

Novell exteNd Composer™ LDAP Connect

4.2

www.novell.com

ユーザガイド



Novell®

保証と著作権

Copyright ©1999, 2000, 2001, 2002, 2003 SilverStream Software, LLC. All rights reserved.

SilverStream ソフトウェア製品は、SilverStream Software LLC により著作権とすべての権利が保留されています。

SilverStream は SilverStream Software, LLC の登録商標です。Novell は、Novell, Inc. の登録商標です。

ソフトウェアとマニュアルの所有権、および特許、著作権、およびそれに関連するその他のすべての財産権は常に、単独で排他的に SilverStream とそのライセンサーに保留され、当該所有権と矛盾するいかなる行為も行わないものとします。本ソフトウェアは、著作権法と国際条約規定で保護されています。ソフトウェアならびにそのマニュアルからすべての著作権に関する通知とその他の所有権に関する通知を削除してはならず、ソフトウェアとそのマニュアルのすべてのコピーまたは抜粋に当該通知を複製しなければなりません。本ソフトウェアのいかなる所有権も取得するものではありません。

Jakarta-Regexp Copyright ©1999 The Apache Software Foundation. All rights reserved. Ant Copyright ©1999 The Apache Software Foundation. All rights reserved. Xalan Copyright ©1999 The Apache Software Foundation. All rights reserved. Xerces Copyright ©1999-2000 The Apache Software Foundation. All rights reserved. Jakarta-Regexp, Ant, Xalan, Crimson, および Xerces ソフトウェアは、The Apache Software Foundation によりライセンスを付与され、Jakarta-Regexp, Ant, Xalan, Crimson, および Xerces のソースおよびバイナリ形式での再配布および使用は、変更のあるなしにかかわらず、以下の条件が満たされることを前提として許可されます。1. ソースコードの再配布に上記の著作権に関する通知、条件のリスト、および以下の権利放棄に関する通知が記載されていること。2. バイナリ形式の再配布では上記の著作権に関する通知、条件のリスト、および以下の権利放棄に関する通知がマニュアルまたは配布の際に提供されるその他の資料、あるいはその両方に記載されていること。3. エンドユーザの資料には、適宜、以下の通知を再配布の際に含めてください。「この製品には、Apache Software Foundation (<http://www.apache.org>) により開発されたソフトウェアが含まれています」代わりに、この謝辞をソフトウェア自体に表示し、当該サードパーティに対する謝辞が通常表示される場所に表示することもできます。4. 「The Jakarta Project」、「Jakarta-Regexp」、「Xerces」、「Xalan」、「Ant」、および「Apache Software Foundation」は、書面による事前の許可なく、このソフトウェアから派生する製品を推薦したり、販売促進したりするのに使用してはなりません。書面による許可については、apache@apache.org <<mailto:apache@apache.org>> にお問い合わせください。5. 本ソフトウェアから派生する製品は「Apache」と呼ばれてはならず、「Apache」は The Apache Software Foundation の事前の書面による許可なくその名前に使用することはできません。本ソフトウェアは「現状のまま」提供され、いかなる明示的、暗黙の保証も行われるものではありません。販売可能性や特定の目的に対する適合性に対する暗黙の保証も行われません。いかなる場合でも、Apache Software Foundation またはその関係者はいかなる直接的、間接的、偶発的、特別な、免除的、または結果的な損害（代替品やサービスの調達、使用機会、データ、または利益の喪失、または業務の中断などを含む）についても、理論上責任がある場合でも、契約上の責任がある場合でも、厳密な責任、または瑕疵（怠慢などを含む）があった場合でも、ソフトウェアの使用の過程で生じ、当該損害の可能性を助言した場合であっても、責任を持ちません。

Copyright ©2000 Brett McLaughlin & Jason Hunter. All rights reserved. ソースおよびバイナリ形式での再配布および使用は、変更のあるなしにかかわらず、以下の条件が満たされることを前提として許可されます。1. ソースコードの再配布に上記の著作権に関する通知、条件のリスト、および以下の権利放棄に関する通知が記載されていること。2. バイナリ形式の再配布では上記の著作権に関する通知、条件のリスト、および以下の権利放棄に関する通知がマニュアルまたは配布の際に提供されるその他の資料、あるいはその両方に記載されていること。3. 「JDOM」という名前は、書面による事前の許可なく、このソフトウェアから派生する製品を推薦したり、販売促進したりするのに使用してはなりません。書面による許可については、license@jdom.org <<mailto:license@jdom.org>> にお問い合わせください。4. 本ソフトウェアから派生する製品は「JDOM」と呼ばれてはならず、「JDOM」は JDOM Project Management (pm@jdom.org) <<mailto:pm@jdom.org>> の事前の書面による許可なくその名前に使用することはできません。本ソフトウェアは「現状のまま」提供され、いかなる明示的、暗黙の保証も行われるものではありません。販売可能性や特定の目的に対する適合性に対する暗黙の保証も行われません。いかなる場合でも、Apache Software Foundation またはその関係者はいかなる直接的、間接的、偶発的、特別な、免除的、または結果的な損害（代替品やサービスの調達、使用機会、データ、または利益の喪失、または業務の中断などを含む）についても、理論上責任がある場合でも、契約上の責任がある場合でも、厳密な責任、または瑕疵（怠慢などを含む）があった場合でも、ソフトウェアの使用の過程で生じ、当該損害の可能性を助言した場合であっても、責任を持ちません。

Sun Microsystems, Inc. Sun, Sun Microsystems, Sun Logo Sun, Sun のロゴ、Sun Microsystems, JavaBeans, Enterprise JavaBeans, JavaServer Pages, Java Naming and Directory Interface, JDK, LDAP, Java, HotJava, HotJava Views, Visual Java, Solaris, NEO, Joe, Netra, NFS, ONC, ONC+, OpenWindows, PC-NFS, SNM, SunNet Manager, Solaris sunburst design, Solstice, SunCore, SolarNet, SunWeb, Sun Workstation, The Network Is The Computer, ToolTalk, Ultra, Ultracomputing, Ultraserver, Where The Network Is Going, SunWorkShop, XView, Java WorkShop, Java Coffee Cup のロゴ、Visual Java、および NetBeans は、米国およびその他の国の Sun Microsystems, Inc. の商標ならびに登録商標です。

Copyright ©2001 Extreme! Lab, Indiana University License. <http://www.extreme.indiana.edu>. 同社により許可が無料で、Indiana University ソフトウェアと関連する Indiana University のドキュメントファイル (「IU Software」) のコピーを取得したすべての人に、制限なく IU Software を取り扱うために付与されます。その際に、IU Software の使用、コピー、変更、マージ、公開、配布、サブライセンス、または販売、あるいはそれらのすべてに関する権利に制限はなく、IU Software が指定した人に以下の条件に基づき権利を付与します。上記の著作権に関する通知とその許可に関する通知は、IU Software のすべてのコピーおよび主要部分に含まれる必要があります。本 IU ソフトウェアは「現状のまま」提供され、いかなる明示的、暗黙の保証も行われるものではありません。販売可能性、特定の目的に対する適合性や権利侵害がないことに対する暗黙の保証も行われません。いかなる場合でも、作成者または著作権所有者は、契約上の責任がある場合でも、厳密な責任、または瑕疵 (怠慢などを含む) があつた場合でも、IU Software に関連して、または IU Software の使用やその他の取引の過程で生じた場合であっても、クレーム、損害、その他の責任について責任を持ちません。

本ソフトウェアは、著作権を持つ SSLava™ Toolkit の一部です。Copyright ©1996-1998 by Phaos Technology Corporation. All Rights Reserved.

Copyright © 1994-2002 W3C/E (Massachusetts Institute of Technology, Institut National de Recherche Informatique et en Automatique, Keio University), all Rights Reserved. <http://www.w3.org/consortium/legal>. この W3C の成果物 (ソフトウェア、ドキュメント、またはその他の関連品目を含む) は、以下のライセンスの下で著作権所有者により提供されています。この成果物の取得、使用、またはコピー、あるいはそれらのすべてにより、ライセンサーは以下の条件を読み、理解し、遵守することに合意するものとします。本ソフトウェアとそのドキュメントの使用、コピー、変更、および配布は、変更のあるなしにかかわらず、いかなる目的でも無料または本契約で許可された使用料をもって許可されます。ただし、変更箇所を含む本ソフトウェアとドキュメントのすべてまたはその一部に以下のとおり記述することを前提とします。1. この通知の全文は、再配布物または派生物のユーザが見やすい場所に掲示しなければなりません。2. すべての前もって存在する知的所有権の放棄、通知、または条件。存在しない場合は、以下の形式の短い通知 (ハイパーテキストが望ましい、テキストでも良い) を再配布または派生コードの本文内で使用しなければなりません。「Copyright © [date-of-software] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University). All Rights Reserved. <http://www.w3.org/Consortium/Legal/>」3. W3C のファイルに変更または修正を加えた場合はその日付を含む通知。(コードが派生する場所への URI を示すことをお勧めします。) 本ソフトウェアは「現状のまま」提供され、いかなる明示的、暗黙の保証も行われるものではありません。販売可能性、特定の目的に対する適合性やサードパーティの特許、著作権、商標またはその他の権利を侵害しないことに対する暗黙の保証も行われません。著作権の所有者は本ソフトウェアまたはマニュアルの使用の結果生じる、直接的、間接的、特殊な、または結果的な損害に対していかなる責任も負いません。著作権所有者の名前および商標は、特別な書面による事前の承諾なしにソフトウェアに関する広告や広報に使用してはなりません。本ソフトウェアおよび関連する資料の著作権の所有権は常に、著作権所有者に帰属するものとします。

米国 Novell, Inc.
1800 South Novell Place
Provo, UT 85606

www.novell.com

LDAP Connect ユーザガイド
2003 年 1 月
000-000000-000

オンラインマニュアル： この製品およびその他の Novell 製品のオンラインマニュアルや更新情報については、
<http://www.novell.com/documentation> を参照してください。

目次

このガイドについて	7
1 exteNd Composer LDAP Connect へようこそ	9
exteNd Composer および Connect の構造について	9
ハブアンドスポークアーキテクチャ	10
exteNd の LDAP Connect について	11
ディレクトリとは	12
ディレクトリへの情報の保存方法	13
LDAP とは	15
LDAP の機能	16
LDAP の動詞	17
DSML とは	18
LDAP Connect で作成できるアプリケーションの種類	19
セキュリティについて	19
詳細情報	20
2 LDAP コンポーネントエディタをお使いになる前に	21
接続リソースについて	22
式駆動型の接続パラメータについて	22
LDAP 接続パラメータ	23
セキュリティの設定	24
LDAP 接続リソースの作成	25
接続に関するトラブルシューティング	28
接続が成功した後の例外	30
サイレントフェイルオーバー	31
接続リソース作成後の編集	32
3 LDAP コンポーネントの作成	33
LDAP アプリケーションモデル	34
LDAP コンポーネントを作成する前に	34
LDAP コンポーネントエディタの特殊な機能	38
LDAP ネイティブ環境ペイン	39
ドラッグアンドドロップ操作	43
特殊なメニューコマンド	44
4 DSML アクション	45
DSML の使用	45
単一の DSML ドキュメントにおける複数の要求	48
Create DSML アクション	49

Add	49
Compare	57
Delete	59
Modify	60
Rename	61
Search	62
Execute DSML アクション	66
LDAP コンポーネントエディタでの他のアクションの使用	67
5 LDAP および DSML の使用	71
DSE クエリの例	71
匿名バインドの接続リソース	72
コンポーネントおよびアクションモデル	72
エラーの処理	77
ECMAScript と LDAP Connect	81
ECMAScript を使用する LDIF の例	81
A LDAP 用語集	87
B LDAP 結果コード	91

このガイドについて

目的

このガイドでは、LDAP Connect を使用する方法について説明します。

対象読者

このガイドは、exteNd Composer を使用して (Web サービスを含む) ディレクトリに対応したサービスやコンポーネントの開発を計画している開発者およびシステムインテグレータを対象にしています。

前提条件

このガイドには、exteNd Composer の動作環境および配備オプションについての予備知識が必要です。また、LDAP (Lightweight Directory Access Protocol) に関する理解も多少必要になります。

追加のドキュメント

Novell exteNd の完全なマニュアルのセットは、Novell マニュアルの Web サイト (<http://www.novell.com/documentation-index/index.jsp>) を参照してください。

1

exteNd Composer LDAP Connect へようこそ

Novell exteNd LDAP Connect ユーザガイドへようこそ。このガイドは、Connect コンポーネントエディタに固有な一部の機能を除き、Composer の標準的なあらゆる機能の使用方法が詳しく説明されている『exteNd Composer ユーザガイド』に付属しています。そのため、『Composer ユーザガイド』をご覧になっていない場合は、このガイドを使用する前に読んで内容を確認してください。

Novell exteNd Composer には、Connect ごとに異なるコンポーネントエディタが用意されています。各コンポーネントエディタの特殊な機能は、これと同じような別のガイドで説明されています。

exteNd Composer を使用しており、コアのエディタ (XML Map コンポーネントエディタなど) に精通している場合は、このガイドにより LDAP コンポーネントエディタについて効率的に学習できます。

exteNd Composer および Connect の構造について

Novell exteNd Composer は、固有な XML 対応の統合アプリケーションを構築 (および配備) するためのツールです。

Composer は 2 つの部分から構成されています。設計側の部分 (このガイドで「Composer」という場合こちらを意味します) は、固有なカスタムアプリケーションを作成するための IDE (統合開発環境) です。ランタイムの部分 (Composer Enterprise Server または簡単に「Composer サーバ」) は、使用するアプリケーションに対するサーバ常駐の実行エンジンです (ただし、サービスを設計、テスト、およびデバッグする場合 Composer サーバにアクセスする必要はありません)。

Composer を使用して構築、配備するサービスは、さまざまな方法でトリガする (呼び出す) ことができます。共通のオプションは、URL 上で「リッスン」するサブレットによりサービスを呼び出すことです (呼び出しおよび配備に関するオプションの詳細については、『exteNd Composer ユーザガイド』および特定のアプリケーションサーバの『Composer サーバガイド』を参照してください)。

このガイドでは、Composer により作成されるサービスで使用できる LDAP のコンポーネントを構築する方法について説明します。通常必要な作業は、次のとおりです。

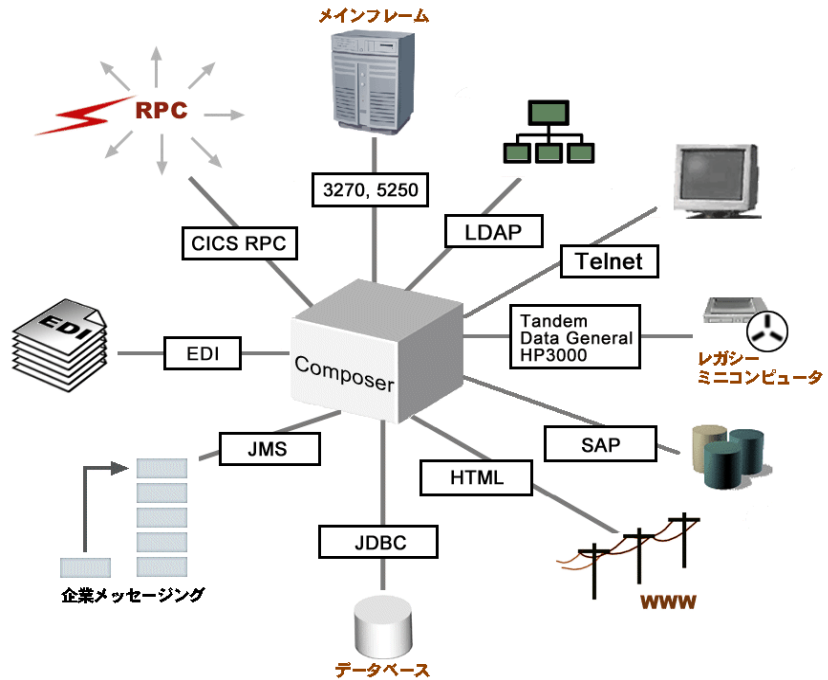
- ◆ プロジェクトを作成する (Composer または Workbench の .spf ファイル)
- ◆ LDAP コンポーネントおよび関連するリソース(接続リソースなど)を作成する
- ◆ オプションで、別の「ビジネス論理」操作を実行する他のコンポーネント (XML Map、JDBC など) を作成する
- ◆ 他のコンポーネントを呼び出すサービス (または「サービスコンポーネント」) を作成する。つまり、サービスにより低いレベルのコンポーネントが「包み込まれ」ます。
- ◆ 必要に応じて JAR、WAR、または EAR 形式でサービスをパッケージ化して配備する

サービスにより、コンポーネントの呼び出す層 (またはコンポーネントのグループ) が指定されます。その場合、アプリケーションサーバ上でパブリック向けのインタフェースを使用せず WSDL で記述された Web サービスとして表示するか、ローカルで実行できます。

ハブアンドスポークアーキテクチャ

Novell exteNd Composer は、単純なハブアンドスポークアーキテクチャに基づいて構築されています。ハブは、XML ドキュメントを受け付けてドキュメントを処理し、該当するビジネスプロセスの特定の要件に従って作成された XML ドキュメントを返す、強力な XML 変換エンジンです。スポーク (つまり *Connect*) は、XML 対応でないデータのソースを「XML 対応にする」アドインモジュールです。つまり、Connect を使用すると、レガシー COBOL-VSAM により管理される情報システム、Telnet またはその他の端末ストリーム、メッセージキュー、EDI など、XML 対応でないデータを XML 形式でキャプチャできます (または、反対にエンドポイントシステムが認識できるように XML データを再パッケージ化できます)。Composer Connect は、業界標準のパーサおよびトランスコーダを使用して、従来のシステムで XML を読み取り、書き込み、および変換できます。

このようなすべての処理は、既存のシステムまたはインフラストラクチャに影響しないように、非強制的または非侵略的な方法で行われます (たとえば、アプリケーションサーバそのものを除き、既存のシステムに新しいソフトウェアをロードする必要はありません)。



Composer は、コアとなる 2 つの Connect、JDBC および LDAP に付属しています (JDBC については個別の『ユーザガイド』で説明されています)。

その他の Connect は、CICS RPC、Telnet、3270、5250、HP3000、Tandem、Data General、JMS、SAP、EDI、および HTML のデータソースに対応しています。これらの別の Connect は、コアの Composer インストールの一部ではなく、個別に購入できる付加価値です。

exteNd の LDAP Connect について

JDBC Connect を使用するとデータベース対応の XML 統合アプリケーションを構築および配備できるように、LDAP Connect を使用すると、ディレクトリ対応であるコンポーネントおよびサービスを構築できます。コンポーネントまたはサービスにより、LDAP クライアントとして動作するパワーがもたらされます。ベンダに関係なく、LDAP プロトコルをサポートする任意のディレクトリに対してクエリを実行します (またはディレクトリの内容を更新します)。

Composer の LDAP Connect に関するパワーおよび柔軟性で重要な点は、ディレクトリの要求および応答をエンコードするための業界標準の XML 文法である DSML (Directory Services Markup Language) を処理する機能です (後の詳しい説明を参照してください)。DSML は XML の方言のようなものであるため、人間が認識でき、マシンにより解析可能で、移動可能、ファイアウォールに対応しやすいなど XML の長所をすべて持ち合わせています。

注記: LDAP Connect を使用する場合、実際の DSML ドキュメントを作成したり、用意しておく必要はありません。Composer により必要な DSML が即時に作成されます。

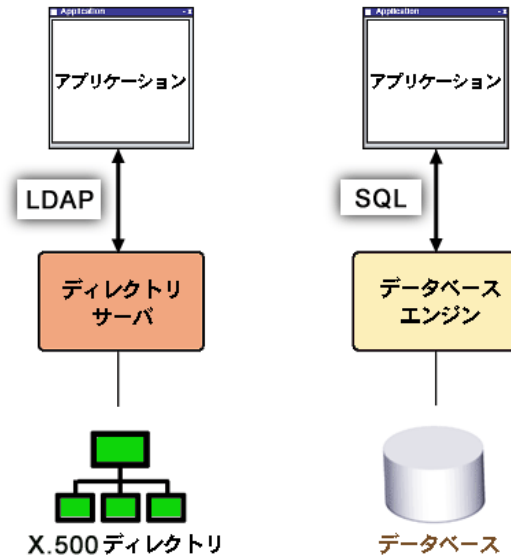
DSML については、後で詳しく説明します。

ディレクトリとは

ディレクトリとは、構造化されたデータストアです。したがって、リレーショナルデータベース (おそらく構造化されたデータストアのうち最も良く知られているタイプです) の特徴を多く持ちます。ただし、ディレクトリはさまざまな重要な点で従来のデータベースとは異なります。

- ◆ データベースが潜在的に不安定な大容量のデータ (頻繁に更新する必要があるようなデータ) に対応できるように設計されているのに対し、ディレクトリは頻繁に更新する必要がない比較的小規模で、高度に分類されたデータのまとまりを処理する場合に適しています。
- ◆ ディレクトリを使用すると、管理者は柔軟な命名や格納に関する規則に基づいて構築される垂直的なオブジェクトの階層でデータを統合することができます。
- ◆ ディレクトリ内の各データ要素 (またはエントリ) は、固有な ID (「識別名」として知られています) により直接アドレス可能です。ディレクトリエントリのアドレスは、URL と同じように区分的に構成される (および分解される) ため、相対的な修飾名および完全な修飾名が存在します。データベースの場合、データ要素は直接アドレス可能ではありません。

データベースでは、ディレクトリ内のデータをクエリしたり、書き込み (更新) または削除したりできます。データベースは、何らかの種類の SQL (Structured Query Language) によりクエリを実行できます。ディレクトリの場合、クエリは RFC 2251 および 2254 で説明されている構文に従います。LDAP のクエリ「言語」は、効率的に標準化されています。



前の図は、アプリケーションが基本的な2つのタイプの構造化されたデータストア（ディレクトリとデータベース）と通信の様子を示しています。LDAPを利用して、アプリケーションはディレクトリをクエリまたは更新します。SQLおよび適当なドライバを利用して、アプリケーションはデータベース内のデータにアクセスします。

ディレクトリへの情報の保存方法

ディレクトリにより、データはエン트리として保存されます。エントリの集合は通常「オブジェクト」と呼ばれます。オブジェクトには、エン트리としてその他のオブジェクトを含めることができます。

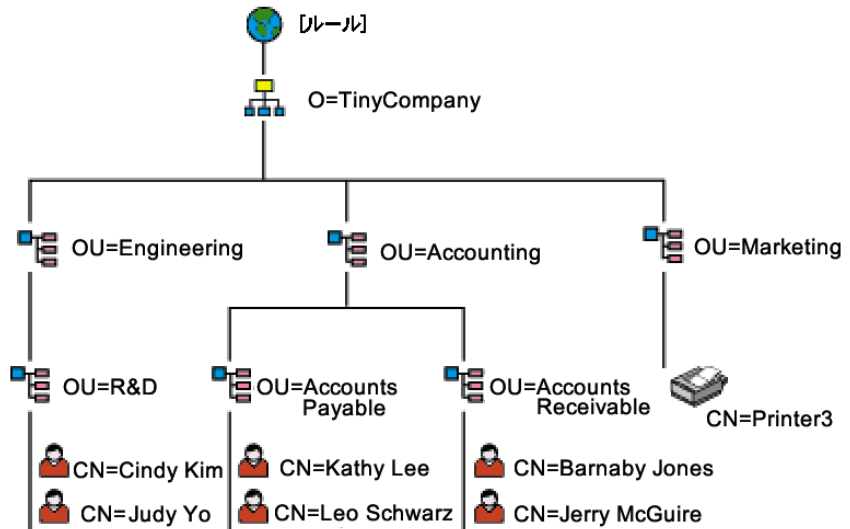
ディレクトリオブジェクト内のデータは、属性 - 値の対で構成されています。属性の一部は「単一値」と呼ばれるもので、その他は「複数值」です。たとえば、複数の電話番号を持っている人がいます。この場合、その人物の電話番号が保存される属性は、おそらく複数值の属性になります。このことは、ディレクトリの「スキーマ」で指定されます（ディレクトリスキーマをXMLスキーマと混同しないでください。この2つはまったく別のものです）。

エントリに他のエントリが含まれている場合、含む方のエントリを「コンテナオブジェクト」と呼びます。

含まれる方のエントリは、コンテナの「サブオーディネートエントリ」と呼ばれます。反対に、コンテナは含まれるエントリの「スーパーリア」と呼ばれます。

注記： DOM ノードがペアレントおよびチャイルドのいずれにもなることができるように、エントリはコンテナ (スーパーリア) および別のエントリのサブオーディネートのいずれにもなることができます。

エントリをネストできることがディレクトリの重要な特徴で、ツリー構造のようになります (実際、ディレクトリを表す場合に DIT またはディレクトリ情報ツリーという用語が登場します)。ツリー構造は、XML 表現に好適です。



前のグラフィックは、小規模な企業で使用されるようなディレクトリ構造を示しています。この例では、TinyCompany のトップレベルのノードが、TinyCompany という組織 (O) 属性になっています。TinyCompany の下のレベルには、エンジニアリング、会計、およびマーケティング部門のコンテナが存在します (OU または組織ユニット属性というラベルが付いています)。会計部門には、さらにその下に 2 つの組織があります (支払勘定と受取勘定)。これまでのエントリはすべてコンテナです。リーフノードレベルで、コンテナの下のレベルに存在するさまざまな担当者の CN (共通名) エントリに到達します。現在、マーケティング部門に担当者はいませんが、プリンタが存在していることに注意してください。

ツリー内の任意の項目を名前空間の連結 (またはフェデレーション) によりアドレス指定できます。Judy Yo の場合、次のように明確にアドレス指定できます。

```
cn=Judy Yo,ou=R&D,ou=Engineering,o=TinyCompany
```

この「完全修飾された」名前は、「識別名」と呼ばれます（通常、「DN」または「dn」と表示されます）。識別名は、完全修飾されたパスまたは URI と類似しています。

注記： DN では、順序が重要です。URL またはファイルシステムの命名スキームと異なり、DN 内部の（左から右への）順序は組織の下位レベルから上位レベルへ（チャイルドからペARENTへ）、常にツリー構造で高いレベルに向かいます。

順序が重要であるため、前の DN を次のように書き換えるのは間違いです。

```
cn=Judy Yo,ou=Engineering,ou=R&D,o=TinyCompany
```

R&D はエンジニアリング部門の下に属するため、この DN では階層関係が間違っ
て記述されています。ここで表示されている DN では、前の図で示されているオブ
ジェクト階層が正しく解決されません。

命名規則およびその他の規則の詳細については、後の章で説明します。

LDAP とは

LDAP (Lightweight Directory Access Protocol) は、ディレクトリと通信するための
メッセージングプロトコルです。より詳細な DAP (Directory Access Protocol) で指
定された機能性のコンパクトなサブセットが実装されます。完全な DAP は、X.500
ディレクトリサーバと通信するための機能豊富、広範囲で、綿密な（重厚な）プ
ロトコルです。対照的に LDAP は機能が制限されており、初心者学習に比較的
適した軽量なプロトコルです。あまり一般的でない OSI ネットワークプロトコル
スタックで DAP が使用されるのとは対照的に、LDAP について、メッセージング
のために通常の TCP/IP を基礎する DAP の複雑さが軽減されたバージョンだと考
えることができます。

多くの環境で、LDAP は RFC 3377 で説明されている単なるネットワークプロトコ
ル以上の意味を持つようになりました。LDAP が話題になる場合、その対象とし
て次の内容が考えられます。

- ◆ LDAP 「機能モデル」：ディレクトリで実行できる操作の種類
- ◆ LDAP 「名前空間モデル」：名前によりデータのグループを判断または認識する
方法
- ◆ LDAP 「データモデル」：ディレクトリで保存できる情報の種類およびデータ
を整理する規則の定義

このような概念の多くは ITU (国際電気通信連合) の T 勧告 X.500、1993 年の「The Directory: Overview of Concepts, Models and Service」(ディレクトリ: 概念、モデル、およびサービスの概要) から派生しています。LDAP は核として (HTTP のような) ネットワークプロトコルを持ちながら、セマンティックは ITU の X.500、X.501、および X.511 標準により定義されている概念的なフレームワークに堅くバインドしています。

LDAP を使用すると、ディレクトリとの通信を確立 (または「バインド」)、ディレクトリ内の情報へのアクセス (読み取り)、ディレクトリの更新 (情報の書き込み) が可能になります。

LDAP バージョン 3 の仕様については、<http://www.ietf.org/rfc/rfc2251.txt> を参照してください。

LDAP のその他の情報については、<http://www.openldap.org> を参照してください。

LDAP の機能

LDAP はディレクトリと通信するためのプロトコルです。他のネットワークプロトコルと同様、LDAP には固有の「ハンドシェイク」規則およびセマンティックを含むキーワードのポキャブラリがあります。さまざまなタイプの操作を記述するために、合計で 20 の動詞があります (次の節を参照してください)。

- ◆ **bind - bind** (バインド) 操作により、クライアントアプリケーションとディレクトリサーバの間に LDAP セッションが確立されます (この操作により、クライアントアプリケーションはサーバに認証情報を渡すこともできます)。
- ◆ **unbind - unbind** (バインド解除) 操作により、LDAP セッションを終了し、「接続」を破棄できることがサーバに伝達されます。
- ◆ **search - search** (検索) 操作を使用すると、LDAP 対応のクライアントアプリケーションはディレクトリサーバにより提供される検索サービスを利用できます。クライアントアプリケーションの検索要求におけるパラメータに応じて、サーバは単一エントリから、ディレクトリツリー上にある特定エントリの下レベルのエントリすべてから、またはディレクトリツリーの分岐全体から情報を返します。

注記: クライアントアプリケーションは特定の検索について範囲を定義できるだけでなく、LDAP 3 のパラメータを使用すると、返すエントリの数の制限、検索を実行する時間の制限など検索フィルタを定義することもできます。

- ◆ **modify - modify** (変更) 操作を使用すると、クライアントは特定のエントリに対する属性値を変更できます。Modify 操作のパラメータにより、クライアントアプリケーションは属性値の追加、属性値の削除、および属性値の変更ができます。たとえば、クライアントアプリケーションにより追加、削除、または置換操作を要求できます。

- ◆ *Add* - 特定のユーザエントリの `telephoneNumber` 属性に電話番号を追加します。
- ◆ *Delete* - `telephoneNumber` 属性から電話番号を削除します。
- ◆ *Replace* - 新しい電話番号で、既存の電話番号を置換します。
- ◆ *Modify Distinguished Name (DN) - Modify-DN* (DN 変更) 操作を使用すると、クライアントアプリケーションは識別名の一番左にある要素を変更することにより、そのエントリの識別名を変更できます。
- ◆ **Compare - Compare** (比較) 操作を使用すると、クライアントは規定された属性値と特定のエントリにある属性値を比較します。たとえば、**Compare** 操作を使用して特定のユーザのパスワードを検証できます。
- ◆ **Abandon - Abandon** (破棄) 操作を使用すると、クライアントアプリケーションは完了していない操作を中止できます。

LDAP 3 では、「間接化」機能 (あるディレクトリにより、クライアントアプリケーションで別の LDAP 3 ディレクトリを参照させる方法) も定義できます。たとえば、クライアントアプリケーションが特定のエントリに関する情報を LDAP Directory A から要求したものの、Directory A で要求したエントリが見つからなかったとします。Directory A が Directory B に該当する情報が含まれていると認識していた場合、A はクライアントに対し、B を参照して検索を続行するように指定します (参照範囲が拡大する場合があります)。

ホストによって独自の参照のリダイレクトスキームが実装される場合、この処理は「チェーン」と呼ばれます。LDAP では、チェーンはサーバによって完全に制御されるため、クライアントはチェーンに関する知識はなく、また (一般的に) チェーンを制御する能力もありません。

一方で、参照のリダイレクトはクライアントの制御下にあります。クライアントは、受信されるたびに各参照に対して動作を行うかどうか、また各「ホップ」に対してどのようなセキュリティを使用するかを決定する必要があります。

注記: exteNd Composer LDAP Connect の現在のバージョンでは、参照のリダイレクトのネイティブサポートは提供されていません。ただし、この処理を実行するために、通常のループ構成および動的な接続リソースパラメータの値を使用して独自のアクションモデル論理を作成できます。

LDAP の動詞

前の説明では、LDAP における操作の「非プログラマ」用語を説明しました。下位のレベルでは、LDAP の操作は LDAP の「動詞」により指定されます。

LDAP の動詞 (バージョン 3 の仕様に固有なものを含む) の現在のリストは次のとおりです。

- ◆ `BindRequest`

- ◆ BindResponse
- ◆ UnbindRequest
- ◆ SearchRequest
- ◆ SearchResponse (バージョン 2 のみ、バージョン 3 にはなし)
- ◆ SearchResultEntry (バージョン 3)
- ◆ SearchResultDone (バージョン 3)
- ◆ SearchResultReference (バージョン 3)
- ◆ ModifyRequest
- ◆ ModifyResponse
- ◆ AddRequest
- ◆ AddResponse
- ◆ DelRequest
- ◆ DelResponse
- ◆ ModifyDNRequest
- ◆ ModifyDNResponse
- ◆ CompareRequest
- ◆ CompareResponse
- ◆ AbandonRequest
- ◆ ExtendedRequest (バージョン 3)
- ◆ ExtendedResponse (バージョン 3)

これらの動詞は、単に LDAP で実行可能な操作の種類を示す目的で表示したものです。Composer LDAP Connect を使用するために、動詞を直接扱う方法を理解する必要はありません。

これらのコマンドの使用法および意味の詳細については、<http://developer.novell.com/ndk/> の開発者用ドキュメントを参照してください。

DSML とは

DSML (Directory Services Markup Language) を使用すると、ディレクトリの情報またはディレクトリのクエリ、あるいはその両方を XML ドキュメントとして表現できます。

注記： LDAP の経験がある場合、DSML を LDIF ファイルの XML パージョンと考えることができます (LDIF は LDAP Data Interchange Format の略で、ディレクトリクエリおよび LDAP クエリを指定するためのテキスト形式のファイルです。LDIF は RFC 2849 で説明されています)。

DSML の仕様 (<http://www.oasis-open.org> を参照) は、SOAP (Simple Object Access Protocol) のようなファイアウォールに対応しやすいメカニズムを使用して、XML ベースの企業アプリケーションでディレクトリに保存されているリソース情報を簡単に利用できるように作成されました。

DSML により、XML とディレクトリをともに使用できるようになりました。XML ベースのアプリケーションがディレクトリベースの情報にアクセスするためのメカニズムが一般化されたのです。DSML の目的は、成長を続ける XML ベースの企業アプリケーションをディレクトリ対応にすることです。

注記: Composer LDAP Connect を使用するために、DSML を理解する必要はありません。LDAP Connect の機能の多くはウィザードにより実行されます。ウィザードのダイアログボックスへの入力に応じて、Composer LDAP Connect により必要な DSML ドキュメントまたは DOM が動的に作成されます。

LDAP Connect で作成できるアプリケーションの種類

LDAP Connect および Composer を利用することにより、XML 統合アプリケーション (Web サービスまたはローカルコンテキストのプライベートアプリケーションいずれの場合でも可) に「ディレクトリ対応性」を構築できます。LDAP 対応アプリケーションは交換形式として XML を使用し、LDAP によりアクセス可能な任意のデータストアでデータを出し入れします (DSML は、実際に使用される XML の方言のようなものです)。この動作について、DSML の知識は必要ありません。たとえば、企業のディレクトリから電話番号、電子メールアドレス、および従業員の役職を取り出すコンポーネント (おそらく大容量の Web サービスの一部) を記述できます。アプリケーションで取り出す情報が 2 つ以上のディレクトリに存在する場合、個別のディレクトリから情報をマージした後、ユーザに表示したり、アプリケーションの別のコンポーネントに渡したりできます。

セキュリティについて

LDAP アプリケーションでは通常、LDAP エンドポイントの認証および LDAP セッションデータの暗号化に SSL および TLS (Transport Layer Security) が使用されます。

TLS はさまざまなトランスポート層セキュリティの一般的なラップです。TLS セッションへの参加者はセッションの開始時に、使用可能な複数のセキュリティメカニズムのいずれか (通常は SSL 技術) を使用することに同意し、そのメカニズムを使用して「セキュリティ保護されたセッション」を実行します。

TLS が有効な場合、すべての通信は暗号化され、パスワード (およびデータ) がそのまま送信されることはありません。

ホスト認証も TLS のコンポーネントです。デフォルトでは、ホストがその X.509 証明書の情報をクライアントに送信し、クライアントはそのホストが正しい LDAP サーバであることを確認することによりホストを検証 (認証) します。

注記: 認証に使用されるその他のハンドシェイクとして、クライアントによりホストの認証をせずホストがクライアントを認証する方法、および双方向で相互認証を行う方法がありますが、exteNd Composer LDAP Connect では現在、サーバがその証明書の情報をクライアントに送信するというもっとも一般的なハンドシェイクシナリオのみサポートしています。

LDAP Connect のセキュリティの詳細については、後で説明します (「LDAP コンポーネントエディタをお使いになる前に」)。

TLS (Transport Layer Security) の詳細については、RFC 2246 (<http://www.faqs.org/rfcs/rfc2246.html>) を参照してください。

詳細情報

Web には、LDAP およびディレクトリに関する優れたリソースが数多くあります。LDAP に関連する多くの記事、仕様、および開発者向けリソースへのリンクが表示されている <http://developer.novell.com/edirectory/ndsldap.htm> から始めることをお勧めします。

LDAP およびその他のリソースに関する記述がある RFC へのリンクは、<http://nldap.com/nldap/> を参照してください。RFC の一部として、次のようなものがあります。

- ◆ RFC 2251 - LDAP (バージョン 3)
- ◆ RFC 2252 - LDAP (バージョン 3): 属性構文の定義
- ◆ RFC 2253 - LDAP (バージョン 3): 識別名の UTF-8 文字列表現
- ◆ RFC 2254 - LDAP 検索フィルタの文字列表現
- ◆ RFC 2255 - LDAP URL 形式
- ◆ RFC 2256 - LDAP バージョン 3 で使用される X.500(96) ユーザスキーマの要約

公開されている LDAP テストディレクトリは、<http://nldap.com> で Novell により管理されています。LDAP Connect のコンポーネントおよびサービスをテストするローカル LDAP サーバがない場合、テストの目的で、プライベートコンテナをセットアップするためにこれを使用することができます。

X.500 ディレクトリ標準 (および DAP) に関する情報は、ITU (国際電気通信連合) の Web サイト <http://www.itu.int> を参照してください (注記: ITU 標準をダウンロードするには、購入する必要があります)。

2

LDAP コンポーネントエディタをお使いになる前に

LDAP Connect を使用して Composer コンポーネントを作成する場合の手順は、他のあらゆるコンポーネントを作成する場合と基本的に同じです。その手順は次のとおりです。

- 1 コンポーネントに必要な XML テンプレートを決定します。
- 2 接続リソースを作成して、コンポーネントが LDAP ホストにバインドできるようにします (この手順は、後で詳しく説明します)。
- 3 新しいコンポーネントを作成します。
- 4 コンポーネントに固有なアクションを作成します。
- 5 アニメーションモードでコンポーネントを実行して、アクションモデルをテストします。
- 6 アニメーション中に見つかった問題を解決します。
- 7 作業を保存します。
- 8 コンポーネントを呼び出す配備可能なサービスを作成します (コンポーネントを直接配備することはできません。Composer プロジェクトでは、サービス単位で配備します)。
- 9 サービスをステージング領域またはアプリケーションサーバ環境 (Novell exteNd Application Server、IBM WebSphere、または BEA Weblogic) に配備します。

次の節では、手順 2 (接続リソースの作成) について詳しく取り上げます。LDAP に固有なアクションの使用方法に関する詳細は、後の「LDAP コンポーネントの作成」の説明を参照してください。

接続リソースについて

動作する LDAP コンポーネントを作成する前に、そのための接続リソースを作成する必要があります (LDAP コンポーネントを作成するときリソースリストに LDAP 接続リソースがない場合、ダイアログボックスにプロンプトが表示され、Connection Resource ウィザードに移動できます)。

LDAP 接続リソースには、コンポーネントが LDAP ホスト (またはディレクトリサーバ) に接続する場合に必要な情報が含まれています。接続を確立するそれぞれのホストに対して、または指定のホストに使用するアカウント情報の固有な各セット (またはタイムアウト設定) に対して、通常は個別のリソースを作成します。

注記: リソースで「式駆動型の接続パラメータ」を指定する場合、次に説明するように、指定の接続リソースに関連付けられているさまざまなパラメータは遅延バインディングになる場合があります。

式駆動型の接続パラメータについて

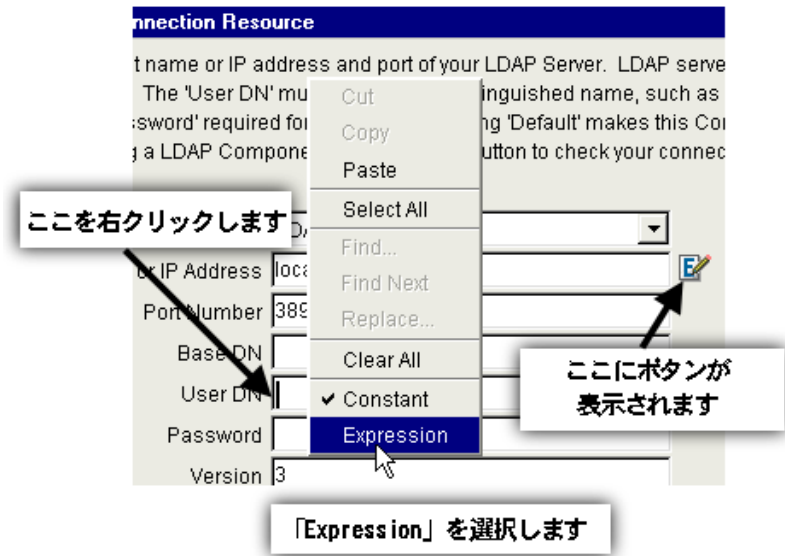
接続パラメータの値は、定数または式のいずれかにより指定できます。

定数ベースのパラメータでは、接続が使用されるたびに Connection ウィザードで入力するそのままの値が使用されます。式ベースのパラメータでは、ランタイム時に評価されるプログラム (ECMAScript) の式により値が設定されます。後者により提供されるパラメータの「遅延バインディング」により、接続リソースが使用されるたびに別のユーザ DN またはパスワードを指定できます。つまり、プログラムにより接続パラメータをランタイム時に選択できます。

ファイルまたはデータベースからランタイム時にログインアカウント情報を引き出すとします。接続リソースをセットアップして、ECMAScript File I/O 拡張機能または Java により直接ファイルまたはデータベースから必要な情報を検索する式を使用できます。式を使用すると、接続の動作が柔軟になり、ランタイム条件に応じて動的に変更することができます。

➤ 定数駆動型から式駆動型にパラメータを切り替える

- 1 変更するパラメータフィールド内でマウスを右クリックします。
- 2 表示されるコンテキストメニューから [Expression] を選択します。パラメータフィールドの右側に、小さなエディタボタン (アイコン) が表示されます。



- 3 ボタンをクリックしてから、[Expression Builder] ダイアログボックスを使用してランタイム時に有効なパラメータ値を評価する ECMAScript 式を作成します。

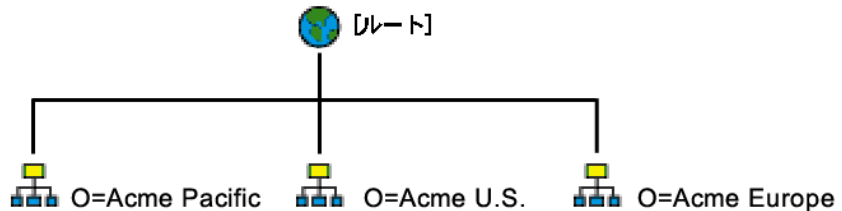
LDAP 接続パラメータ

LDAP 接続を作成するには、最低でもセッションが発生するサーバの IP アドレスおよびポート番号を指定する必要があります。この 2 つの項目を指定すれば、匿名バインドを許可するホストに匿名でバインドできます(LDAPにおける匿名バインドは、匿名の FTP セッションと類似しています)。IP アドレスは、設計時には「127.0.0.1」または「localhost」のようになります (ただし、ランタイム時には変わることが想定されます。おそらく式駆動型の IP アドレスパラメータにより切り替えられます)。ポート番号は通常、セキュリティ保護されていないセッションでは 389 で、SSL セッションでは 636 です。この値については、ほとんどのディレクトリサーバでこのように設定されますが、実際には有効な任意の値を指定できます。

匿名バインドの場合、ユーザ DN (ユーザ識別名) およびパスワードを指定する必要があります。ユーザ DN とは、クライアントがバインドするディレクトリオブジェクトの名前です (RFC 2251 より)。これは通常、「cn=John Doe, ou=Mailroom, o=Rising Star Industries」のようにディレクトリ内の人物またはエンティティの識別名になります。

指定のサーバに複数のツリーが存在する場合、サーバはクライアントがバインドするツリー (より正確にはサブツリー) を認識する必要があります。この情報は、ベース DN の値で指定されます。

たとえば、架空の国際企業体 Acme, Inc. で使用されるディレクトリサーバを考えます。Acme ディレクトリサーバには上位レベルに 3 つのコンテナオブジェクト (3 つの「ツリー」) があり、次の図のようにそれぞれ Acme Pacific、Acme U.S.、および Acme Europe の完全なディレクトリ構造であるとしています。



LDAP クライアントが Acme ディレクトリにバインドする場合、サーバではどのツリー (Pacific、U.S.、または Europe) にクライアントがバインドする必要があるのか認識する必要があります。この例で、スペインにいる Acme 人事部のマネージャが Acme Europe のツリーにバインドしたい場合、ベース DN を「o=Acme Europe」と指定します。

注記： LDAP の場合、ルートオブジェクトは実際にはツリー内のエントリではなく、直接アドレス可能ではありません。ルートオブジェクトは、より正式にはルート DSE または DSA に固有なエントリとして知られています (DSA は *directory system agent* の略で、ディレクトリサーバに対する X.500 の用語です)。DSE に対してディレクトリサーバに関する「メタ」情報を検索できます (LDAP バージョン 3)。

セキュリティの設定

Composer の LDAP Connect では、SSL により暗号化されたセッション (および X.509 デジタル証明書技術を使用したサーバ認証) がサポートされています。

暗号化および認証を有効にするには、接続セットアップダイアログボックスで [TLS (Transport Layer Security)] チェックボックスをオンにする必要があります。ほとんどの場合、(同じダイアログボックスで) ポート番号を 636 に設定します。LDAP サーバでは通常、このポートで暗号化されたセッションが実行されるためです。

注記： [TLS] チェックボックスをオンにしても、ダイアログボックスに表示されるポート番号が自動的に 636 に変更されるわけではありません。ポート番号を手動で入力する必要があります (同様に、ポートの値として 636 を入力しても、[TLS] チェックボックスは自動的にオンになりません)。

[TLS] チェックボックスがオンの場合、接続を通じて移動するすべての情報が暗号化され、いかなる情報もそのまま送信されることはありません。また、ホスト (またはディレクトリサーバ) に対して証明書に関する情報が要求されます。ホストは、その X.509 証明書情報を返します。Composer は、xcrootca.jar ファイルに保存されている認証局のデータに対し証明書情報を確認します (xcrootca.jar ファイルは Composer 付属の Java アーカイブで、業界標準のさまざまな証明書発行元に関する認証局情報が含まれています。詳細については『Composer ユーザガイド』を参照してください)。

注記: xcrootca.jar は、Composer インストールの lib フォルダにあります。WinZip を使用してファイルを開くと、確認したり、新しい証明書を追加したりできます。アプリケーションサーバのランタイム環境に対して X.509 証明書およびアプリケーションに必要なその他のセキュリティリソースを追加する方法を学習するには、ご使用のアプリケーションサーバのマニュアルを参照してください。

Composer の LDAP Connect でサポートされている SSL3 セキュリティメカニズムは、「全か無か」です (動的に切り替えることはできません)。つまり、クリアテキストモードで LDAP セッションを開始し、(同じ接続リソースを使用しながら同じセッションの一部として) 即時に「セキュリティ保護された」モードになる機能は、LDAP Connect のこのバージョンではサポートされていません。

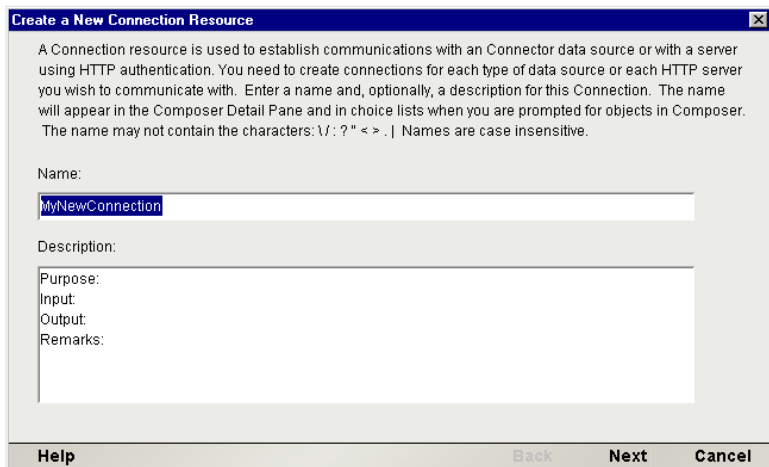
LDAP 接続リソースの作成

LDAP 接続リソースを作成するプロセスは単純です。

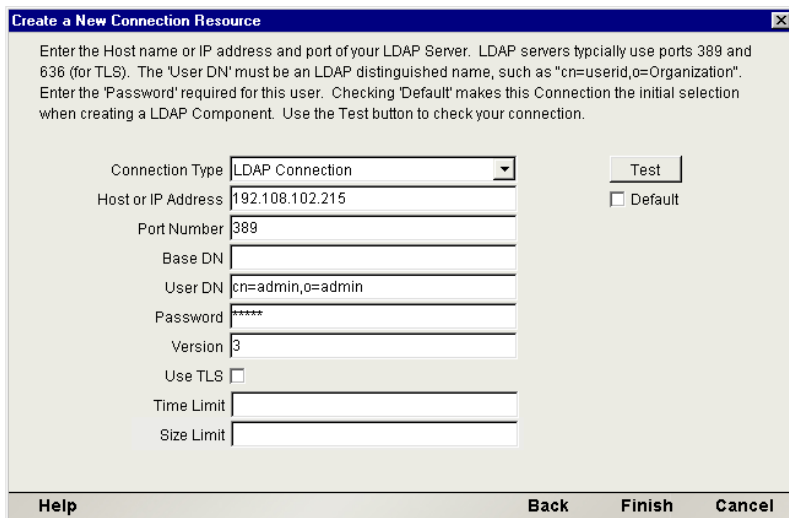
➤ LDAP 接続リソースを作成する

- 1 Composer のメインメニューバーから、[File]、[New xObject]、[Resource]、[Connection] の順に選択します。

「Create a New Connection Resource」ウィザードが表示されます。



- 2 [Name] に、接続オブジェクトの名前を入力します (これは、Connection カテゴリでリソースを参照すると、後で Composer のメインウィンドウにあるナビゲータペインに表示されます)。
- 3 (オプション) ダイアログボックスの [Description] 領域に、テキストを入力します。
- 4 [Next] をクリックします。新しいダイアログボックスが表示されます。



- 5 [Connection Type] プルダウンメニューから [LDAP Connection] を選択します。

- 6 [Host or IP Address] の横に、接続先ディレクトリサーバの名前または IP アドレスを入力します (テストの場合、通常 localhost です)。

注記: このパラメータおよびこのダイアログボックスにおける後続のすべてのテキストフィールドは、ECMAScript 式を使用して動的に設定できます。この章の前半の「式駆動型の接続パラメータについて」を参照してください。

- 7 接続で、デフォルトの 389 以外のポート番号を使用する場合、[Port Number] の横に適当な値を入力します (接続が SSL を使用した TLS 接続で、ここに入力する値がわからない場合は、LDAP-over-SSL の標準ポートである 636 を入力します)。

ヒント: localhost:389 のようにコロンを使用すると、(前の手順の) IP アドレスの一部としてポート番号を指定できます。1 つにまとめた「IP アドレス: ポート番号」の形式を使用する場合、[Port Number] フィールドには 0 (ゼロ) を入力します。

- 8 該当する場合は、[Base DN] に有効なベース DN (ツリー名) を入力します。この値は、t=DEVNET-TREE のようになります。

- 9 [User DN] に、有効なユーザ DN (ユーザ識別名) を入力します。識別名については、前の説明を参照してください。

注記: 匿名 (パスワードなし) のバインドを実行している場合は、任意の値を入力できます。

- 10 匿名バインドでは、パスワードは必要ありません。その他のすべての場合には、[Password] に指定のユーザ DN に対して有効なパスワードを入力する必要があります。

注意: 接続で TLS が有効になっていない場合、このパスワードはそのまま送信されます。

- 11 LDAP ホストで別の値が必要でない限り、[Version] (LDAP のバージョン) でデフォルトの値「3」を受け入れます。

- 12 セキュリティ保護されている接続 (SSL3) の場合は、[TLS] チェックボックスをオンにします。X.509 証明書ベースの認証の場合、(Composer の lib ディレクトリ) xrootca.jar ファイルに適切な認証局のエントリを持つ必要があります。また、サーバに正しい証明書の設定が存在していることが必要です。詳細については、ご使用のアプリケーションサーバのマニュアルを参照してください。

- 13 オプションで、[Time Limit] に、コンポーネントがサーバとの接続を確立する際に待機する最大時間の値 (ミリ秒) を入力します。デフォルト値は無制限です。

- 14** オプションで、[**Size Limit**] に、設計時にディレクトリのツリービューを表示するために受け入れるツリーエントリ (ノード) の最大数を示す整数値を入力します。デフォルト値は 1000 です。これは、Composer のツリーブラウザでは、指定のペアレントノード (エントリ) の下に、最大 1000 のチャイルドノードを表示できることを示します。

非常に大きな数字を入力すると、次の 2 つの状況が発生する可能性があります。

- ◆ **Composer** は要求されたすべてのエントリを識別して表示しようとするため、パフォーマンスに影響がある可能性があります。
- ◆ リモートホストで、処理できるエントリ数の事前に設定された制限に達し、エラーが返される可能性があります (この場合、Composer は対象のノードの下にあるエントリを表示できなくなります) 。

大多数のチャイルドノードを含むツリーノードを参照する必要があると確実にわかっていない限り、デフォルト値の 1000 を受け入れることをお勧めします。

注記： これは、設計時のみにおける注意事項です。ランタイム時に、サーバではこの設定によって何の動作も行われません。

- 15** [**Test**] をクリックして、リソースで定義された接続が実際に動作するかどうかを確認します。数秒後、成功または失敗を示すダイアログボックスが表示されます (接続できない場合、テストが「タイムアウト」になるまでに 30 秒以上かかることがあります。待機時間を短縮するには、前の手順の説明に従って [**Time Limit**] の値を設定してください) 。

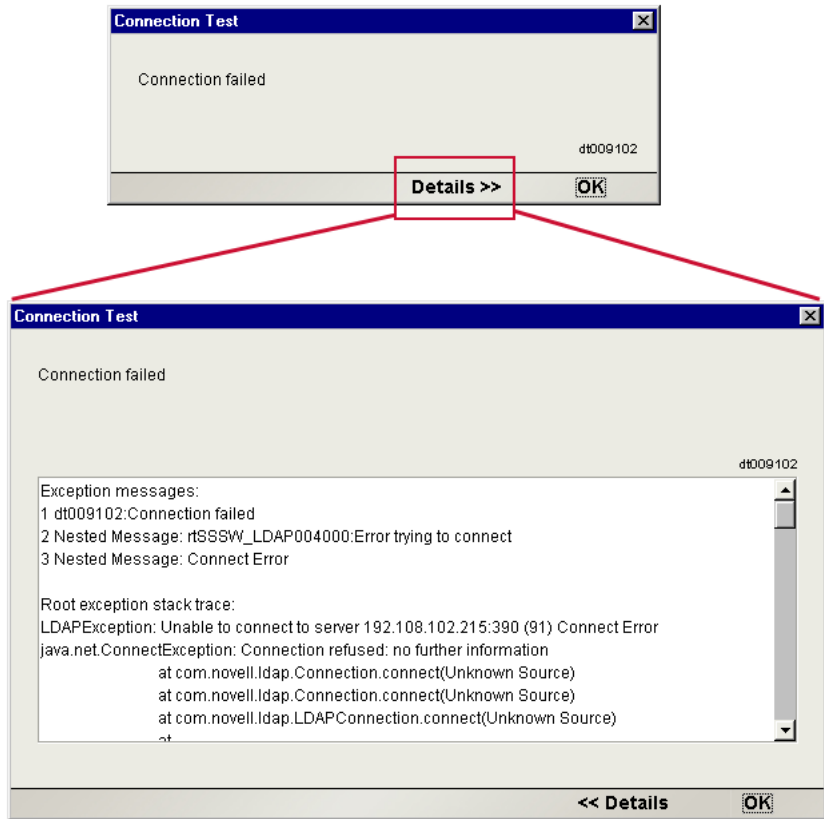
注記： 接続が失敗した場合でも、リソースの編集を続けることができます。または、リソースを保存し、後で編集することもできます。

- 16** オプションとして、新しい LDAP コンポーネントを作成するたびにデフォルトの接続としてこの接続を使用する場合は、[**テスト**] ボタンの下にある [**Default**] チェックボックスをオンにします (この設定はいつでも解除できます) 。

- 17** [**Finish**] をクリックします。新しく作成されたリソース接続オブジェクトが、Composer 接続リソースの詳細ペインに表示されます。

接続に関するトラブルシューティング

[**Test**] ボタンを押した後、(「**Connected Successfully**」というメッセージではなく) エラーダイアログが表示された場合、[**Details**] ボタンをクリックすると例外に関する完全なスタックトレースが表示されます。次の図を参照してください。



スタックトレースのメッセージにより、問題の原因を特定するための便利なヒントが得られることがよくあります。前の図の例では、接続が拒否されています。詳しく確認すると、IP アドレスの最後、コロンの上に 390 というポート番号が指定されています。このサーバに対する正しいポート番号は 389 なので、接続が取得されず LDAPException がスローされました。

問題が発生した場合は、次のような点に注意します。

- ◆ **匿名バインド**：匿名バインドを確立できない場合、「匿名 FTP」セッションを許可するために FTP サーバを設定できないのと同様に、匿名バインドを許可するためにホストサーバを設定することはできません。サーバで匿名バインドを実行できない場合、何らかの方法でサーバの認証を行う必要があります（ポート 389 に対して簡単なパスワードとユーザ DN、ポート 636 の場合は完全な SSL など）。

- ◆ **ECMAScript式:** 式駆動型のパラメータを使用する場合(前の「式駆動型の接続パラメータについて」を参照してください)、リテラル文字列の値を引用符で囲む必要があります。反対に、式駆動型のパラメータを使用しない場合は、パラメータの値を引用符で囲まないでください。
- ◆ **TLS:** セキュリティ保護された接続が必要ない場合、このチェックボックスがオフになっていることを確認します。オンの場合は、「セキュアポート」(636 など) が指定されていることを確認してください。また、セキュリティ保護されている接続を使用する場合は、認証局に関する **.jar** ファイルに適切なエントリが含まれており、正しくインストールされていることを確認します。Composer インストールで、**xcrootca.jar** という名前のファイルを検索します。このファイルに対する完全修飾されたパスは、設計時のインストールだけでなく、ランタイム時(またはサーバ)のインストールでも **xconfig.xml** ファイルの <XCERTFILE> 要素で指定する必要があります。また、追加の手順として、ランタイム時にアプリケーションサーバによりこのファイルが見つかることの確認が必要になる場合があります。X.509 証明書および認証局のファイルをセットアップする方法の詳細については、ご使用のアプリケーションサーバのマニュアルを参照してください。

接続が成功した後の例外

接続のテストを行う場合、「Connected Successfully」というメッセージの後にエラーメッセージが表示されることがあります。成功のメッセージは、指定したホスト IP アドレスおよびポート番号が正しく、ホストとの接続が確立されたことを表します。続いて例外メッセージが表示された場合、タイムアウト、不明なベース DN など接続後に問題が発生したことを意味します。

注記： LDAP では、接続とバインドはわずかに異なります。接続が構築されたということは、ディレクトリ内の特定のターゲットオブジェクトを参照することなく、単にホストとクライアントの間で TCP/IP により LDAP セッションを開始できた、という意味です。セッションアクセスのコンテキストで、ユーザ DN およびアカウント情報の指定の組み合わせとサーバにおける指定のオブジェクトを関連付けについて、ホストが適当だと判断した場合、「バインド」が発生します。

例を使用すると、次のようになります。10歳の Danny が、隣の家のドアをノックするとします。大人がドアを開けます。Danny はその大人に尋ねます。「Tommy は遊びに行ける？」この例で、Danny を LDAP クライアント、大人を LDAP ホストに例えることができます。ドアが開いたとき、接続が確立されたこととなります。大人(サーバ)は、Danny(クライアント)に Tommy(ターゲットオブジェクト)へのアクセスを許可するかどうか判断する必要があります。大人が Danny と顔見知り(Danny を何らかの方法で認証できる)、Danny に Tommy へのアクセス権がある場合(1日のうちの適当な時間である、Tommy が宿題を終えているなど)、2人の子供は遊びに行くことができます(「バインド」できます)。反対に、その大人が Danny を見たことがない、または信用できない場合、あるいはアクセスルールにより Tommy が遊びに行けない場合、ドアが開いて会話が開始されたとしても Danny は帰ることになります。

サイレントフェイルオーバー

ディレクトリは、重要なシステムの中核であることがよくあります。したがって、(いかに短時間であっても)システム停止のような事態や、通常のメンテナンスに関連する中断が大きく影響します。そのため、クライアントがサーバ A に接続している場合、サーバ A がダウンしてもサーバ B に接続できるように、通常ディレクトリツリーを別のサーバに複製しておきます。

必要な従来のサーバが物理的にダウンするか、高い要求のために単に滞っただけの場合に使用できる1つまたは複数のバックアップサーバの IP アドレスがわかっているときは、サイレントフェイルオーバー機能を提供することにより LDAP 接続リソースを構成できます。

プライマリサーバが **server1.acme.com** に存在し、このディレクトリの複製が **server2.acme.com** および **server3.acme.com** に存在するとします。接続リソースを作成する際に、この3つのアドレスすべてを [Host or IP Address] フィールドに入力します(スペース文字で区切ります)。

The screenshot shows a 'Properties' dialog box with the following fields and values:

- Connection Type: LDAP Connection
- Host or IP Address: server1.acme.com server2.acme.com serv
- Port Number: 389
- Base DN: (empty)
- User DN: cn=admin,o=admin
- Password: ****
- Version: 3
- Use TLS:
- Time Limit: 200
- Size Limit: (empty)

Buttons: Test, Default, Help, OK, Cancel

表示されている例では、[Host or IP Address] に 3 つのアドレスが入力されているだけでなく、ダイアログボックスの下にある [Time Limit] では「200」ミリ秒と指定されています。コンポーネントがこのリソースを使用する場合、最初に（指定されているポート 389 で）**server1.acme.com** に接続します。このサーバがダウンした場合、または応答に 200 ミリ秒以上要した場合、次のアドレス **server2.acme.com** で新しい接続が実行されます。そこでサーバから 200 ミリ秒以内に応答がなかった場合、**server3.acme.com** が使用されます。最後に、接続できるサーバがなくなった場合に例外がスローされます。

注記： この例では、ダイアログボックスの [Port Number] フィールドに入力されている値が 389 であるため、3 つのサーバすべてがポート 389 に接続します。すべてのサーバで同じポートを使用しない場合は、**server1.acme.com:389**、**server2.acme.com:636** のように複数の IP アドレスでコロンを使用して、フェイルオーバー用の接続を構築することができます。ただし、その場合はダイアログボックスの [Port Number] フィールドに 0（ゼロ）を入力する必要があります。

接続リソース作成後の編集

接続リソースは、いつでも戻って変更することができます。ナビゲータペイン (Composer ウィンドウの左端) でリソースの名前を見つけてダブルクリックし、最初にリソースを作成したときウィザードに入力した情報を含むタブ付きのダイアログボックスを表示します(前の図を参照)。必要なタブを選択し、該当するフィールドに新しい情報を入力します。[OK] をクリックすると、変更内容が保存されます。[キャンセル] をクリックすると、リソースの元の設定に戻ります。

3

LDAP コンポーネントの作成

Novell exteNd Composer の LDAP Connect を使用すると、ディレクトリに対応した XML 統合アプリケーションを作成できます。具体的にいうと、確立された LDAP API を、次にあげる各操作に利用できるということです。

- ◆ ディレクトリからのエントリの検索および取得
- ◆ ディレクトリへの新しいエントリの追加
- ◆ ディレクトリでのエントリの更新
- ◆ ディレクトリからのエントリの削除
- ◆ ディレクトリでのエントリの名前変更
- ◆ バインド操作
- ◆ 破棄操作

LDAP の検索機能により、「*list*」および「*read*」などのより複雑な X.500 操作がエミュレートされます。例は同じで、検索するベースオブジェクト、および検索対象となるツリーの部分(またはスコープ)を指定します。「フィルタ」では、検索によって特定のエントリを見つけるために満たされなければならない必要条件が指定されます(LDAP 検索操作では、DAP の場合と同一の機能が提供されますが、より簡潔な形式にエンコードされています)。

注記： 理論的には、クライアントが前の操作の結果を受信する前に操作が完了することを待つかどうかによって、LDAP API でアプリケーションは「同期的」または「非同期的」に操作を実行できるようになりますが、実際には、exteNd Composer の LDAP Connect でのすべての操作は同期的です。

LDAP Connect では、独自のカスタム Java コードを作成して LDAP SDK でクエリをパッケージおよびアンパッケージするのではなく、業界標準のクエリ応答の構文を使用して DSML(Directory Services Markup Language)形式の LDAP クエリを発行します。DSML が特に魅力的である理由は、XML であることにあり、通常の Composer アクションを使用して簡単にデータをマップできます。LDAP Connect が特に強力である理由は、低レベルの DSML 内部を知る必要なく DSML を自動生成できる機能にあります。

LDAP アプリケーションモデル

一般的に、LDAP アプリケーションによって次の 5 つの手順が実行されます。

- 1 LDAP サーバに接続します。**この手順には、セッションの初期化、セッションの初期設定、およびサーバへのバインドが含まれます。通常、セッションの初期設定には次のような項目の定義が含まれます。
 - ◆ 検索で返されるエントリの最大数
 - ◆ 検索時間の最大秒数
 - ◆ 参照の処理方法
 - ◆ セキュリティの初期設定
- 2 サーバに対して認証します。**クライアントは、パブリック権利を使用して匿名で認証するか、簡単な (クリアテキスト) パスワード認証を通じて、または完全な暗号化を使用して認証できます。Novell の LDAP サービス v3 では、完全な暗号化および認証に対して SSL3 が使用されます。
- 3 操作を実行して結果を取得します。**この手順には、通常はディレクトリの検索が含まれますが、ディレクトリデータの修正が含まれる場合もあります。
- 4 結果を処理します。**この手順には、カスタムのビジネス論理を通じて、返された情報を何らかの形で活用する操作が含まれます。
- 5 セッションを閉じます。**つまり、サーバからバインド解除し、セッションハンドルを削除します (Composer では、コンポーネントが範囲外に移動すると自動的にこの手順が実行されます)。

LADAP コンポーネントを作成する前に

すべての exteNd Composer コンポーネントと同様に、新しい LDAP コンポーネントを作成する最初の手順は、XML テンプレートが必要かどうかを決定することです (詳細については、『Composer ユーザガイド』の「新しい XML テンプレートの作成」を参照してください)。接続リソースを事前に作成している場合 (すでに説明しました) は、テンプレートのサンプルドキュメントを使用してコンポーネントに処理される入力および出力を表し、LDAP コンポーネントを作成できます。

➤ 新しい LDAP コンポーネントを作成する

- 1** Composer のメインメニューバーで、[File] メニューから [New xObject] > [Component] > [LDAP] の順に選択します。「Create a New LDAP Component」ウィザードが表示されます。

Create a New LDAP Component

Enter a Name and description for this LDAP component. The name will appear in the Composer window and in choice lists when you are prompted for objects of this type as you work in Composer. The Name is required and may not contain the characters: \ / : ? " < > . | Names are case insensitive (i.e. MyObjectName is the same as myobjectname).

Name:

Description:
 Purpose:
 Input:
 Output:
 Remarks:

Help **Back** **Next** **Cancel**

注記： 現在プロジェクトにLDAP 接続リソースが含まれていない場合は、この時点で作成するように要求され、図のようなダイアログボックスに到達する前に接続リソース作成プロセスを完了させます。

- 2 新しいLDAP コンポーネントの **[Name]** を入力します。
- 3 オプションとして、**[Description]** に独自の説明テキストを追加します。
- 4 **[Next]** をクリックします。新しいダイアログボックスが表示されます。

Create a New LDAP Component

Specify one or more XML Templates to help design Input to this Component or Web Service and only one to design Output. The sample XML Documents in each Template are design time aids to help you build Action Models for the component. The samples are not actually used at runtime after deployment to your application server. The Identifier is fixed and represents the name used to refer to the XML Document during component execution. Selecting System {ANY} allows you to use an empty template (i.e. accept any document as input).

Input Message

Part	Template Category	Template Name
Input	{System}	{ANY}

Add
Delete

Output Message

Part	Template Category	Template Name
Output	{System}	{ANY}

Add
Delete

Help **Back** **Next** **Cancel**

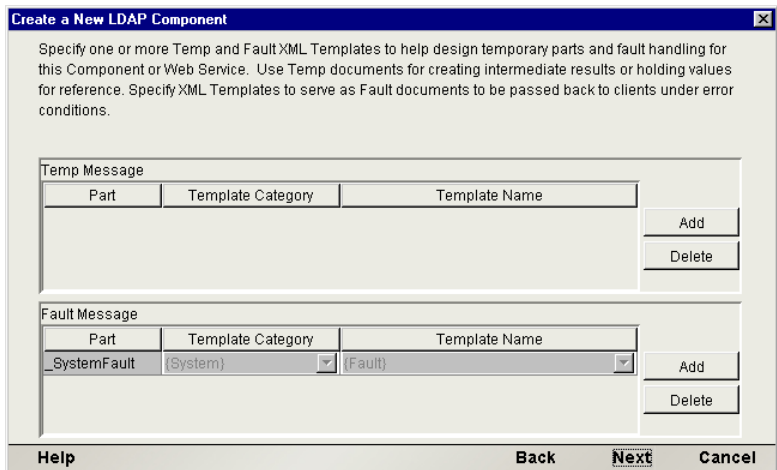
- 5 1つまたは複数の入力テンプレートを、次のように指定します。
 - ◆ デフォルトのカテゴリと異なる場合は、「テンプレートカテゴリ」を選択します。

- ◆ すぐ右にあるドロップダウンメニューを使用して、選択した「テンプレートカテゴリ」にある XML テンプレートのリストから「テンプレート名」を選択します。
- ◆ 入力 XML テンプレートをさらに追加するには、[Add] をクリックして、これらの手順を繰り返します。
- ◆ 入力 XML テンプレートを「削除する」には、エントリの中をクリックして [Delete] をクリックします。

6 [Output] の XML テンプレートを選択します。

注記： 出力テンプレートとして {System}{ANY} を選択すると、空の出力 XML テンプレートを指定できます。Map アクション内で ECMAScript または XPath を使用してカスタムの出力 DOM を動的に生成する場合にこのように指定します (詳細については、『Composer ユーザガイド』、特に XML Map コンポーネントの作成に関する章を参照してください)。

7 [Next] をクリックします。新しいダイアログボックスが表示されます。



- 8 このダイアログボックスでは、(オプションとして) 「スクラッチパッドの DOM」として使用する Temp ドキュメントをコンポーネントに追加できます。必要な場合は障害ドキュメントも追加できます。必要に応じて、[Add] ボタンおよび [Delete] ボタンを使用してドキュメントを追加または削除します。
- 9 [Next] をクリックします。ウィザードの接続パネルが表示されます。

Specify which Connection you wish to use for this Component or Service. To change any connection parameters, you must change them in the Connection Resource object or create a new Connection Resource of the same type with different parameters.

Connection: LDAPMasterTestConnection [Test]

Host or IP Address: XPath(*USERCONFIG/ldap_host_name*)

Port Number: CT.XPath(*USERCONFIG/ldap_host_port*)

Base DN: _____

User DN: CT.XPath(*USERCONFIG/ldap_login_dn*)

Password: _____

Version: ECT.XPath(*USERCONFIG/ldap_version*)

Use TLS:

Time Limit: 200

Size Limit: _____

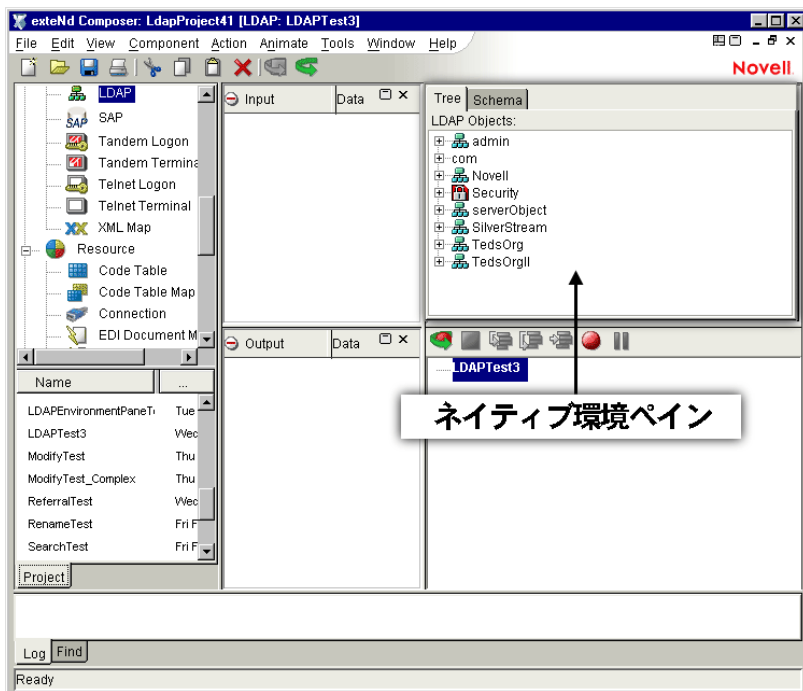
Buttons: Help, Back, Finish, Cancel

- 10** プルダウンリストから「Connection」を選択します (リストには既存の LDAP 接続リソースの名前が表示されます)。

注記： 必要に応じて、後でこの選択を変更して異なる接続リソースを使用できます。この接続を使用することで「ロック」されることはありません。

ヒント： 接続リソースのフィールドは、このダイアログボックスではグレー表示になっています。接続リソースを編集する必要がある場合は、コンポーネントの作成が終了した後で個別に開きます。

- 11** [Finish] をクリックします。空白のアクションモデルペインがある Composer のメインウィンドウが表示されます。右上のネイティブ環境ペインには、[Schema] タブおよび [Tree] タブがあります (ここでの「スキーマ」は、XML スキーマではなく LDAP ディレクトリスキーマのことです)。[Tree] タブが選択されている場合は、ターゲットディレクトリのツリービューが表示されます。次の図を参照してください。



この時点で、アクションモデルでアクションの作成を開始するか、または [Save] で作業を保存して後でコンポーネントに戻ってくることができます。

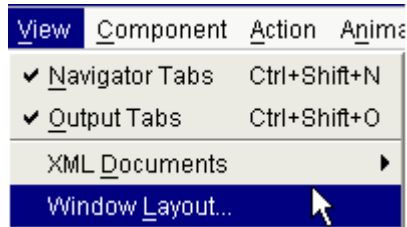
LDAP コンポーネントエディタの特殊な機能

LDAP コンポーネントエディタには、XML Map コンポーネントエディタのすべての機能が含まれます。また、アクションペインだけでなく、入力 XML ドキュメントと出力 XML ドキュメントのマッピングペインも含まれています。Create DSML および Execute DSML という 2 つの LDAP 固有のアクションに加えて、すべての通常の Composer アクション (XML Map、Function、Log、Decision、Send Mail など) が使用できます(これらのアクションの詳細については、次の章で説明します)。

LDAP ネイティブ環境ペイン

LDAP コンポーネントエディタには、ネイティブ環境ペイン (図を参照) が含まれ、コンポーネントの接続リソースによって参照されるディレクトリの検索可能なツリービューを提供する [Tree] タブ、および [Schema] タブがあります (後で詳しく説明します)。[Tree] タブのブラウザは、Novell の ConsoleOne ツリーブラウザと同じように動作します。

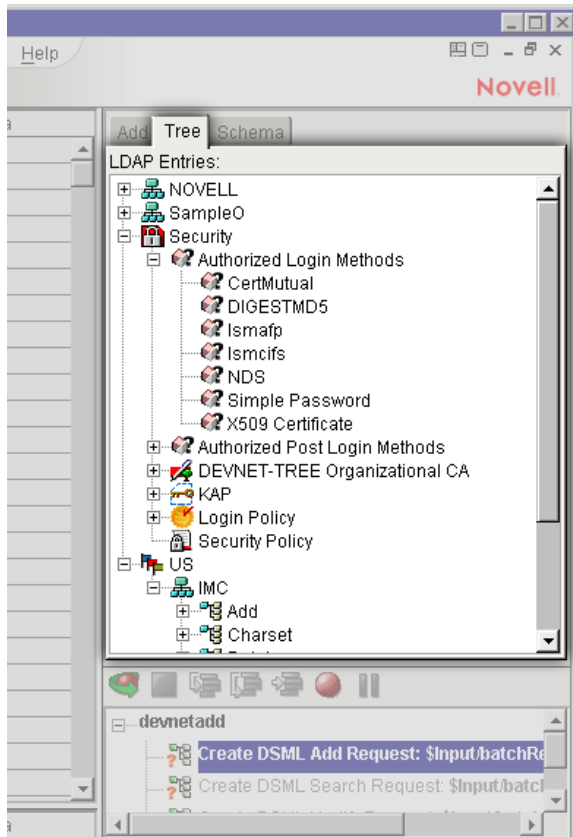
ネイティブ環境ペインの「デフォルト」の場所は、Composer メインウィンドウの右上端です。ただし、ペインの場所は、[View] から [Window Layout] メニューコマンドを使用すると変更できます。



このコマンドによって、ネイティブ環境パネル、XML ドキュメントウィンドウ、およびアクションモデルペインの東西南北の位置を指定できるダイアログボックスが表示されます。

[Tree] タブ

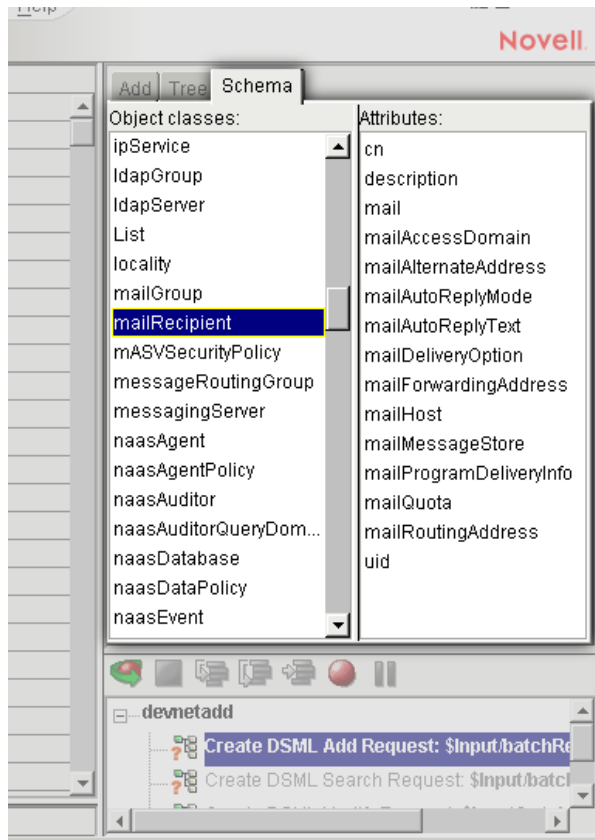
ネイティブ環境ペイン (次の図を参照) の [Tree] タブでは、ターゲットディレクトリでオブジェクトの場所を検索および検証する場合における、設計時のインタラクティブな視覚的補助が提供されます。このビューの内容は、コンポーネントを開くと自動的に表示されます。そのコンテンツは、コンポーネントの接続リソースで指定されるホストによって異なります。



注記： アイテムをドラッグしたりドロップしたりして、ディレクトリのツリービューの中または外に移動することはできません。

[Schema] タブ

[Schema] タブ (次の図を参照) は、この特定のツリーに対してスキーマによって定義されたオブジェクト / 属性の関係を検査する際に役立つ読み取り専用の設計時の補助です。スキーマを修正するアクションがコンポーネントに含まれる場合は、このタブを検査するだけで変更内容を視覚的に検証できます (ここでの「スキーマ」は、XML スキーマではなくディレクトリスキーマのことです)。

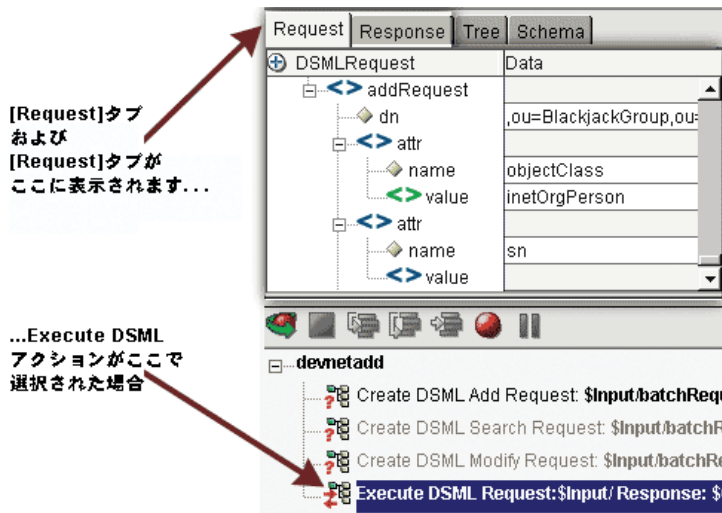


特定のオブジェクトに対して定義されるすべての属性を設定するには、タブの上部スクロールペインで任意のオブジェクト名をクリックします。それに対応して、下部スクロールペインで属性名のリストがリアルタイムで更新されます。

Composer の作業環境の詳細については、別冊の『Composer ユーザガイド』を参照してください。

[Request] タブおよび [Response] タブ

アクションモデルペインで Execute DSML アクションが選択 (ハイライト) されている場合は、[Request] および [Respond] という 2 つの特別なタブがネイティブ環境ペインに表示されます。これらのタブでは (次の図を参照)、実行中に Execute DSML アクションによって使用される DSML 要求 / 応答 DOM のツリービューが表示されます。



注記： タブは、最初は空白です。適切な DOM でタブを形成するには、アニメーションモードで（または Composer のメインツールバーにある [Execute All] ボタンを使用して）アクションモデルを実行してから、Execute DSML アクションを選択 / ハイライトします。

DSML の要求および応答 DOM は、変動しやすく（一時的なものであり）、ドラッグ元としてもドロップ先としても使用できません。これらの DOM 要素との間でマップするには、（次の章で説明されているように）Create DSML アクションを使用して「batchRequest」要素を「*Input*」にマップするか、または Execute DSML アクションを使用して「batchResponse」要素を「*Temp*」や「*Output*」などにマップしてから、DOM 間でドラッグアンドドロップします（または任意の方法で DOM コンテンツをマップします）。

DSML DOM の操作については、次の章でより詳しく説明します。

前後関係で決まるタブ

さまざまな「特色」がある Create DSML アクションをアクションモデルに追加すると、（ [Add]、[Attributes]、[Compare]、[Filter]、[Modify]、[Rename]、および [Search] などの名前が付いた）追加のタブが動的に表示されます。これらの各タブの表示および使用については、後で説明します。

ドラッグアンドドロップ操作

Composer の UI では、1 つの DOM ツリーから別の DOM ツリーに DOM ノードをドラッグアンドドロップでき、場合によっては、DOM ツリーからネイティブ環境ペインのフィールドにドラッグアンドドロップできます。データマッピング規則を指定するとき、ドラッグアンドドロップを使用しない場合、手動でコーディングした XPath 式または ECMAScript 式が必要となるため、ドラッグアンドドロップは便利で簡単です。

DOM から DOM

DOM から DOM へのドラッグアンドドロップを使用すると、ドキュメント間を渡るデータマッピングを指定できます (たとえば、DSML 応答ドキュメントの特定の位置から、カスタム DOM の特定の場所にデータを転送するように指定できます)。

ツリービューで (たとえば) 入力 DOM の DOM 要素をクリックし、別の DOM ウィンドウまでドラッグしてから特定の要素上でマウスを放すと、Composer によってソースのデータ、属性、およびチルドレン (その属性なども含む) のコピーがターゲット要素 (「ドロップ先」) にマップされます。同時に、生成されるマッピングに対応する Map アクションが、Composer によって自動的にコンポーネントのアクションモデルに追加されます。

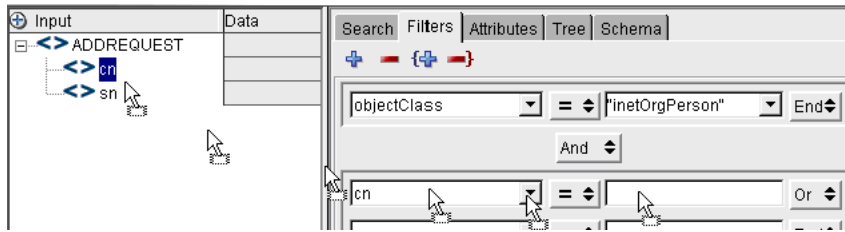
多くの場合、必要なマッピング動作は、このようなすべての子孫をマップする動作です。しかし、場合によっては、異なるマッピング動作が必要な可能性があります。データマッピングをより正確に制御するには、Map アクションを手動で作成し、[Map Action] ダイアログボックス (および [Advanced] ボタン) を使用して必要なマッピングを実現させます。

注記： これらの方法については、『Composer ユーザガイド』で詳しく説明されています。詳細情報についてはガイドを参照してください。

DOM から NEP(ネイティブ環境ペイン)

多くの場合、(たとえば) 入力 DOM の DOM 要素をクリックしてから、その要素をネイティブ環境ペインのテキストフィールドにドラッグして放すことによって、特定の XML 要素への参照でターゲットフィールドを形成できます。

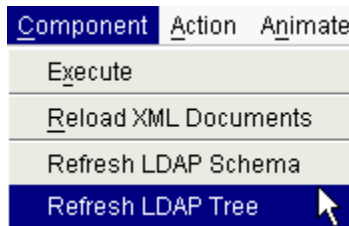
次の例について考えます。



この例で、ユーザは、検索フィルタを作成したり、ドラッグアンドドロップを活用して入力ドキュメントのノードをフィルタ式のパラメータにバインドしたりしています。ユーザは、[Input] で [ADDREQUEST] の下にある [cn] ノードをクリックして、ネイティブ環境ペインの空白フィールドまでドラッグしました。マウスを放すと、適切な ECMAScript 式、つまり「Input.XPath ("ADDREQUEST/sn")」が、検索式のターゲットフィールドに表示されます。(作成された) フィルタは、「objectClass」が「inetOrgPerson」に等しく、「sn」が「Input/ADDREQUEST/sn」の値に等しいエントリに一致します。

特殊なメニューコマンド

LDAP Connect 固有のメニューコマンドには、[Component] メニューにある 2 つのコマンドに加えて、2 つの新しいアクション ([Action] の [New Action] にある Create DSML および Execute DSML) が含まれます。



[Refresh LDAP Schema] コマンドでは、[Schema] タブ (前の説明を参照) のコンテンツが更新されます。このコマンドには、ランタイムでの重要性はありません。ターゲットディレクトリのスキーマを修正した後、Compser の [Schema] タブコンテンツを通じて反映される変更内容を設定する場合は、このコマンドが設計時に便利である可能性があります。

同様に、[Add] または [Rename] などの DSML 要求を実行することによって設計時にディレクトリのコンテンツを修正し、ネイティブ環境ペインの [Tree] タブで変更内容を確認する場合は、[Refresh LDAP Tree] コマンドが便利です (このコマンドは、ランタイムでは効果はありません)。

4

DSML アクション

「アクション」はプログラミングステートメントに類似しており、パラメータの形式で入力を受け付け、特定の操作を実行します。Java または ECMAScript における「式」と同様、アクションは Composer コンポーネントにおける実行の最小単位です。

コンポーネントで作成するアクションのリストを「アクションモデル」といいます。アクションモデルは統合アプリケーションの論理的な中心で、ディレクトリと XML ドキュメントの間のデータマッピング、データ変換、および転送が発生する場所です。これは、アプリケーションを構成するステートメントが整理されたリストです。

Composer で使用できるアクションの一部はデータ固有で、その他はループ、条件付き分岐、例外ベクトルのようなものを構成するコントロールフローに関係します。LDAP Connect により、*Create DSML* および *Execute DSML* という 2 つの主要な LDAP 関連アクションが提供されます。これらのアクションの使用については、後の節で詳しく説明します。

DSML の使用

DSML (Directory Services Markup Language) は、LDAP クエリの提示およびクエリに対する応答の保存に関する XML 文法です。場合によっては、LDIF (LDAP クエリおよびオブジェクトに対するテキストベースファイルの交換形式) の代わりになることもあります。

Composer では、DSML を使用してディレクトリをクエリする場合、次の 2 つの方法があります。

- ◆ 既成の DSML ファイルがある場合 (通常、サービスをトリガしたサーブレットからの入力として受け取られます)、Execute DSML アクションでそれを使用できます。Composer は DSML を使用して適当な LDAP クエリを構成し、対象のディレクトリサーバに対してそのクエリを実行します。
- ◆ 既成の DSML クエリドキュメントがない場合は、Create DSML アクションを使用します。Composer により必要な DSML が作成されます。

後者の場合、マウス操作で UI ツールを使用してクエリのパラメータを選択します。Composer によりパラメータ値を使用する DSML DOM が作成され、DOM ウィンドウに表示されます。他の Composer DOM ウィンドウと同様、DOM ウィンドウではドラッグアンドドロップによりデータを出し入れできます。

サーバからクエリ結果が返されると、DSML としてコンポーネントに返されます。この情報 (Output、Temp など) のターゲット DOM を指定すると、他の DOM と同様、その DOM を操作することができます。

次の図は、DSML の要求および応答の全体的な構造を表しています。

Input	Data
batchRequest	urn:oasis:names:tc:DSML:2.0:core
xmlns	
searchRequest	derefAlways
derefAliases	
dn	baseObject
scope	false
typesOnly	
filter	
equalityMatch	
attributes	
attribute	
name	*

Output	Data
batchResponse	urn:oasis:names:tc:DSML:2.0:core
xmlns	
searchResponse	256
requestID	
searchResultEntry	
searchResultDone	
requestID	256
resultCode	
code	0
descr	Success

この場合は応答ドキュメントが「Output」にマップされていますが、Temp またはその他の DOM にも簡単にマップできます。

DSML は LDAP のニーモニックおよび操作言語と類似しているため、DSML で指定された操作により LDAP SDK メソッドの署名にかなり近くマップできます。したがって、たとえば各応答には、XML 属性 code および descr を持つ resultCode という要素が含まれます。code 属性の値は、成功の場合ゼロ、その他（失敗）の場合はこのマニュアルの付録 B に示されている結果コード値のいずれかになります（順番に Novell の JLDAP SDK から送信されます）。descr 値は、エラーの原因を説明するプレーンテキストの文字列です（結果コード 34 に関連付けられる「Invalid DN Syntax」など）。

単一の DSML ドキュメントにおける複数の要求

1 つの DSML ドキュメントに複数の要求をまとめ、バッチ処理のような方法により 1 回のサーバクエリで一連の Add 操作、Add と Delete 操作などを実行するようサーバに伝えることができます。

この処理は、複数の連続する Create DSML アクションを構成し、そのすべてを同じドキュメント (通常 Input) の同じルートノード (batchRequest) にマップすることにより実行されます。Create DSML アクションが実行されるたびに、指定のルートノードに新しい要求が追加されます。DOM は、必要に応じて拡張します。最後に Execute DSML アクションが発生すると (通常 Create DSML アクションのリストの最後)、この DOM は 1 つのクエリとして送信されます。要求がバッチとして実行され、バッチによる応答が生成されます。

クエリ応答に複数の検索結果が含まれている場合、結果はその応答ドキュメント内で searchResultEntry 要素ノードの下に蓄積されます。

Search 要求では、通常ワイルドカードを含むフィルタを使用して複数「ヒット」することがあるため、要求に関する Create DSML アクションが 1 つだけの場合でも、応答には複数の結果を含めることができます。

The screenshot shows a software interface with two main panes. The left pane, titled 'Output', displays a hierarchical tree structure of search results. The root node is 'searchResultEntry', which contains several 'attr' (attribute) nodes. Each 'attr' node has a 'name' and a 'value' child. The right pane, titled 'Data', shows a table of values corresponding to the tree structure.

searchResultEntry	Data
dn	
requestID	256
attr	
name	errors
value	664
attr	
name	subschemaSubentry
value	cn=schema
attr	
name	directoryTreeName
value	XC
attr	
name	securityErrors
value	0
attr	
attr	

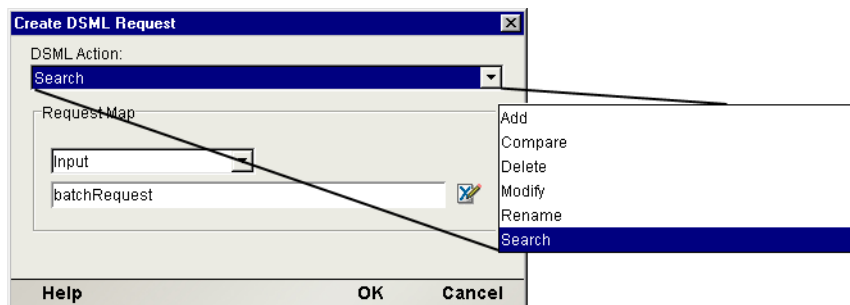
この例では、searchResultEntry 要素の下に多くの attr 要素が存在しています。ワイルドカードを使用した元のクエリでは、属性の配列が要求されています。これが、返された内容です。名前と値のペアがたくさんあることに注意してください。

Create DSML アクション

Create DSML アクションでは、次の LDAP に関する操作がサポートされています。

- ◆ **Add** - ディレクトリにエントリを追加します。
- ◆ **Compare** - 値の比較に基づき、ブール値を生成します (「このコンテナに paymentHistoryRef 属性が存在しない場合、その属性を追加するためにスキーマを変更する」というような論理操作を実行する場合など、ディレクトリにおける特定のオブジェクトまたはオブジェクトタイプの存在を検証する場合に便利です)。
- ◆ **Delete** - ディレクトリからエントリを削除します。
- ◆ **Modify** - ディレクトリの値を変更 (編集) します。
- ◆ **Rename** - エントリの名前を変更するか、エントリを移動します。
- ◆ **Search** - フィルタ条件に基づいてエントリを検索します。

これらの操作は、[Create DSML Request] ダイアログボックスのプルダウンメニュー (DSML Action の下) に表示されます。

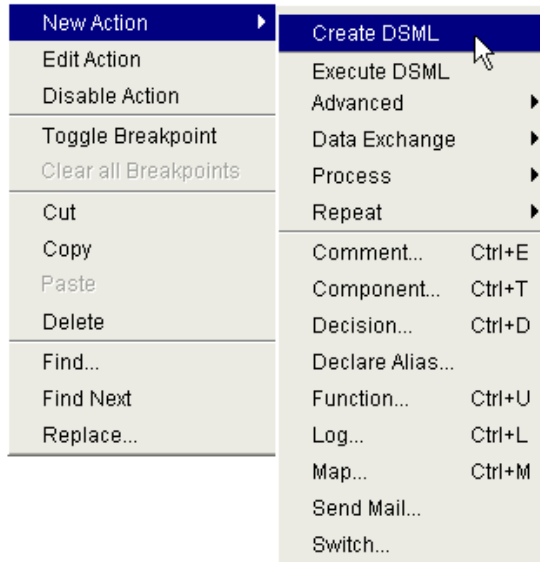


次に、各オプションについて順番に説明します。

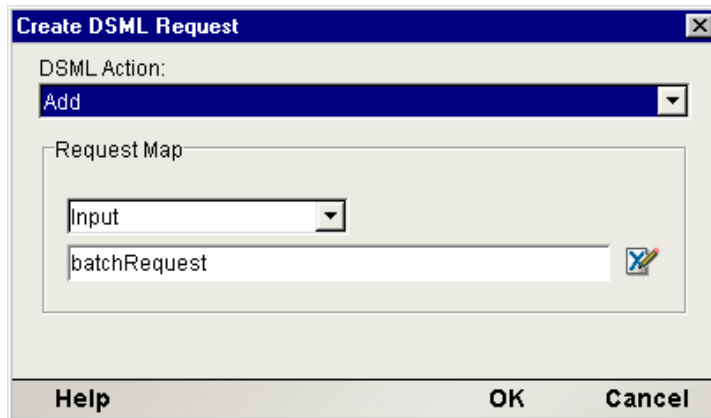
Add

➤ Add 要求を実行する Create DSML アクションを作成する

- 1 アクションモデルペイン内でマウスを右クリックして (または Composer メインメニューバーの [Action] メニューを使用して)、次の図のようにメニューから [New Action]、[Create DSML] の順に選択します。



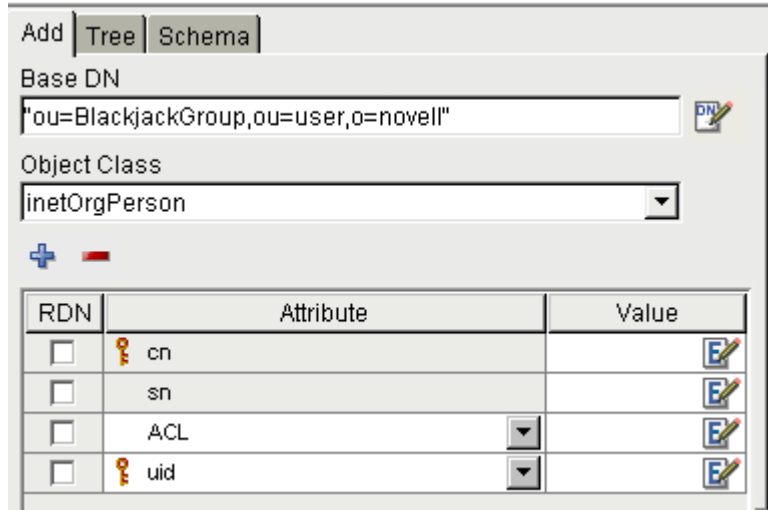
- 2 表示されるダイアログボックスで、プルダウンメニューから [Add] を選択します。











- 3 [Request Map] のプルダウンメニューを使用して、ターゲット DOM (またはターゲットメッセージ) を指定します。メニューについては、前にコンポーネントを作成したときに指定した DOM (メッセージ) の名前が表示されます。
- 4 [DOM-name] プルダウンメニューの下にあるテキストフィールドに、要求の作成に対するターゲットノードを表す XPath 式を入力します (ほとんどの場合、batchRequest のデフォルト値を受け入れることができます)。

重要： DSML クエリとして直接この DOM を渡す場合は、ルートノードの名前を変更しないでください。この DOM が有効な DSML ドキュメントであるためには、ルートの名前は batchRequest である必要があります。

- 5 [OK] をクリックしてダイアログボックスを閉じます。ネイティブ環境ペインのタブが更新されます。また、「Create DSML Add Request」というラベルが付いた新しいアクションが、アクションモデルに表示されます。



RDN	Attribute	Value
<input type="checkbox"/>	 cn	
<input type="checkbox"/>	sn	
<input type="checkbox"/>	ACL 	
<input type="checkbox"/>	 uid 	

- 6 このアクションで有効なDSML要求を作成するには、Add要求のベースDN(ディレクトリツリーのターゲットノードを表します) および追加される新しいエンティティのオブジェクトクラスを認識する必要があります。この情報を、ネイティブ環境ペインの [Add] タブで入力します (前の図を参照)。[Add] タブがまだ選択されていない場合は、選択します。
- 7 [Base DN] の横に、Add 要求が実行されるコンテナオブジェクトの名前を表す文字列を引用符で囲んで入力します。

注記： Composer には、この参照の作成に便利な機能があります。テキスト入力フィールドの右側にある [DN] アイコンをクリックすると、[Expression Builder] ダイアログボックスが表示されます。このダイアログボックスには、ネイティブ環境ペインの [Tree] タブで見られるように、ディレクトリがツリー表示されます。選択ツリーの項目をダブルクリックすると、ターゲットツリーノードに対して適切な形式の DN 文字列が Composer によって自動的に生成されます。

- 8 [Object Class] というラベルの下の (事前に入力された) プルダウンメニューを使用して、ツリーに追加するオブジェクトの「オブジェクトタイプ」を選択します。この処理を実行する場合、タブの下部にある Attributes テーブルは自動的に更新され、対象のオブジェクトクラスの必要なネーミング属性が表示されます。

- 9 パネルの下の部分では、対象のコンテナまたはオブジェクトクラスに応じて属性が入力されます (値は入力されません)。属性の中には、対象のオブジェクトクラスに必要なものもあります。これらの属性は一色の背景で表示されます。また、属性の中には (必須またはオプションに関わらず) ネーミング属性として使用できるものもあります。これらの属性は、小さな縦のキーアイコンで表示されます。

注記: キーによる表示および背景のシェーディングは、対象の LDAP サーバが Novell eDirectory によって起動されている場合のみ使用できます。その他の場合には、[Attribute] リストのアイテムはプレーンテキストの形式で表示されます。

視覚的なヒントの考えられるすべての組み合わせは、次の図に示されています。

The screenshot shows a dialog box with tabs for 'Add', 'Tree', and 'Schema'. The 'Base DN' field contains 'ou=BlackjackGroup,ou=user,o=novell'. The 'Object Class' dropdown is set to 'inetOrgPerson'. Below this is a table of attributes:

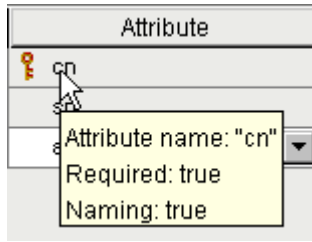
RDN	Attribute	Value
<input type="checkbox"/>	cn	<input type="text"/>
<input type="checkbox"/>	sn	<input type="text"/>
<input type="checkbox"/>	ACL	<input type="text"/>
<input type="checkbox"/>	uid	<input type="text"/>

cn属性およびsn属性はinetOrgPersonオブジェクトに必要です。これは、cnセルおよびsnセルの一色の背景によって示されます (これらの属性は [Attribute] 列から削除できないことからわかります)。

この例のその他の2つの属性は (青いプラス記号のボタンによって) ユーザによって追加されています。これらの属性はACLおよびuidです。これらの属性は白の背景を持ち、対象のオブジェクトクラスの属性に対しては必要ではないことを示しています。

uid属性 (および必要なcn属性) には横にキーアイコンが付いており、uidはinetOrgPersonのインスタンスの「ネーミング属性」として使用できることを示しています (つまり、[uid] または [User ID] フィールドはユーザの識別名および相対識別名の一部として使用できます)。

注記: 項目上にマウスを置いてしばらくそのままの状態にすると、マウス移動ヘルプヒントによって項目のステータスについての説明が表示されます。



- 10 項目の相対識別名の一部として使用するために属性を指定するには、対象の属性の左にある [RDN] チェックボックスをオンにします。単にキーが存在するだけでは、属性は新しいエントリの RDN にはなりません。チェックボックスをオンにする必要があります。
- 11 メンバーを属性リストに追加するには、「プラス記号 (+)」をクリックして、[Attribute] セルを追加します (その後、新しいセルに表示されるプルダウンメニューから属性を選択します)。エントリを「削除」するには、対象のエントリをクリックして (焦点を合わせる)、「マイナス記号 (-)」をクリックして削除します。
- 12 必ず値を各属性と関連付けてください。入力する値は、有効な ECMAScript 文字列または文字列を評価する式であることが必要です。リテラルなデータを入力する場合、値を引用符で囲む必要があります (**John** ではなく、**笛 ohñi** と入力します)。

注記： E アイコン (右側) をクリックして [Expression Builder] ダイアログボックスを表示して、入力するか選択ツリーの項目をダブルクリックすることによって式をインタラクティブに作成することもできます。

Add 要求 (詳細な例)

o=mondocorp の下に user と呼ばれるコンテナ (organizationalUnit オブジェクト) を設定したと仮定します。

ここで、コンテナオブジェクト user に Joey Jacobs という新しい inetOrgPerson を追加するとします。また、UserID の値および Joey のアシスタントの名前など、Joey に関するその他の情報も保存するとします。ディレクトリに新しいエントリを追加するには、前に説明されているとおりに (Add モードで) Create DSML アクションを作成して、ネイティブ環境ペインを次のように設定します。

Add | Tree | Schema

Base DN

Object Class

+ -

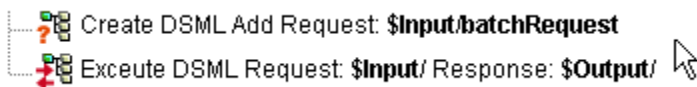
RDN	Attribute	Value
<input checked="" type="checkbox"/>	cn	"Joey"
<input type="checkbox"/>	sn	"Jacobs"
<input type="checkbox"/>	assistant	"Bettina Kilroy"
<input checked="" type="checkbox"/>	uid	132845901

この設定の内容は、「Joey の一般名 (cn) および Jacobs の名字 (sn) を持つ inetOrgPerson を、assistant および uid の値が表示された状態で、mondocorpと呼ばれるorganizationコンテナのuser organizationalUnit (ou) に追加する」です。

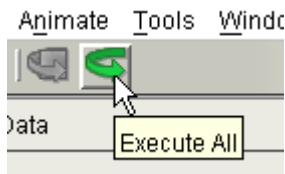
チェックボックスは、新しいエントリの相対 DN を形成する場合にどの情報を使用するかを示しています。この例では、新しいRDNはcn=Joey,uid=132845901であるため、ベース DN が ou=user,o=mondocorp であるコンテナに配置されます。このため、新しいエントリの完全識別名は次のようになります。

cn=Joey,uid=132845901,ou=user,o=mondocorp

Create DSML アクションを実行すると、この要求に対して DSML が作成され、Input (または [Create DSML Request] ダイアログボックスで指定した DOM) に配置されます。ただし、要求はまだサーバに送信されません。これを実行するには、Execute DSML アクションを作成する必要があります (要求が Input DOM に存在する場合、Execute DSML アクションを追加してダイアログボックスのデフォルト値を受け入れます)。アクションモデルは次のようになります。

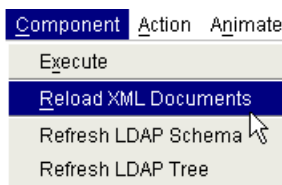


Composer メインツールバーの [Execute All] ボタンをクリックすると、アクションをテストできます。

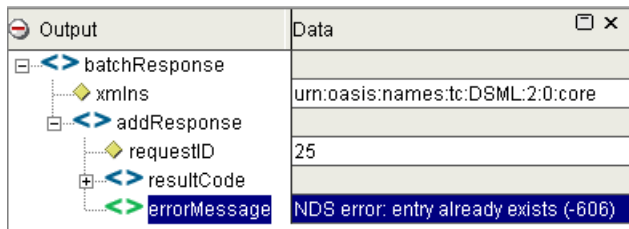


アクションが実行され、出力 DOM に内容が表示されます。出力の結果コードが 0 である場合、操作が成功したことを示します。

この操作が実際に成功だったことを確認するには、メインメニューバーに移動して [Component]、[Reload XML Documents] の順に選択します。



再び [Execute All] ツールバーアイコンをクリックします (コンポーネントを再び実行します)。今回は、出力 DOM にエラーメッセージが表示されます。



errorMessage 要素に、作成しているエントリがすでに存在していることを示すメッセージが含まれているため、アクションモデルの最初のテストが成功だった (Joey Jacobs がツリーに追加された) ことがわかります。

注記： この動作は、X.500 ディレクトリの構造に関する重要な原則を表しています。2つの兄弟オブジェクト (ツリーの同じレベルに存在するインスタンスオブジェクト) が同じ ID を所有することはできません。つまり、ここで試したように同じオブジェクトを 2 度追加することはできない、ということです。2 回目の Add 要求で、エラーが発生します。

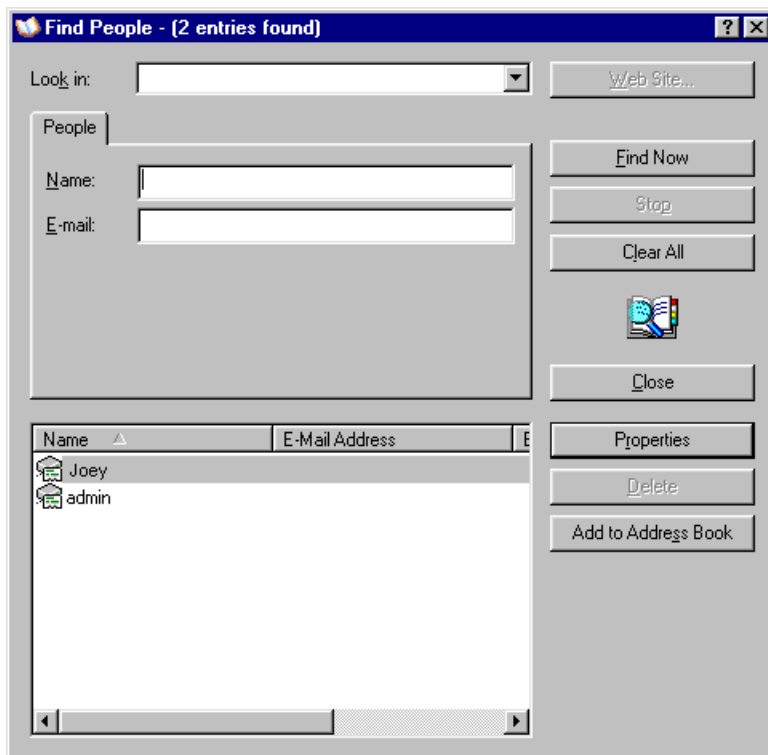
ディレクトリをクエリする場合、Web ブラウザなどサードパーティの LDAP クライアントを使用して Add 要求を検証することもできます。

ほとんどの Web ブラウザは LDAP クライアントで、**ldap://** URL プロトコルをサポートしています。前の例で実行した Add 要求を検証する場合、次の URL を Microsoft Internet Explorer のアドレスウィンドウに入力できます。

ldap://[server-domain]/ou=user,o=mondocorp?cn?sub?objectClass=inetorgperson

この URL の実際の内容は、「ldap: プロトコルを使用して [server-domain] にあるサーバに移動し、user コンテナに匿名でバインドしてから、そのコンテナの下にあるサブツリーで inetOrgPerson オブジェクトを検索して、cn(一般名)に対応する属性値を取り出す」ということです。

Microsoft Internet Explorer でこの URL に移動すると、[アドレス帳] ウィンドウが開き、検索結果が表示されます。



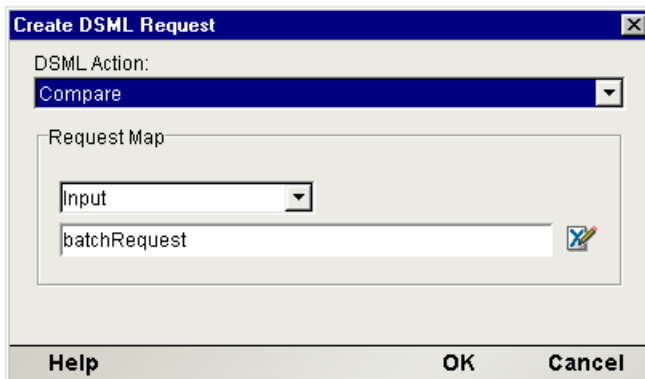
一般名「John」という新しいエントリがリストに表示されています。

Compare

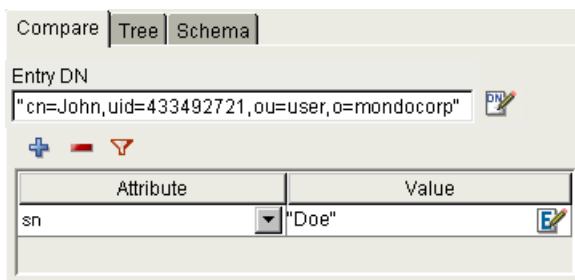
Compare 操作は、ディレクトリエントリの存在を確認する場合に便利です。Compare 要求のセットアップ手順は、(前に説明した) Add 要求を実行する場合と似ています。

➤ Compare 要求を実行する Create DSML アクションを作成する

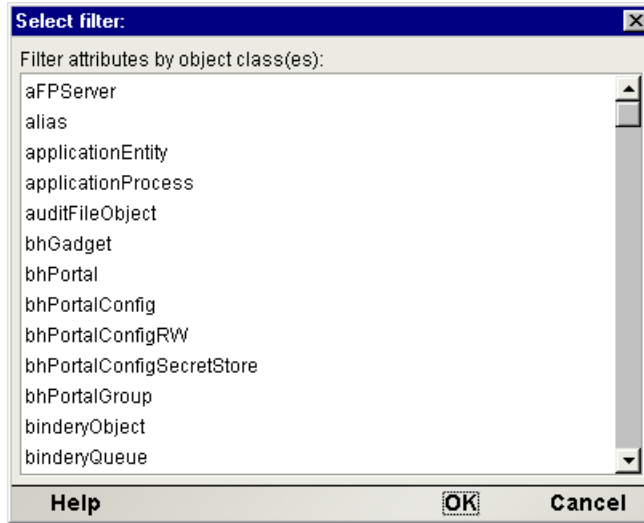
- 1 アクションモデルペイン内でマウスを右クリックして(またはComposerメインメニューバーの [Action] メニューを使用して)、メニューから [New Action]、[Create DSML] の順に選択します。
- 2 プルダウンメニューを使用して、[Compare] を選択します。



- 3 [Request Map] に入力する値がわからない場合は、表示されるデフォルト値を受け入れます。[OK] をクリックすると、アクションモデルにアクションが追加されます。
- 4 ネイティブ環境ペインで [Compare] タブが最前面に表示されていない場合は、このタブをクリックします。
- 5 [Entry DN] の横に、比較するオブジェクトの識別名を入力します(下の図に表示されているように、文字列を引用符で囲みます)。



- 6 小さなろうとの形をした [Filter] アイコンをクリックします (プラスおよびマイナスアイコンの横にあります)。スクロールリストを含むダイアログボックスが表示されます。



- 7 存在を確認するエントリに対応するオブジェクトクラスを選択します(1度だけクリックしてハイライトします)。
- 8 [OK] ボタンをクリックして、[Select Filter] ダイアログボックスを閉じます。
- 9 ネイティブナビゲーションペインの下の部分にある [Attribute-Value] テーブルで、プルダウンメニューを使用して属性名を選択します。この属性のリストは、前の手順で選択したオブジェクトクラスに使用できる属性に対応します。
- 10 必要に応じて「プラス記号」アイコンをクリックし、[Attribute-Value] テーブルに他のエントリを追加します。
- 11 各属性名に対して対応する値を [Value] の下に入力します。

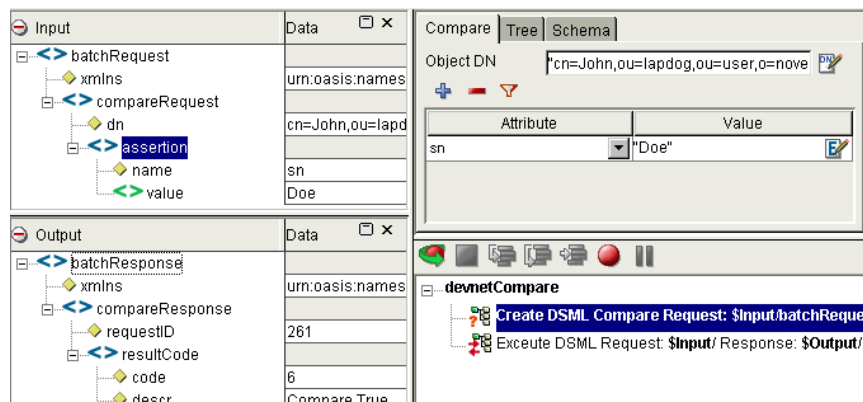
注記: 値がプログラム(ECMAScript)式でない場合、値を引用符で囲む必要があります。
これで、アクションをテストできます。

Compare 要求の例

前の Add 要求の例では、テストディレクトリの lapdog コンテナノードに John Doe を追加しました (セットアップの詳細については、例を参照してください)。一般名 John と名字 Doe を持つエントリの存在を確認するには、lapdog という名前の organizationUnit オブジェクトで、次を実行する必要があります。

- ◆ 前の説明に従って、正しく Create DSML アクションを作成する
- ◆ 前の表示に従って、[Compare] タブをセットアップする
- ◆ アクションモデルに Execute DSML アクションを追加する (Execute DSML アクションのセットアップダイアログボックスのデフォルト値を受け入れる)
- ◆ ステップスルー (アニメーション) モードで、または Composer メインツールバーの [Execute All] ボタンを使用して、アクションモデルを実行します。

モデルを実行した後のコンポーネントウィンドウは、次の図のように表示されます。



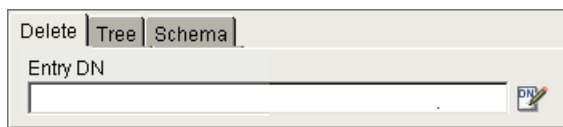
Delete

Delete 操作は、ディレクトリからエントリを削除する場合に便利です。

Delete 要求のセットアップ手順は、(前に説明した) Add 要求を実行する場合と似ています。

➤ Delete 要求を実行する Create DSML アクションを作成する

- 1 アクションモデルペイン内でマウスを右クリックして(またはComposerメインメニューバーの [Action] メニューを使用して)、メニューから [New Action]、[Create DSML] の順に選択します。
- 2 プルダウンメニューを使用して、[Delete] を選択します。
- 3 ネイティブ環境ペインで [Delete] タブが最前面に表示されていない場合は、このタブをクリックして前面に表示します。
- 4 [Delete] タブで、[Entry DN] の横に削除するエントリの固有な DN (識別名) を入力します。



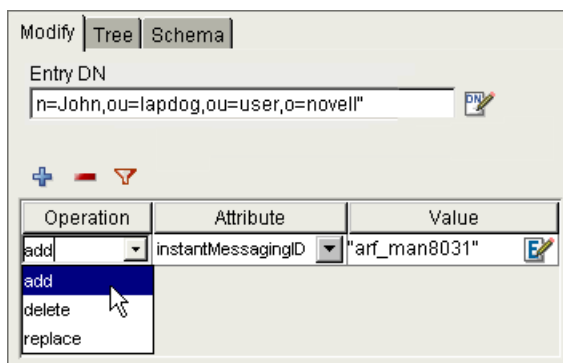
これで、アクションを使用できます。アクションを実行すると、DSML ターゲットドキュメントに適当な要求が追加されます。

Modify

Modify 操作は、ディレクトリの値を追加、削除、または置換する場合に便利です。たとえば、ディレクトリに **John Doe** のエントリはすでに含まれているものの、彼のインスタントメッセージング ID のエントリがまだ含まれていない場合、Modify 要求の「追加」を使用してディレクトリに彼のインスタントメッセージングハンドルを追加できます。その実行方法を次の手順で説明します。

➤ Modify 要求を実行する Create DSML アクションを作成する

- 1 アクションモデルペイン内でマウスを右クリックして(またはComposerメインメニューバーの [Action] メニューを使用して)、メニューから [New Action]、[Create DSML] の順に選択します。
- 2 プルダウンメニューを使用して、[Modify] を選択します。
- 3 ネイティブ環境ペインで [Modify] タブが最前面に表示されていない場合は、このタブをクリックして前面に表示します。
- 4 [Modify] タブで、[Entry DN] の横に変更するエントリの固有な DN (識別名) を入力します。



- 5 小さなろうとの形をした [Filter] アイコンをクリックします (プラスおよびマイナスアイコンの横にあります)。スクロールリストを含む [Select Filter] ダイアログボックスが表示されます。

- 6 スクロールリストで、変更するエントリに対応するオブジェクトクラスをクリックします。
- 7 **[OK]** ボタンをクリックして、**[Select Filter]** ダイアログボックスを閉じます。
- 8 ネイティブ環境ペインの **[Modify]** タブで「プラス記号」をクリックし、メッセージを追加します。
- 9 前の図に表示されているように、実行する操作のタイプ (「追加」、「削除」、または「置換」) をプルダウンメニューから選択します。
- 10 **[Attribute]** 列でプルダウンメニューを使用して、値を追加、削除、または置換するものに対応する属性を選択します (属性リストに、前に **[Select Filter]** ダイアログボックスで選択したオブジェクトクラスに対応する属性が表示されます)。
- 11 **[Value]** 列で、属性に対する適切な値を入力します。

これで、アクションを使用できます。アクションを実行すると、DSML ターゲットドキュメントに適切な要求が追加されます。

Rename

Rename 操作を使用すると、その使用方法に応じてディレクトリ内におけるエントリの名前の変更または移動ができます。

次の手順のスクリーンショットは、John という一般名 (cn) から Johann という cn への変更を示しています

➤ Rename 要求を実行する Create DSML アクションを作成する

- 1 アクションモデルペイン内でマウスを右クリックして (または Composer メインメニューバーの **[Action]** メニューを使用して)、メニューから **[New Action]**、**[Create DSML]** の順に選択します。
- 2 プルダウンメニューを使用して、**[Rename]** を選択します。
- 3 ネイティブ環境ペインで **[Rename]** タブが最前面に表示されていない場合は、このタブをクリックして前面に表示します。
- 4 **[Rename]** タブで、**[Old DN]** の横に削除または移動するエントリの固有な DN (識別名) を入力します。
- 5 **[New RDN]** の横に、このエントリの新しい相対 DN を入力します (右側に **[k]** アイコンが表示されている場合、値を引用符で囲む必要はありません。このアイコンについては、前の Add 要求の説明を参照してください)。

- 6 エントリを移動する場合は [New Parent DN] の横に新しいペアレントコンテナ DN を入力します (オプション)。
- 7 エントリを新しい場所に移動して元の場所にコピーを残さない場合は、[Delete Old RDN] チェックボックスをオンにします (オプション)。

これで、アクションを使用できます。

Search

Search 操作は、ディレクトリからデータを取り出す場合に使用します。LDAP は、複雑な検索条件を作成するために組み合わせることができる論理操作に関連するクエリのフィルタ構文を定義します。構文については RFC 2254 に記述されているため、ここでは繰り返しません。Composer では、UI コントロールの使用に基づき適切な式構文が自動的に生成されます (後の説明を参照)。

注記： メールのフィルタ条件を作成するために電子メールクライアントで提供されるルールビルダ GUI に詳しい場合は、ここでも同じ原理 (および GUI メタファ) を適用できます。

➤ Search 要求を実行する Create DSML アクションを作成する

- 1 アクションモデルペイン内でマウスを右クリックして (または Composer メインメニューバーの [Action] メニューを使用して)、メニューから [New Action]、[Create DSML] の順に選択します。
- 2 プルダウンメニューを使用して、[Search] を選択します。
- 3 ネイティブ環境ペインで [Search] タブが最前面に表示されていない場合は、このタブをクリックして前面に表示します。

Search Filters Attributes Tree Schema

Base DN
"ou=lapdog,ou=user,o=novell"

Scope
wholeSubtree

Dereference Aliases
derefAlways

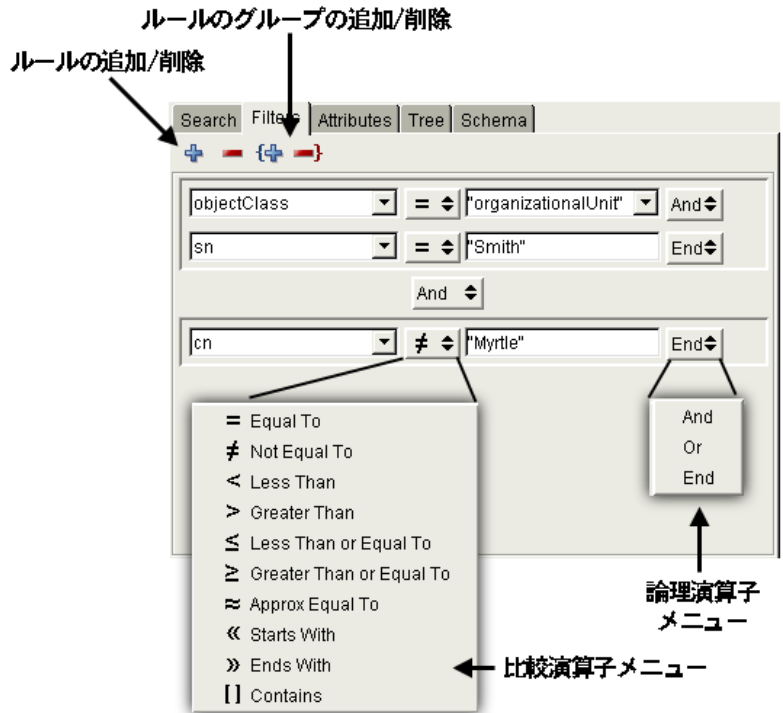
Size Limit
0 k

Time Limit
10 k

Types Only

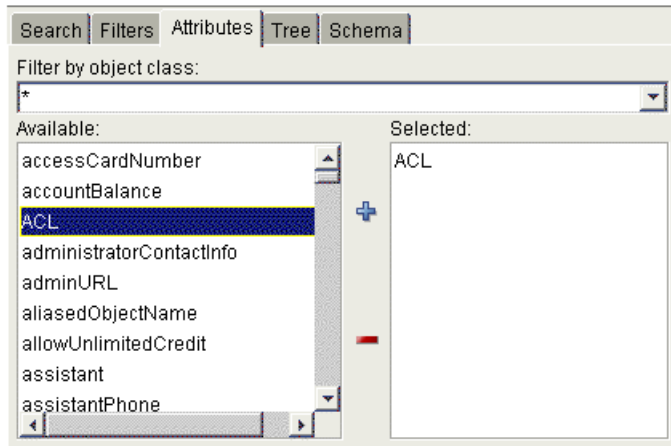
- 4 [Base DN] に、検索を開始するコンテナオブジェクトの識別名を入力します (文字列が引用符で囲まれていることを確認してください)。
- 5 [Scope] に対して、3つの値のいずれかを選択します。
 - ◆ **baseObject** - ベースオブジェクト ([Base DN] で指定されたもの) のみに検索範囲を制限します。
 - ◆ **singleLevel** - 検索の範囲は、ベースオブジェクトとすぐ下のチャイルドレベル (1つ下のレベル) です。
 - ◆ **wholeSubtree** - 検索の範囲には、チャイルドレベルの深さに関係なく、ベースオブジェクトおよびその下に存在するすべてのチャイルドが含まれます。
- 6 [Deref Alias] で、プルダウンメニューから次の値のいずれかを選択します。
 - ◆ **derefAlways** - 検索の開始ポイントを見つける場合およびその開始エントリより下のエントリを検索する場合のいずれでも別名 (別のエントリへのポインタを含むエントリ) が常に逆参照されます。
 - ◆ **derefFindingBaseObject** - 検索の開始ポイントを見つける場合は別名が逆参照 (解決) されますが、開始エントリより下を検索する場合は逆参照されません。
 - ◆ **derefInSearching** - 検索の開始ポイントより下のエントリを検索する場合は別名が逆参照されますが、開始エントリを見つける場合は逆参照されません。
 - ◆ **neverDerefAliases** - 別名のエントリは検索されません。

- 7** [Size Limit] に、コンポーネントがサーバから受信する最大「ヒット」数を表す数値を入力します (サーバで応答バッチサイズが制限されている場合がありますが、これは一般的にクライアントから制御することはできません)。デフォルト値はゼロで、その場合は返される項目の数に制限はなく、コンポーネントの動作はすべての結果が受信されるまでブロックされます。
- ヒント:** フライアウトメニューを使用して、定数の [k] を選択するか、ランタイム時に数値を解決するプログラム (ECMAScript) 式を入力する場合は [ECMA Expression] を選択します。説明については、前に登場した「Add 要求」の手順を参照してください。
- 8** [Time Limit] に、サーバが要求を処理する間コンポーネントが待機する最大時間を表す数値を秒単位で入力します (ミリ秒単位ではありません)。この値に応じてサーバは検索を中止し、エラーを返します。この値を空白 (またはゼロ) にすると、待機時間は制限されず、クライアント (コンポーネント) は永久に待機します。
- ヒント:** 当然ながら、永久に待機することはお勧めしません。ここには適当な値を入力してください。
- 9** 検索で見つかった属性の名前のみを返し、それに関連付けられている値は必要ない場合、[Types Only] チェックボックスをオンにします (オプション)。
- 10** [Filter] タブをクリックして、前面に表示します。
- 11** さまざまな機能を使用して、検索をフィルタする場合の規則を作成します (後の説明を参照してください)。



この図における設定内容は、「*objectClass* は *organizationalUnit* に等しく、かつ名字が *Smith*、かつ一般名は *Myrtle* に等しくない」ということとなります。

- 12** [Attributes] タブをクリックして、前面に表示します。
- 13** [Filter by object class] のプルダウンメニューを使用してオブジェクトクラス名を選択し、その結果左下の [Available] サブペイン内に表示されるオブジェクトクラスに使用可能な属性タイプのリストを表示します。
- 14** プラス記号およびマイナス記号のアイコンを使用して、[Attribute] 列から [Selected] 列に属性名を移動します (その反対も可能)。



この例では、isManager 属性および fullName 属性が organizational Person に対して使用可能な属性のリストから選択されています。つまり、検索結果には、(存在する場合は)「ヒット」の isManager 属性および fullName 属性 (およびそれぞれの値) が含まれます。

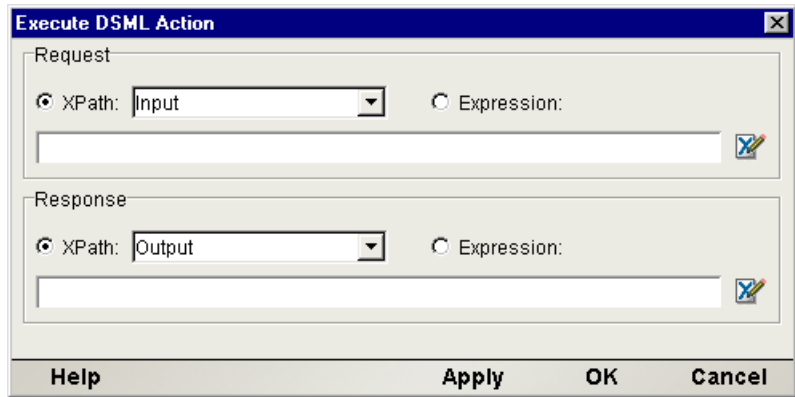
Execute DSML アクション

Execute DSML アクションにより、DSML でパッケージ化された要求が低いレベルの LDAP 要求に変換され、サーバに対して (同期的に) 実行されます。このアクションはサーバから受信したあらゆる応答を取り出し、選択した XML ドキュメント (通常、出力 DOM または一時 DOM) にマップします。

有効な Execute DSML アクションをセットアップするには、Composer に対して DSML 要求ドキュメント (メモリ内 DOM) の検索場所、および要求が実行された後に返される DSML 応答のマップ先を指定する必要があります。

➤ Execute DSML アクションを作成する

- 1 アクションモデルペイン内でマウスを右クリックして (または Composer メインメニューバーの [Action] メニューを使用して)、メニューから [New Action]、[Execute DSML] の順に選択します。ダイアログボックスが表示されます。



- このダイアログボックスでデフォルト設定を変更する必要がない場合は、**[OK]**をクリックしてすぐにダイアログを閉じます。この設定では、Composer に対して「要求に Input の DSML を使用し、Output に応答をマップする」ように指定されています。別の設定が必要な場合（たとえば、Output ではなく Temp、Temp1、またはその他の「スクラッチパッド DOM」に応答をマップする場合など）、ラジオボタン、プルダウンメニュー、およびテキストフィールドを使用して、適切な要求および応答 DOM（およびターゲットノード）を指定します。

注記：以前に Composer で Map アクションを作成したことがある場合は、これらの機能を理解しやすくなります。以前に実行したことがない場合、XPath から XPath へのマップの指定方法については、『Composer ユーザガイド』を参照してください。

- [OK]** をクリックして、ダイアログボックスを閉じます。アクションモデルに新しい Execute DSML アクションが追加されます。

LDAP コンポーネントエディタでの他のアクションの使用

これまで説明した DSML 関連アクションの他に、LDAP コンポーネントエディタにおけるその他すべての標準的な Composer コンポーネントのアクションを使用できます（各アクションの完全な説明については、別の『Composer ユーザガイド』を参照してください）。

Composer メインメニューバーの [Action] メニューには、基本的なアクションおよび高度なアクションの両方が表示されます。次の表で、そのアクションについて簡単に説明します。

基本的なアクション	説明
Comment	アクションモデルを記録します。特に、アクションモデルに [Decisions] または [Repeats]、あるいはその両方が使用されている場合、コメントを使用して処理を明確にすることができます。
Component	別のコンポーネントを実行し、呼び出されたコンポーネントで受け渡しするランタイム DOM を指定します。
Decision	指定した条件に基づいて、アクションの 2 つのセットのうちいずれかを実行できます。コンポーネントの実行で指定した条件がどのように解決されるかによって、True または False へのパスの分岐を処理します。
Function	ECMAScript 関数または以前に作成したカスタムスクリプトのいずれかを実行します。カスタムスクリプトは、Composer のカスタムスクリプトリソースエディタを使用して作成できます (『Composer ユーザガイド』の第 9 章を参照してください)。
Log	コンポーネントに指定されているさまざまなログファイルに情報を書き込みます。ログのタイプには、システム出力、システムログ、およびユーザログの 3 種類があります。
Map	要素のデータのある XML DOM から別の XML DOM へ転送し、オプションで変換します。
Send Mail	コンポーネントの実行中、指定した電子メールアドレスに自動的に電子メールを送信します。
Switch	入力値と大文字小文字の値との一致に基づいて、プログラムの制御をアクションの特定のブロックに分岐させることができます。これは、長く、読み取りが困難な if/else (Decision アクション) のチェーンを排除する場合に使用できる便利なアクションです。

次の表で説明されている高度なアクションは、コンポーネントエディタの [Action] メニューで、[Advanced]、[Data Exchange and Repeat] の順にサブメニューを選択すると利用できます。

高度なアクション	説明
Apply Namespaces	Namespace プリフィックスを上書きしたり、新しい Namespace プリフィックスを宣言したり、または Namespace 全体を無視したりする方法を指定します。

高度なアクション	説明
Raise Error	条件を評価し、true の場合は ERROR と呼ばれるグローバル変数に式のコンテンツを記述します。単独で使用された場合は、例外をスローしてコンポーネントを停止し、サービスに制御を返します。Try On Error アクションの Execute 分岐内で使用された場合は、評価され、On Error 分岐でアクションに制御が渡されます。
Simultaneous Components	2 つまたはそれ以上のコンポーネントを同時に (つまり、マルチスレッド方式で) 実行できるようにします。
Transaction	非コンテナ管理サービスの一部として配備されるコンポーネントで <i>User Transaction</i> コマンド (<i>begin</i> 、 <i>commit</i> 、および <i>rollback</i>) を呼び出したり、コンテナ管理 EJB 配備の一部となるコンポーネントで <i>setRollbackOnly</i> を呼び出したりできます。
Try On Error	一連のアクションを実行することで、エラーを生成するアクションに応答します。Try On Error アクションは、本質的にエラートラップおよび解決を行うアクションです。
XSLT Transform	XSL ファイルの指示に従って XML ファイルを変換します。出力は、一般的に Web ブラウザに XML ファイルを表示するために使用されます。

次の表で、Data Exchange アクションおよび Repeat アクションについて説明します。

Data Exchange アクション	説明
URL/File Read	XML でないファイル形式を Composer に読み込むことができます。
URL/File Write	ファイルを XML 以外の別の形式で書き込むことができます。
WS Interchange	WSDL リソースで定義されたメッセージおよび操作を使用して Web サービスを実行します。
XML Interchange	外部 XML ドキュメントをコンポーネントの DOM に読み込んだり、外部 XML ドキュメントにコンポーネントの DOM を書き込んだりします。読み込み / 書き込みメソッドには、ファイル、FTP、HTTP、および HTTPS プロトコルを使用した Get、Put、Post、および Post with Response が含まれます。

Repeat アクション	説明
Break	Repeat for Element、Repeat for Group、または Repeat While ループの実行を停止し、ループ外で次のアクションの実行を続行します。

Repeat アクション	説明
Continue	Repeat for Element、Repeat for Group、または Repeat While ループで現在のループ反復の実行を停止し、次の反復で同じループの一番上から続行します。
Declare Group	複数回発生する要素に基づきグループを作成して、グループに名前を付けることができます。グループは、Repeat for Group アクションで使用されます。
Repeat for Element	DOM ツリーに指定した要素が発生するごとに 1 つまたは複数のアクションを繰り返します。Repeat for Element アクションでは、複数回発生する要素に基づき、ループを作成できます。
Repeat for Group	グループの各メンバーに対して 1 つまたは複数のアクションを繰り返します。Repeat For Group アクションでは、データを再作成して、データを集約計算できます。
Repeat While	ループを作成することで、1 つまたは複数のアクションを繰り返します。Repeat While アクションでは、処理ループを任意の有効な ECMAScript 式に基づかせることができます。

5

LDAP および DSML の使用

この章は、Composer 対応の LDAP Connect に適用される LDAP のプログラミングイディオムを理解するために構成されており、各種の高度なLDAPおよびComposer機能を合わせて使用する方法、特にテストとデバッグを中心に説明します。

DSE クエリの例

LDAP により提供される便利な検索メカニズム（および監査機能）は、ディレクトリサーバの DSE ルートまたは DSA 固有のエントリをクエリする機能です (DSA とは *directory system agent* の略で、ディレクトリサーバに対する X.500 の用語です)。

サーバで LDAP バージョン 3 がサポートされている場合、DSE ルートでサーバに関する「メタ」情報を検索できます。DSE ルートのエントリは、実際にはそれ自体がアドレス可能なオブジェクトではありませんが、特別な構文を利用してクエリすることができます (後で説明します)。

DSE のエントリをクエリすることにより、次の内容を検索できます (他にもあります)。

- ◆ サポートされているセキュリティメカニズム
- ◆ サポートされている拡張機能
- ◆ 実装 (ベンダのバージョン) 情報
- ◆ サーバの起動後に発生した簡易認証および厳密認証のバインド数
- ◆ サブツリーに対するスキーマの場所

次の例では、www.nldap.com に存在する Novell のパブリックサーバの DSE エントリをクエリするコンポーネントを構築する場合の手順を示します。ただし、(すでに前の節で説明されている) それぞれの動作やリソースを構築する段階ごとの詳細を繰り返すのではなく、設計セッションの一部としてコンポーネントをテストおよびデバッグする方法など、LDAP コンポーネントの構築に関連する広範囲なワークフローに焦点を当てます。

匿名バインドの接続リソース

まず、接続リソースが必要です。この例では、LDAP 接続リソースに次の設定を使用します。パスワードが指定されていないことに注意してください。そのため、バインドは (定義により) 匿名になります。

Create a New Connection Resource

Enter the Host name or IP address and port of your LDAP Server. LDAP servers typically use ports 389 and 636 (for TLS). The 'User DN' must be an LDAP distinguished name, such as "cn=userid,o=Organization". Enter the 'Password' required for this user. Checking 'Default' makes this Connection the initial selection when creating a LDAP Component. Use the Test button to check your connection.

Connection Type: LDAP Connection

Host or IP Address: www.nldap.com

Port Number: 389

Base DN:

User DN: anonymous

Password:

Version: 3

Use TLS:

Test

Default

Help Back Finish Cancel

コンポーネントおよびアクションモデル

次に、(検索要求タイプの) Create DSML アクションおよび Execute DSML アクションの 2 つを含む、DSETest と呼ばれる新しい LDAP コンポーネントを作成します。

DSE ルートダンプの検索パラメータは単純です。次の値でフィルタ設定された空のベース DN に関するベースレベルの検索のみが必要です。

(objectClass=*)

ネイティブ環境ペインの [Search] タブを次のように設定します。

Search Filters Attributes Tree Schema

Base DN
"ou=lapdog,ou=user,o=novell"

Scope
baseObject

Dereference Aliases
derefAlways

Size Limit
0 k

Time Limit
10 k

Types Only

注記： この例には、故意にバグが含まれています。この設定を使用すると、エラーが発生します（問題を見つけてみてください）。

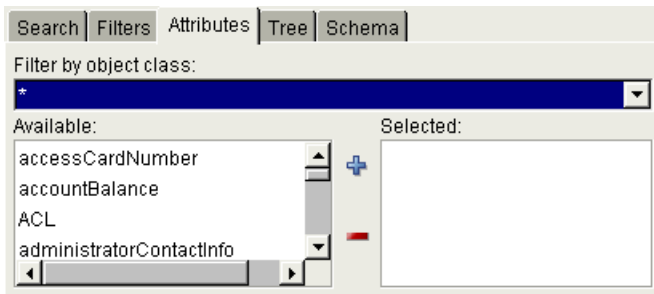
[Search] タブは次のようになります。

Search Filters Attributes Tree Schema

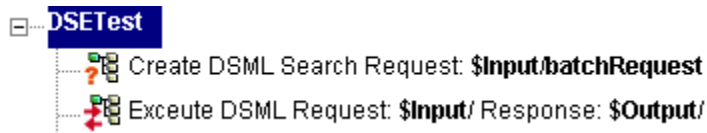
+ - {+ -}

objectClass = *** End

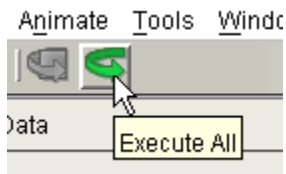
すべての属性に関する情報を受信するために、[Attributes] タブには何も入力しません（デフォルト値を受け入れます）。[Selected] ペイン（右下）が空白であることに注意してください。このリストが空白の場合、デフォルトでは、クエリで「ヒット」したすべての属性がサーバにより返されます。



テストのため、単純に (Create アクションから) DSML を [Input] にマップし、応答を [Output] にマップします。したがって、アクションモデルは最初、次のようになります。



アクションをテストするには、メインツールバーの [Execute All] ツールアイコンをクリックするだけです。



コンポーネントを実行すると、出力 DOM にデータが存在しています。ただし、注意して確認すると、問題があることがわかります。出力ドキュメントがかなり短く見え、resultCode がゼロではありません。

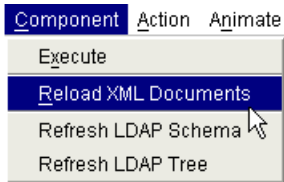
Input	Data
batchRequest	
xmlns	urn:oasis:names:tc:DSML:2:0:core
searchRequest	
derefAliases	derefAlways
dn	org.mozilla.javascript.Undefined@3cd5b5
scope	baseObject
timeLimit	10
typesOnly	false
filter	
equalityMatch	
name	objectClass
value	*

Output	Data
batchResponse	
xmlns	urn:oasis:names:tc:DSML:2:0:core
searchResponse	
requestID	46
searchResultDone	
requestID	46
resultCode	
code	34
descr	Invalid DN Syntax

出力 DOM の下にあるエラーの説明には「Invalid DN Syntax」と表示されます。それから、searchRequest における dn の横、データ値が org.mozilla.javascript.Undefined@3cd5b5 の部分について入力 DOM ツリーで反映されます。

問題： [Search] タブのベース DN 値には、有効な ECMAScript が必要です。このフィールドには何も入力しませんでした。その代わりに、間に何も入力しない 2 つの引用符を使用して、空白文字列を指定する必要があります。

修正は簡単です。[Base DN] フィールドに戻り、2 つの引用符を入力します。ただし、再び [Execute All] ボタンを押す前に、既存の DOM をクリアする必要があります。これを実行するには、メインメニューバーに移動して [Component]、[Reload XML Documents] の順に選択します。



DOM のウィンドウが、元の (空の) 状態にリセットされます。

注記： DOM がリセットされない場合、実行の次のラウンドでは Create DSML アクションにより既存の入力ドキュメントにデータ (別の要求) が単に追加されます。コンポーネントを実行すると、出力ドキュメントには元の (正しくない) 要求および入力ドキュメントに追加された要求に対する 2 つの応答が含まれます。

コンポーネントを再実行すると、エラーなく動作します。出力 DOM はかなり長く、DSE ルートエントリに対応する多くの attr 要素が含まれています。

XML Element	Data Value
urn:oasis:names:tc:DS	urn:oasis:names:tc:DS
requestID	49
dn	
requestID	49
attr (name)	errors
attr (value)	21423
attr (name)	subschemaSubentry
attr (value)	cn=schema
attr (name)	directoryTreeName
attr (value)	DEVNET-TREE
attr (name)	securityErrors
attr (value)	969
attr (name)	compareOps
attr (value)	718

エラーの処理

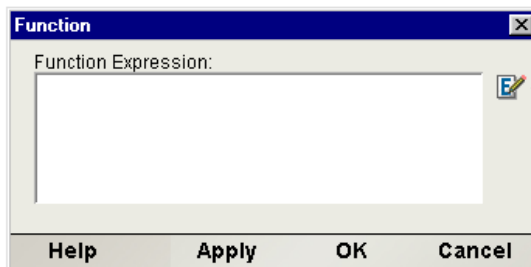
例に示されているように、LDAP または DSML が正しくなくても、アクションモデルレベルでは操作自体でエラーは発生しません。クエリのコンテキストで LDAP の例外が発生した場合、resultCode 要素の DSML ドキュメントでクライアントに報告されるだけです。アプリケーションで例外の状態を認識し、特別なアクションを実行する場合がありますが、ただ情報が転送されるだけの場合があります。クエリが正常に行われなかった理由を認識する必要がある場合、これに対処する論理を追加する必要があります。

例として、アプリケーションで不正な DSML 要求のログを作成します。1 つの方法は、次に示す動作が行われるようアクションモデルで論理を含めることです。「成功以外の結果の場合、その理由をシステム出力 (またはログファイル) に書き出す」

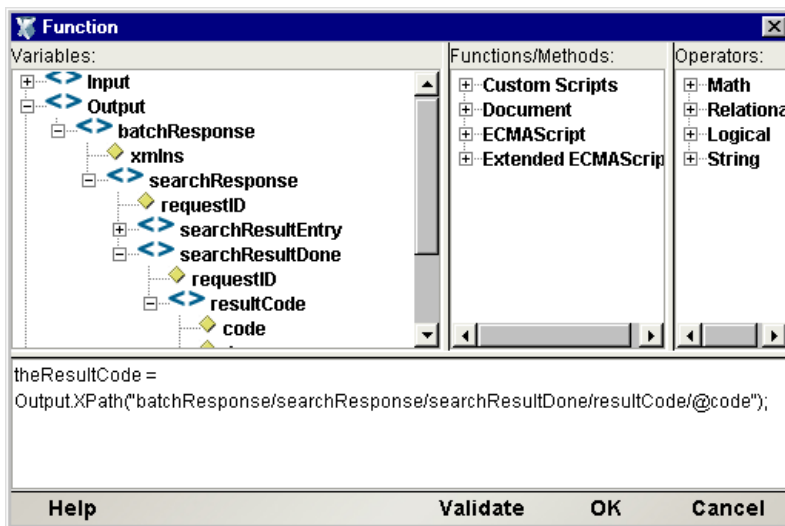
これを実行する方法は次のとおりです。

➤ クエリの失敗を通知するログを作成する

- 1 アクションペイン (アクションモデル) 内で、新しいアクションを表示する場所にカーソルを置きます。
- 2 メインメニューバーの [Action] メニューを使用して、[New Action]、[Function] の順に選択します。[Function Action] ダイアログボックスが表示されます。
- 3 小さな [E] アイコンをクリックすると、[Expression Builder] ダイアログボックスが開きます。



- 4 Expression Builder で、resultCode ノードおよびそのチャイルドノードが表示されるまで Output ノードツリーを展開します。code 属性ノードをダブルクリックします。テキスト編集ペインに、ECMAScript 式が表示されます。



注記： この例では、DSML ドキュメントがすでに出力 DOM にロードされていることを想定しています。必要に応じて Create DSML および Execute DSML アクションを構成し実行すると、Expression Builder ツリーに完全に参照可能な (表示と類似した) 出力 DOM が作成されます。

式では、resultCode の XPath 構文が Composer の拡張メソッド XPath () への呼び出しの中でラップされます。Expression Builder を使用すると視覚的なマウス操作だけで XPath に関連する ECMAScript 式を作成できるため、手動で式を入力する (そしてデバッグする) 必要はありません。

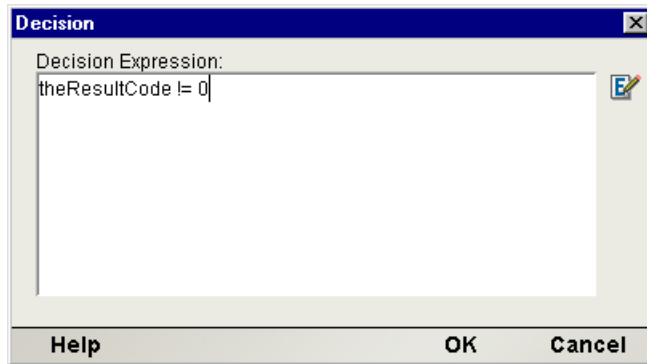
- 5 ECMAScript 式の前に、「theResultCode =」(かぎ括弧は不要) と入力します (これはスクリプト変数に結果コード値を保存する簡単な方法です)。
- 6 [OK] をクリックして、[Function Aciton] ダイアログボックスに戻ります。ダイアログボックスの編集ウィンドウに次のステートメントが含まれていることを確認します。

```
theResultCode =
Output.XPath ("batchResponse/searchResponse/searchResultDone/res
ultCode/@code");
```

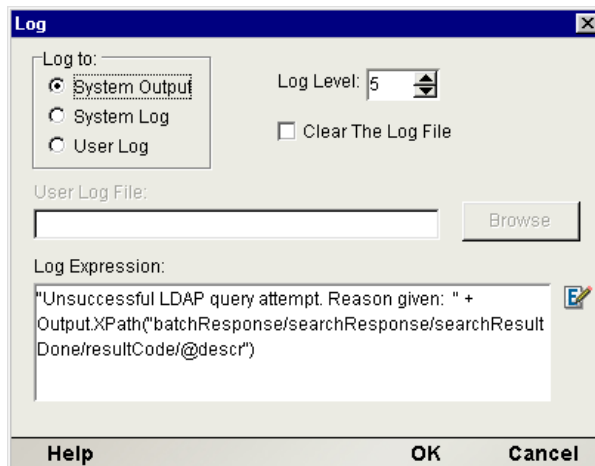
(すべて 1 行で表示されます。改行は無視してください。)

- 7 [OK] をクリックするか、[Enter] を押して [Function Action] ダイアログボックスを閉じます。新しいアクションがアクションモデルに表示されます。

メインメニューバーの [Action] メニューを使用して、[New Action]、[Decision] の順に選択します。[Decision Action] ダイアログボックスが表示されます。

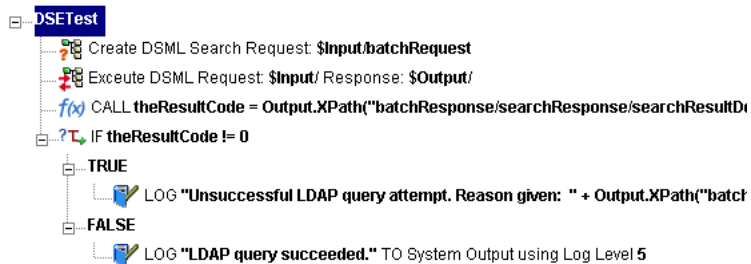


- 8 表示されているとおり、ボックスに「theResultCode != 0」と入力します。これは、[Condition] に対する正しい条件です。theResultCode がゼロの場合 (正常)、条件は False(偽) です。値がゼロ以外の場合、条件は True(真) になります。
- 9 [OK] をクリックするか、[Enter] を押してダイアログボックスを閉じます。新しい Decision アクションがアクションモデルに表示されます。
- 10 Decision アクションの後、TRUEおよびFALSEステートメントがそれぞれの行に表示されます。TRUE 行を 1 回だけクリックします。
- 11 メインメニューバーの [Action] メニューを使用して、[New Action]、[Log] の順に選択します。[Log Action] ダイアログボックスが表示されます。



- 12** [Log to] の下にあるラジオボタンのいずれかを選択して、このアクションによりログが作成されたメッセージの保存先を指定します。ここではテストのために [System Output] を選択しました。そうすると、(Composer の場合) ログメッセージが Composer のメインウィンドウの下にある [Log] タブに表示されるため、設計時のテスト中に簡単に認識できます。ランタイムでは、ユーザまたはシステムログに設定し直すこともできます。
- 13** このアクションのログメッセージで表示される、わかりやすい任意のテキストを入力します (TRUE 分岐に存在するため、このメッセージのログはクエリ結果がゼロ以外、つまり問題があった場合に作成されることに注意してください)。この例では、Expression Builder を使用して出力 DSML ドキュメントでプレーンテキストのエラーメッセージを検索する ECMAScript ステートメントを作成しました。
- 14** [OK] をクリックして、ダイアログボックスを閉じます。
- 15** 必要に応じて Decision アクションのアクションモデルで FALSE 行を 1 回だけクリックし、すでに説明した手順を FALSE 分岐で繰り返して Log アクションを作成します (この分岐はクエリが問題なく正常に実行された場合に対応しています)。

この時点で、アクションモデルは次のようになります。



- 16** メインメニューバーで (DOM ウィンドウをパーズするために) [Component]、[Reload XML Documents] の順に選択して、コンポーネントまたはその個別のアクションを実行し、テストします。
- 17** [Save] をクリックして保存します。

ECMAScript と LDAP Connect

Composer の ECMAScript をバインドすると、Java オブジェクトへの直接的な呼び出しを含め、プログラムによるさまざまな操作を実行するための強力で柔軟なツールが提供されます。Composer 対応の LDAP Connect では Novell の JLDAP ライブラリ (すでにプロジェクトの CLASSPATH に含まれています) が使用されるため、Java レベルの LDAP API を直接簡単に呼び出すことができます。次の例でその方法を示します。

ECMAScript を使用する LDIF の例

DSML が登場する前は、LDAP オブジェクトおよびクエリを実行する場合、ディレクトリユーザはテキストベースの LDIF(LDAP Data Interchange Format) ファイルタイプにかなり (今でもそうですが) 依存していました (LDIF 仕様は RFC2849 で公表されています)。LDIF は、XML と同様に人間が認識でき、比較的簡単なルールで構成され、プラットフォームを選ばないなど多くの理由で便利です。

クエリ結果の DSML バージョンは、次のようになります。

```
<?xml version="1.0" encoding="UTF-8"?>
<batchResponse xmlns="urn:oasis:names:tc:DSML:2:0:core">
  <searchResponse requestID="160">
    <searchResultEntry dn="cn=admin,ou=lapdog,ou=user,o=NOVELL"
requestID="160">
      <attr name="cn">
        <value>admin</value>
      </attr>
      <attr name="loginTime">
        <value>20030315051622Z</value>
      </attr>
      <attr name="objectClass">
        <value>inetOrgPerson</value>
        <value>organizationalPerson</value>
        <value>person</value>
        <value>top</value>
        <value>ndsLoginProperties</value>
      </attr>
      <attr name="securityEquals">
        <value>ou=lapdog,ou=user,o=NOVELL</value>
      </attr>
    </searchResultEntry>
  </searchResponse>
</batchResponse>
...
[ etc. ]
```

同様に、LDIF バージョンは次のようになります。

```
version: 1

dn: cn=admin,ou=lapdog,ou=user,o=NOVELL
cn: admin
loginTime: 20030315051622Z
objectClass: inetOrgPerson
objectClass: organizationalPerson
objectClass: person
objectClass: top
objectClass: ndsLoginProperties
securityEquals: ou=lapdog,ou=user,o=NOVELL
. . .
[ etc. ]
```

LDIF ファイルの構造は、レコードが 1 行あたり 1 つ存在し各レコードがコロンで区切られた属性 - 値の対を表すような単純なフラットファイルの構造とさほど異なります (構文はそれほど単純ではありませんが、この内容とかなり近いものです)。

監査のために、特定のツリーオブジェクトの「LDIF ダンプ」を取得すると便利なことがあります。実行にはそれほど多くの ECMAScript は使用しません。

プログラムによりディレクトリをクエリして結果を LDIF ファイルに書き出すには、カスタムスクリプトリソースの作成から始まります (Composer のメインメニューバーを使用して、**[File]**、**[New xObject]**、**[Resource]**、**[Custom Script]** の順に選択します)。 **[Custom Script Editor]** ウィンドウ内で、次のようなカスタム関数を入力します。

```

function query2Ldif( ldapHost,
    loginDN,
    password,
    searchBase,
    scope,
    searchFilter,
    attribs,
    fileName )
{
    var jldap = Packages.com.novell.ldap;
    var ldapPort    = jldap.LDAPConnection.DEFAULT_PORT;
    var ldapVersion = jldap.LDAPConnection.LDAP_V3;
    var typesOnly   = false;

    var lc = new jldap.LDAPConnection();

    // An ECMAScript array reference cannot
    // be passed to a Java method that expects
    // a Java array. So we have to copy our
    // 'attribs' array into a legit Java array:
    var javaStringArray =
        java.lang.reflect.Array.newInstance(
            java.lang.String,
            attribs.length);
    for (var i = 0; i < attribs.length; i++)
        javaStringArray[i] =
            new java.lang.String(attribs[i]);

    // bind to server
    lc.connect( ldapHost, ldapPort );
    lc.bind(ldapVersion, loginDN, password );

    // set up file I/O objects
    var fos = new java.io.FileOutputStream(fileName);
    var writer = new jldap.util.LDIFWriter(fos);

    // send query
    var results = lc.search( searchBase,
        scope,
        searchFilter,
        javaStringArray,
        typesOnly);

    // write the results
    while ( results.hasMore() )
        writer.writeEntry( results.next() );

    // clean up
    writer.finish();
    fos.close();
    lc.disconnect();

    java.lang.System.out.println("LDIF file written.");
}

```

関連する手順を一続きにすべて実行する形で該当するすべての機能性を表現しているため、この関数は故意に多少大規模になっています (実際には、クエリをバインドして発行する関数とクエリ結果を LDIF ファイルに書き出す関数の 2 つの小規模な関数に分解できます)。

8 つの引数は、LDAP ディレクトリのバインドに必要な標準パラメータおよびクエリの構成に必要な標準パラメータを表しています。

関数内部の最初の行は次のとおりです。

```
var jldap = Packages.com.novell.ldap;
```

この行により、「Packages」で始まるやや長めのコンテキスト文字列に対して (jldap の) 簡単な表記を使用できます (ECMAScript を使用すると、Packages の間接化技法で Java メソッドにアクセスできます)。この略記を作成した後、次のように記述できます。

```
var lc = new jldap.LDAPConnection();
```

次のように記述する必要はありません。

```
var lc = new Packages.com.novell.ldap.LDAPConnection();
```

JLDAP オブジェクトを取得する必要がある場合に、多少入力が少なくなります。

LDAPConnection オブジェクトがインスタンス化された後、次を入力してツリーにバインドできます。

```
lc.connect( ldapHost, ldapPort );  
lc.bind(ldapVersion, loginDN, password );
```

これらの行は標準の JLDAP 呼び出しで、『Novell NDK Javadocs』に記載されています。

次に、file-I/O オブジェクトを設定します。

```
var fos = new java.io.FileOutputStream(fileName);  
var writer = new jldap.util.LDIFWriter(fos);
```

ファイルは引数 fileName に入力されたパスに書き出されます。

サーバをクエリするには、1 行のコードが必要です。

```
// send query  
var results = lc.search( searchBase,  
                        scope,  
                        searchFilter,  
                        javaStringArray,  
                        typesOnly );
```

検索結果は単純に列挙し、書き出すことができます。

```
while ( results.hasMoreElements() )  
    writer.writeEntry( results.nextElement() );
```

最後に、すべてのファイル、ストリーム、および接続を閉じます。

```
writer.finish();
fos.close();
lc.disconnect();
```

スクリプトのテスト

スクリプトをテストする場合、**Function** アクションで次のテキストを入力して実行できます。

```
query2Ldif( 'www.nldap.com'
            'anonymous',
            '', /* no password */
            '', /* no search base DN */
            0, /* 0 for base-object scope */
            '(objectClass=*)', /* filter == all objects */
            new Array('*'), /* attrib array */
            'c:\\temp\\test.ldif' )
```

このスクリプトを持つ **Function** アクションを実行すると、Novell パブリックサーバで **DSE** ルート情報がクエリされます。**LDIF** ファイルはディスクに書き出されます。

LDIF ファイルは、従来のテキストエディタで確認できます。次の内容が含まれています (一部)。

```
# This LDIF file was generated by the LDIF APIs of Novell's Java LDAP SDK
version: 1
```

```
dn:
errors: 21428
subschemaSubentry: cn=schema
directoryTreeName: DEVNET-TREE
securityErrors: 972
compareOps: 218
bindSecurityErrors: 850
outBytes: 2279628812
vendorVersion: eDirectory v8.7.0 (10410.57)
simpleAuthBinds: 383333
supportedFeatures: 1.3.6.1.4.1.4203.1.5.1
supportedFeatures: 2.16.840.1.113719.1.27.99.1
repUpdatesIn: 0
abandonOps: 572
supportedSASLMechanisms: EXTERNAL
supportedSASLMechanisms: DIGEST-MD5
supportedSASLMechanisms: NMAS_LOGIN
referralsReturned: 0
removeEntryOps: 6683
searchOps: 654935
addEntryOps: 20253
strongAuthBinds: 32
modifyEntryOps: 888
vendorName: Novell, Inc.
listOps: 0
modifyRDNops: 9
chainings: 300
...
[ etc. ]
```


A

LDAP 用語集

CA

Certification Authority (認証局)。デジタル証明書を発行したり、証明書の信頼性を保証したりする機関。

cn

一般名。

DAP

Directory Access Protocol (ディレクトリアクセスプロトコル) の略 (X.519)。

dn

「識別名」を参照

DSA

(ディレクトリサーバエージェント) の略で、ディレクトリサーバまたは (L)DAP ホストに対する X.500 の用語。

DSE

DSA specific entry (DSA の固有なエントリ) の略で、サーバの機能を記述する、ディレクトリのルートレベルのエントリ。

DSML

Directory Services Markup Language の略で、ディレクトリの情報および要求をエンコードするための XML 文法。

JLDAP

Novell により開発されたオープンソースの LDAP SDK である Java LDAP ライブラリ。

LDIF

LDAP Data Interchange Format。

OID (オブジェクト識別子)

オブジェクトタイプを識別するドット区切り 10 進形式の文字列。

RFC

Request for Comment の略。IETF (インターネット技術標準化タスクフォース) が Web プロトコルの仕様を発表するメカニズムです。

TLS

Transport Layer Security (トランスポート層セキュリティ) の略で、ネットワーク接続で、暗号化され、認証された通信を実行するための一般的な業界標準。従来の SSL (Secure Socket Layer) メソッドに適用できます (ただしそれだけに限りません)。

X.500

ITU (国際電気通信連合) により発表されるドキュメントで、ディレクトリの概念の基礎となる根本的な概念が説明されています。通常、X.500 は「LDAP 以外のプロトコル」(DAP としても知られていません) と同義で使用されます。ただし、実際には、DAP プロトコルの仕様は X.519 で指定されており、ITU による完全なディレクトリの「仕様」は、10 を超える X.500 シリーズのドキュメントで配布されています。

エントリ

一般的には、ディレクトリ内のエントリは、データベースにおけるレコードまたは行に類似していません。Robert という名前を持つノードには、そのマネージャの名前、電子メール、インスタントメッセージ名などその人物に関する情報も含めることができます。ノード全体が「エントリ」です。

オブジェクト

オブジェクトクラスのインスタンスで、属性および値の集合 (次の「オブジェクトクラス」を参照)。

オブジェクトクラス

(ディレクトリスキーマで格納されている) オブジェクトの正式な定義。必須およびオプションの属性の数や種類、OID (オブジェクト識別子)、オブジェクトタイプ (抽象的、構造的、または補助的)、およびそのオブジェクトクラス名を含みます。

コンテナ

他のオブジェクトを格納できるディレクトリオブジェクト。

サブオーディネートエントリ

「コンテナオブジェクト」に含まれるオブジェクトまたはエントリ。

参照

名前解決のヒント。サーバはクライアントに参照を送信して、クライアントが現在のホストのローカルに存在しない情報を見つけられるようにします。参照をリダイレクトするかどうかは、クライアントが決定します。

識別名 (DN)

識別名は、ディレクトリ内のエントリを固有に認識する完全修飾された名前です。たとえば、Web サイトのユーザは、「cn=Theo87,ou=Visitors,o=Blogsville」という固有な DN を使用してディレクトリにアクセスできます。特定の DN を持つエントリは 1 つだけです (DN の順序および DN を読み取る方向が重要です)。DN は、左から右へ「リーフ」または末端として解析されます (この場合、cn=Theo87 が最初)。

スキーマ

LDAP ディレクトリのスキーマによりディレクトリに含まれる情報のレイアウトが提供され、その情報をグループ化する方法が指定されます。したがって、クライアントまたは外部のインターフェースはディレクトリの構造的特徴、および検索、追加、削除、変更などのためにツリーにアクセスする方法を認識することができます。LDAP のオブジェクトクラスおよび属性については、RFC 2256 を参照してください。

スコープ

操作が有効である範囲。LDAP の検索要求の場合、スコープはベース、最初のチャイルドレベル、またはサブツリーのいずれかです。検索のスコープがベースレベルである場合、ベース DN コンテナ内にあるエントリのみが検索されます。検索のスコープが最初のチャイルドレベルである場合、コンテナおよびその直下のチャイルドレベルで検索されます。「サブツリー」スコープではコンテナ、そのチャイルドオブジェクト、およびチャイルドのチャイルドすべて (末端のエントリまで) が検索されます。

相対識別名 (RDN)

RDN (Relative Distinguished Name) は、エンティティの完全修飾された DN の一部で、cn=Rich などのように、エントリの末端または「リーフノード」識別子を含みます (またはそれと同等です)。

属性

属性は値と関連付けられます。たとえば、cn (一般名) 属性に、Robert という値と関連付けることができます。ツリー内のオブジェクトは、属性および関連付けられた値の集合です。

チェーン

クライアントに対するプロキシとして動作するサーバによる名前の解決機能で、参照をリダイレクトすることによりローカル以外の DIT (ディレクトリ情報ツリー) のエントリを見つけます。このタイプの参照のリダイレクトは、クライアントは制御できません。

ディレクトリ情報ツリー (DIT)

ディレクトリの情報ツリー全体は、DIT (ディレクトリ情報ツリー) と呼ばれます。

匿名バインド

パスワードを使用せず (および通常はユーザ ID も使用せず) に確立されるディレクトリサーバへの接続。匿名バインドで許可される権利には、通常制限があります。

認証

通信への参加者の ID を検証するプロセス (この人物は実際にその人なのかどうか)。

バインド

一連のアカウント情報に基づいてディレクトリにアクセスすること (空白のアカウント情報でアクセスが許可される場合は、「匿名」バインドとといいます)。

非同期要求

即時の応答を予期しないで実行される要求。通常、非同期要求を実行するクライアントはサーバからの応答を待機せず、すぐに別の処理を開始します。この点で、クライアントが要求を実行し、ホストから応答を受信するまでブロックする同期要求と対照的です。

ベース DN

検索またはディレクトリへのアクセスの「開始ポイント」を指定する部分的に修飾されている名前 (またはコンテナコンテキスト)。

別名

別のディレクトリエントリを指名するディレクトリエントリ。

B

LDAP 結果コード

値	結果コード
0	SUCCESS (success)
1	OPERATIONS_ERROR (operationsError)
2	PROTOCOL_ERROR (protocolError)
3	TIME_LIMIT_EXCEEDED (timeLimitExceeded)
4	SIZE_LIMIT_EXCEEDED (sizeLimitExceeded)
5	COMPARE_FALSE (compareFalse)
6	COMPARE_TRUE (compareTrue)
7	AUTH_METHOD_NOT_SUPPORTED(authMethodNotSupported)
8	STRONG_AUTH_REQUIRED (strongAuthRequired)
10	REFERRAL (referral)
11	ADMIN_LIMIT_EXCEEDED (adminLimitExceeded)
12	UNAVAILABLE_CRITICAL_EXTENSION (unavailableCriticalExtension)
13	CONFIDENTIALITY_REQUIRED (confidentialityRequired)
14	SASL_BIND_IN_PROGRESS (saslBindInProgress)
16	NO_SUCH_ATTRIBUTE (noSuchAttribute)
17	UNDEFINED_ATTRIBUTE_TYPE (undefinedAttributeType)

値	結果コード
18	INAPPROPRIATE_MATCHING (inappropriateMatching)
19	CONSTRAINT_VIOLATION (constraintViolation)
20	ATTRIBUTE_OR_VALUE_EXISTS (AttributeOrValueExists)
21	INVALID_ATTRIBUTE_SYNTAX (invalidAttributeSyntax)
32	NO_SUCH_OBJECT (noSuchObject)
33	ALIAS_PROBLEM (aliasProblem)
34	INVALID_DN_SYNTAX (invalidDNSyntax)
35	IS_LEAF (isLeaf)
36	ALIAS_DEREFERENCING_PROBLEM (aliasDereferencingProblem)
48	INAPPROPRIATE_AUTHENTICATION (inappropriateAuthentication)
49	INVALID_CREDENTIALS (invalidCredentials)
50	INSUFFICIENT_ACCESS_RIGHTS (insufficientAccessRights)
51	BUSY (busy)
52	UNAVAILABLE (unavailable)
53	UNWILLING_TO_PERFORM (unwillingToPerform)
54	LOOP_DETECT (loopDetect)
64	NAMING_VIOLATION (namingViolation)
65	OBJECT_CLASS_VIOLATION (objectClassViolation)
66	NOT_ALLOWED_ON_NONLEAF (notAllowedOnNonLeaf)
67	NOT_ALLOWED_ON_RDN (notAllowedOnRDN)
68	ENTRY_ALREADY_EXISTS (entryAlreadyExists)
69	OBJECT_CLASS_MODS_PROHIBITED (objectClassModsProhibited)

値	結果コード
71	AFFECTS_MULTIPLE_DSAS (affectsMultipleDSAs)
80	OTHER (other)

ローカルエラー (サーバでの操作以外のアクションによるもの)

値	結果コード
81	SERVER_DOWN
82	LOCAL_ERROR
83	ENCODING_ERROR
84	DECODING_ERROR
85	LDAP_TIMEOUT
86	AUTH_UNKNOWN
87	FILTER_ERROR
88	USER_CANCELLED
90	NO_MEMORY
91	CONNECT_ERROR
92	LDAP_NOT_SUPPORTED
93	CONTROL_NOT_FOUND
94	NO_RESULTS_RETURNED
95	MORE_RESULTS_TO_RETURN
96	CLIENT_LOOP
97	REFERRAL_LIMIT_EXCEEDED
100	INVALID_RESPONSE
101	AMBIGUOUS_RESPONSE
112	TLS_NOT_SUPPORTED

索引

A

Abandon 操作 17
Add 要求 49

B

baseObject (範囲) 63

C

CA (認証局) 25
Compare 操作 17
Compare 要求 57
Composer (Enterprise) Server 9

D

DAP (Directory Access Protocol)
 定義 15
Decision アクション 79
 62
Delete 要求 59
Search 要求の 63
derefAlways 63
derefFindingBaseObject 63
derefInSearching 63
DN (識別名) 15
DOM
 データのマッピング 43
 リセット 76
DSE 71
DSML
 (Directory Services Markup Language)、定義 18
 LDIF との対照 81
DSML アクション
 Add 要求 49
 Compare 要求 57
 Delete 要求 59

Execute DSML 66
Modify 要求 60
Rename 要求 61
Search 要求 62

DSML アクションの作成 49
DSML アクションのバッチ処理 48

E

ECMAScript 81
ECMAScript 式 30
ECMAScript を使用する例 81
errorMessage 55
Execute DSML アクション 66
Expression Builder 77

F

Function、コンポーネントエディタ 68

I

IP アドレス 23, 27, 32
IP アドレスの形式 32

J

JLDAP 47, 81

L

LDAP
 データモデル 15
 動詞 17
ldap
 URL プロトコル 56
LDAP (Lightweight Directory Access Protocol)
 定義 15
LDAP コンポーネント、作成 34
LDIF 81
Log アクション 79

Log、コンポーネントエディタ 68

M

Map アクション 43
Map、コンポーネントエディタ 68
Modify 操作 16
Modify 要求 60
MSIE (Microsoft Internet Explorer) 56

N

neverDerefAliases 63
NO_SUCH_ATTRIBUTE 91
Novell パブリックサーバ 71

O

objectClass 72

P

23, 24, 27, 32

R

44
コマンド 44
44
コマンド 44
Rename 要求 61
50
resultCode、XPath 77

S

40
タブ 40
Search 操作 16
Search 要求 62
63, 64
singleLevel (範囲) 63
Search 要求の 64

SSL 証明書 25

Switch、コンポーネントエディタ 68

T

27, 64
Search 要求 64
接続リソース 27
TIME_LIMIT_EXCEEDED 91
TLS 27
暗号化および認証の有効化 24
接続リソース 30
TLS (Transport Layer Security) 19
39
タブ 39

U

Unbind 操作 16
URL、ldap 56

W

wholeSubtree 63

X

X.509 25
XCCERTFILE 30
xconfig.xml 30
xcrootca.jar 25, 27, 30
69
XML データマッピング 43
XML テンプレート、作成 36
XML ドキュメントの再ロード 76
XML の交換 69
XML の変換 69
XSL 変換 69

あ

アクション 45
DSML アクションの作成 49
Execute DSML 66

Map 43
アクションモデル、定義 45
アドレス帳の例 56

え

エラートラップ 69
エラー、処理 77
エントリ 13
 コンテナオブジェクト 13
 ネスト 14
エントリの移動 61

お

オブジェクト 13

か

間接化機能 17

く

クエリ
 LDAP URL 56
 移動 61
 検索 62
 削除 59
 追加 49
 名前の変更 61
 バッチ処理された要求 48
 比較 57
 変更 60
クライアント認証 19
グループの作成および命名 70

こ

構文、フィルタ 65
コロ記号 32
コンテナオブジェクト 13
コンポーネント
 作成 34
コンポーネントエディタ

概要 38
スキーマビュー 40
ツリービュー 39
コンポーネントのトランザクション 69

さ

サイレントフェイルオーバー 31
サブオーディネートエントリ 14
サブツリー検索 63

し

式駆動型の接続パラメータ 22
証明書 25

せ

セキュリティ
 SSL および TLS 19
 認証 34
接続とバインド 30
接続パラメータ
 式駆動型 22
 定数ベース 22
接続リソース
 サイレントフェイルオーバー 31
 作成 22, 25
 テスト 28
 トラブルシューティング 28
 編集 32

そ

属性 13
 値の追加、削除、変更、および比較 16
 単一値と複数值 13

て

定数ベースの接続パラメータ 22
ディレクトリ 12
クエリ 12
スキーマ 13

定義 12
ディレクトリ情報ツリー 14
データマッピング (DOM から DOM) 43
テストサーバ、Novell 71
デバッグ 28
テンプレート、作成 36

と

動詞 17
匿名バインド 29, 72
ドラッグアンドドロップによるマッピング 43
トラブルシューティング、接続リソース 28

に

認証 19, 34

ね

ネイティブ環境ペイン 39

は

バインド
Bind 操作 16
LDAP サーバへの接続 34
バインドと接続 30
パブリック LDAP サーバ 71
範囲の制限 63

ふ

フィルタ 65
フェイルオーバ 31
複数の要求 48
複製 31

へ

ベース DN 51

ほ

ポート番号 32

ゆ

ユーザ DN
(識別名)、定義 23

よ

要求のバッチ処理 48

る

ルートクエリ 71
ルール、フィルタ 65

れ

連動する要求 48

ろ

ログ出力 77