可透過電話或從 Web 取得。這裡提供的電話號碼適用於美國。

1. 若要訂購產品或取得一般資訊，請撥 1-800-IBM-CALL (1-800-426-2255)
2. 若要訂購出版品，請撥 1-800-879-2755
3. 請造訪網站：www.ibm.com/software/data/integration/db2ii/support.html。

此網站提供下列最新資訊：

- 技術叢書
- 訂購書籍
- 用戶端下載
- 新聞群組
- 修正套件
- 新聞
- Web 資源鏈結

提供關於文件的意見

如果您對本書或其他 DB2 Information Integrator 文件有任何意見，請將意見寄給我們。

您的意見將幫助 IBM 提供高品質的資訊。若您對本書或其他 DB2 Information Integrator 文件有任何意見，可以透過以下任一方法來表達意見：

1. 使用 www.ibm.com/software/data/rcf 的線上讀者意見表來傳送您的意見。
2. 請用電子郵件將您的意見寄到 comments@us.ibm.com。在郵件中註明產品名稱、版本編號，以及書名及產品編號（如適用）。如果您是針對特定的一段文字表達意見，請註明該段文字的位置（如標題、表格編號或頁碼）。

索引

索引順序以中文字，英文字，及特殊符號之次序排列。

〔十五劃〕

- 範例應用程式 56
 - 資料接聽器 API 71
 - 基本 72
 - 至少必要的 57
 - 擷取所有搜尋結果 61
 - 瀏覽及導覽 59

〔十六劃〕

- 機密保護 3
- 資料接聽器 API 69
 - 範例應用程式 71
 - 基本 72
 - 新增資料 66
 - 概觀 3
 - 移除資料 65
 - 屬性 64
 - 需求 64
- 資料接聽器元件 63
- 資料控制項
 - 新增資料 65

A

- addArguments 方法 55
- addMetaField 方法 71
- AdvancedSearchExample 類別 14
- API 概觀 3
- ApplicationInfo 介面 27
- ATTRIBUTE_DOCTYPE 常數 23
- ATTRIBUTE_LANGUAGE 常數 23
- ATTRIBUTE_SOURCE 常數 23

B

- BrowseExample 類別 17
- BrowseFactory 介面 47
- BrowseService 介面 48

C

- Category 介面 48
- CategoryInfo 介面 20
- CollectionInfo 介面 21

- count 方法 23
- createApplicationInfo 方法 20, 29, 47
- createDataSourceMetadata 方法 71
- createQuery 方法 19, 29
- createSearchFactory 方法 19

D

- DLDataPusher 類別 69

F

- FieldInfo 介面 21

G

- getACLConstraints 方法 39
- getArguments 方法 55
- getAvailableAttributeValues 方法 24
- getAvailableFields 方法 21, 25
- getAvailableNumberOfResults 方法 44
- getAvailableSearchables 方法 26, 27
- getBrowseService 方法 47
- getCategories 方法 41
- getCategory 方法 49, 50
- getChildren 方法 49
- getCollectionInfo 方法 21, 26
- getConfidence 方法 43
- getDate 方法 41
- getDefaultLanguage 方法 24
- getDescription 方法 41
- getDocumentID 方法 42
- getDocumentSource 方法 42
- getDocumentType 方法 42
- getEstimatedNumberOfResults 方法 45
- getFields 方法 41, 42
- getFirstRequestedResult 方法 33
- getID 方法 20, 21, 28, 56
- getInfo 方法 43, 49
- getLabel 方法 20, 56
- getLanguage 方法 42
- getLocalizedMessage 方法 56
- getMessage 方法 55
- getName 方法 40
- getNumRequestedResults 方法 33
- getPassword 方法 28
- getPathFromRoot 方法 43, 49
- getPredefinedResults 方法 45
- getProperties 方法 25, 34, 42, 45
- getProperty 方法 25, 33, 43, 45

- getQueryEvaluationTime 方法 45
- getQueryID 方法 39
- getQueryLanguage 方法 32
- getQuerySubstring 方法 46
- getResultCategories DetailLevel 方法 40
- getResults 方法 45
- getReturnedFields 方法 32
- getRootCategory 方法 49, 50
- getScore 方法 43
- getSearchable 方法 26, 27
- getSearchService 方法 19
- getSeverity 方法 54
- getSeverityDescription 方法 54
- getSiapiVersion 方法 18
- getSortKey 方法 34
- getSortOrder 方法 35
- getSortPoolSize 方法 34
- getSpellCorrections 方法 24, 45
- getSuggestions 方法 47
- getTaxonomyBrowser 方法 48
- getTaxonomyID 方法 44
- getTaxonomyInfo 方法 50, 56
- getText 方法 31
- getTitle 方法 43
- getToken 方法 28
- getType 方法 54
- getTypeDescription 方法 54
- getValue 方法 41
- getVersion 方法 20, 47

H

- hasUnconstrainedResults 方法 46

I

- isAttributeReturned 方法 38
- isContentSearchable 方法 21
- isEvaluationTruncated 方法 46
- isFieldSearchable 方法 22
- isFirstOfASite 方法 43
- isParametric 方法 22
- isPredefinedResultsEnabled 方法 36
- isReturnable 方法 22
- isSiteCollapsingEnabled 方法 36
- isSpellCorrectionEnabled 方法 37

J

- Java 原始碼 12

M

MAX_CONFIDENCE 常數 44
MIN_CONFIDENCE 常數 44

N

NameValuePair 介面 40

P

printStackTrace 方法 55
pushData 方法 69

Q

Query 介面 29

R

removeURIs 方法 70
removeURLs 方法 70
resetReturnedFields 方法 33
Result 介面 41
ResultCategory 介面 43
ResultSet 介面 44
RESULT_CATEGORIES_ALL 常數 30
RESULT_CATEGORIES_NO_PATH_TO_ROOT 常數 30
RETURN_RESULT_CATEGORIES 常數 31
RETURN_RESULT_DATE 常數 31
RETURN_RESULT_FIELDS 常數 31
RETURN_RESULT_LANGUAGE 常數 31
RETURN_RESULT_SCORE 常數 31
RETURN_RESULT_TITLE 常數 30
RETURN_RESULT_TYPE 常數 31
RETURN_RESULT_DATE 常數 30
RETURN_RESULT_SOURCE 常數 31

S

search 方法 23
Searchable 介面 23
SearchExample 類別 12
SearchFactory 介面 19
SearchService 介面 27
setACLConstraints 39
setPassword 方法 28
setPredefinedResultsEnabled 方法 36
setProperty 方法 25, 34
setQueryID 方法 39
setQueryLanguage 方法 40
setRequestedResultRange 方法 33

setResultCategoriesDetailLevel 方法 37
setReturnedAttribute 方法 37
setReturnedFields 方法 32
setSiteCollapsingEnabled 方法 36
setSortKey 方法 35
setSortPoolSize 方法 35
setSpellCorrectionEnabled 方法 36
setText 方法 32, 40
setToken 方法 28
SEVERITY_ERROR 常數 50
SEVERITY_FATAL_ERROR 常數 51
SI-API 5, 18
 範例應用程式 56
 使用 5
 取得 Searchable 6
 取得搜尋服務 5
 取得實作 5
 概觀 3
 發出查詢 6
 處理查詢結果 6
SiapiException 類別 50
SiapiSearchImpl 類別 19
SiapiVersion 類別 18
SORT_KEY_DATE 常數 30
SORT_KEY_NONE 常數 30
SORT_KEY_RELEVANCE 常數 29
SORT_ORDER_ASCENDING 常數 30
SORT_ORDER_DESCENDING 常數 30
SpellCorrection 介面 46

T

TaxonomyBrowser 介面 50
TaxonomyInfo 介面 56
TYPE_DOC_EXIST_ERROR 常數 51
TYPE_DOC_NOT_FOUND_ERROR 常數 51
TYPE_ILLEGAL_RESULTS_RANGE 常數 53
TYPE_ILLEGAL_VALUE_ERROR 常數 52
TYPE_IMPL_FACTORY_ERROR 常數 52
TYPE_INDEX_CORRUPTED 常數 53
TYPE_INDEX_DOES_NOT_EXIST 常數 52
TYPE_IO_ERROR 常數 52
TYPE_QUERY_SYNTAX_ERROR 常數 54
TYPE_SEARCH_ENGINE_STATE_ERROR 常數 51
TYPE_TOO_FEW_VALUES 常數 53
TYPE_TOO_MANY_VALUES 常數 53
TYPE_UNKNOWN_ENCODING 常數 53
TYPE_UNKNOWN_ERROR 常數 51
TYPE_UNSUPPORTED_OPERATION 常數 52

U

URI 碼 64
查詢行爲 7
查詢語法 8
其他資訊 1
介面, API
 ApplicationInfo 27
 BrowseFactory 47
 BrowseService 48
 CategoryInfo 20
 CollectionInfo 21
 FieldInfo 21
 NameValuePair 40
 Query 29
 Result 41
 ResultCategory 43
 ResultSet 44
 Searchable 23
 SearchFactory 19
 SearchService 27
 SpellCorrection 46
 TaxonomyBrowser 50
 TaxonomyInfo 56
 種類 48
方法, API
 計數 23
 addArguments 55
 addMetaField 71
 createApplicationInfo 20, 29, 47
 createDataSourceMetadata 71
 createQuery 19, 29
 createSearchFactory 19
 getACLConstraints 39
 getArguments 55
 getAvailableAttributeValues 24
 getAvailableFields 21, 25
 getAvailableNumberOfResults 44
 getAvailableSearchables 26, 27
 getBrowseService 47
 getCategories 41
 getCategory 49, 50
 getChildren 49
 getCollectionInfo 21, 26
 getConfidence 43
 getDate 41
 getDefaultLanguage 24
 getDescription 41
 getDocumentID 42
 getDocumentSource 42
 getDocumentType 42
 getEstimatedNumberOfResults 45
 getFields 41, 42
 getFirstRequestedResult 33
 getID 20, 21, 28, 56
 getInfo 43, 49

方法, API (繼續)

getLabel 20, 56
getLanguage 42
getLocalizedMessage 56
getMessage 55
getName 40
getNumRequestedResults 33
getPassword 28
getPathFromRoot 43, 49
getPredefinedResults 45
getProperties 25, 34, 42, 45
getProperty 25, 33, 43, 45
getQueryEvaluationTime 45
getQueryID 39
getQueryLanguage 32
getQuerySubstring 46
getResultCategories DetailLevel 40
getResults 45
getReturnedFields 32
getRootCategory 49, 50
getScore 43
getSearchable 26, 27
getSearchService 19
getSeverity 54
getSeverityDescription 54
getSiapiVersion 18
getSortKey 34
getSortOrder 35
getSortPoolSize 34
getSpellCorrections 24, 45
getSuggestions 47
getTaxonomyBrowser 48
getTaxonomyID 44
getTaxonomyInfo 50, 56
getText 31
getTitle 43
getToken 28
getType 54
getTypeDescription 54
getValue 41
getVersion 20, 47
hasUnconstrainedResults 46
isAttributeReturned 38
isContentSearchable 21
isEvaluationTruncated 46
isFieldSearchable 22
isFirstOfASite 43
isParametric 22
isPredefinedResultsEnabled 36
isReturnable 22
isSiteCollapsingEnabled 36
isSpellCorrectionEnabled 37
printStackTrace 55
pushData 69
removeURIs 70
removeURLs 70

方法, API (繼續)

resetReturnedFields 33
setACLConstraints 39
setPassword 28
setPredefinedResultsEnabled 36
setProperty 25, 34
setQueryID 39
setQueryLanguage 40
setRequestedResultRange 33
setResultCategoriesDetailLevel 37
setReturnedAttribute 37
setReturnedFields 32
setSiteCollapsingEnabled 36
setSortKey 35
setSortPoolSize 35
setSpellCorrectionEnabled 36
setText 32, 40
setToken 28
搜尋 23

常數, API

ATTRIBUTE_DOCTYPE 23
ATTRIBUTE_LANGUAGE 23
ATTRIBUTE_SOURCE 23
MAX_CONFIDENCE 44
MIN_CONFIDENCE 44
RESULT_CATEGORIES_ALL 30
RESULT_CATEGORIES_NO_PATH_TO_ROOT 30
RETURN_RESULT_CATEGORIES 31
RETURN_RESULT_DATE 31
RETURN_RESULT_FIELDS 31
RETURN_RESULT_LANGUAGE 31
RETURN_RESULT_SCORE 31
RETURN_RESULT_TITLE 30
RETURN_RESULT_TYPE 31
RETURN_RESULT_DATE 30
RETURN_RESULT_SOURCE 31
SEVERITY_ERROR 50
SEVERITY_FATAL_ERROR 51
SORT_KEY_DATE 30
SORT_KEY_NONE 30
SORT_KEY_RELEVANCE 29
SORT_ORDER_ASCENDING 30
SORT_ORDER_DESCENDING 30
TYPE_DOC_EXIST_ERROR 51
TYPE_DOC_NOT_FOUND_ERROR 51
TYPE_ILLEGAL_RESULTS_RANGE 53
TYPE_ILLEGAL_VALUE_ERROR 52
TYPE_IMPL_FACTORY_ERROR 52
TYPE_INDEX_CORRUPTED 53
TYPE_INDEX_DOES_NOT_EXIST 52
TYPE_IO_ERROR 52
TYPE_QUERY_SYNTAX_ERROR 欄位

54

常數, API (繼續)

TYPE_SEARCH_ENGINE_STATE_ERROR 51
TYPE_TOO_FEW_VALUES 53
TYPE_TOO_MANY_VALUES 53
TYPE_UNKNOWN_ENCODING 53
TYPE_UNKNOWN_ERROR 51
TYPE_UNSUPPORTED_OPERATION 52

連絡 IBM 1

移除資料 65

〔十九劃〕

類別

SiapiVersion 18

類別, API

AdvancedSearchExample 14

BrowseExample 17

DLDataPusher 69

SearchExample 12

SiapiException 50

SiapiSearchImpl 19



SC40-0935-00

