FileMaker® Server 11

カスタム Web 公開
with XML and XSLT

FileMaker, Inc.
5201 Patrick Henry Drive
Santa Clara, California 95054

詳細情報については www.filemaker.co.jp をご覧ください。

第 01 版

目次

第 4 章
カスタム Web 公開 with XSLT の概要

第 5 章
Web 公開エンジンを使用した XML データへのアクセス

第 6 章
FileMaker XSLT スタイルシートの開発

第 7 章
サイトのステージング、テスト、および監視

付録 A
クエリー文字列で使用される有効な名前

はじめに

このガイドについて

このガイドでは、XML と XSLT、Web サイトの開発、および FileMaker Pro を使用したデータベースの作成の経験があることを想定しています。データベースの設計の基礎、ならびにフィールド、リレーションシップ、レイアウト、ポータル、およびオブジェクトについてご理解いただく必要があります。このガイドでは、FileMaker Server 上でのカスタム Web 公開 with XML and XSLT に関する次の情報を説明します。

- XML または XSLT を使用してカスタム Web 公開ソリューションを開発するための必要条件
- XML または XSLT を使用してデータベースを公開する方法
- Web ユーザがカスタム Web 公開ソリューションにアクセスするための必要条件
- FileMaker Server でホストされているデータベースから XML データを取得する方法
- FileMaker XSLT スタイルシートを開発する方法

重要　FileMaker に関するドキュメントは、www.filemaker.co.jp からダウンロードすることができます。このドキュメントの最新版も、Web サイトから入手できます。

FileMaker Server のドキュメントには、次の情報が含まれます。

必要な情報	参照先
FileMaker Server のインストールと設定	『FileMaker Server 入門ガイド』 「FileMaker Server ヘルプ」
インスタント Web 公開	『FileMaker インスタント Web 公開ガイド』
カスタム Web 公開 with PHP	『FileMaker Server カスタム Web 公開 with PHP』
PHP Site Assistant の使用	「PHP Site Assistant ヘルプ」
カスタム Web 公開 with XML and XSLT	『FileMaker Server カスタム Web 公開 with XML and XSLT』 （このマニュアル）
XSLT Site Assistant の使用	「XSLT Site Assistant ヘルプ」
ODBC および JDBC ドライバのインストールと設定、ならびに ODBC および JDBC の使用	『FileMaker ODBC と JDBC ガイド』
FileMaker Server Auto Update で最新のプラグインを FileMaker Pro データベースクライアントコンピュータにダウンロードする方法	『FileMaker Server プラグインの更新ガイド』

第 1 章
カスタム Web 公開の概要

FileMaker Server では、次の方法で FileMaker データベースをインターネットまたはイントラネット上に公開できます。

インスタント Web 公開：インスタント Web 公開を使うと、データベースをすばやく簡単に Web 上で公開することができます。互換性のある Web ブラウザソフトウェアを所有し、インターネットまたはイントラネットにアクセス可能な Web ユーザは、データベースファイルを変更したり、他のソフトウェアをインストールしなくても、Web 上で公開されたデータベースの表示、編集、ソート、および検索を行うことができます。ただし、その場合にはこれらの操作を行うためのアクセス権が必要となります。

インスタント Web 公開では、ホストコンピュータによって FileMaker Pro または FileMaker Server が実行されている必要があります。ユーザインターフェースは、FileMaker Pro デスクトップアプリケーションに似ています。Web ユーザが操作する Web ページおよびフォームは、FileMaker Pro データベースで定義されたレイアウトおよび表示形式によって変わります。詳細については、『FileMaker インスタント Web 公開ガイド』を参照してください。

静的な公開：データがあまり変更されない場合、または稼動中のデータベースにユーザが接続しないようにする場合には、静的な公開方法を使用します。静的な公開方法では、FileMaker Pro のデータをエクスポートして Web ページを作成します。静的な公開方法では、FileMaker Pro データベースからデータをエクスポートして Web ページを作成します。Web ページは、HTML を使用してさらにカスタマイズすることができます。データベースの内容を変更しても、Web ページのデータは変更されません。ユーザは、Web サイトに接続してもデータベースには直接接続しません（インスタント Web 公開では、Web ブラウザが FileMaker Server に情報更新の要求を行うたびに、Web ブラウザのウインドウに表示されているデータが更新されます）。詳細については、『FileMaker インスタント Web 公開ガイド』を参照してください。

カスタム Web 公開：公開されるデータベースの表示方法と機能をさらに拡張する場合は、FileMaker Server に含まれるカスタム Web 公開テクノロジーを利用してカスタム Web を作成してください。公開されるデータベースは FileMaker Server でホストされ、カスタム Web 公開を利用可能にするために FileMaker Pro がインストールまたは実行されている必要はありません。

カスタム Web 公開では、次の操作を行うことができます。

- データベースを他の Web サイトに統合する
- ユーザによるデータの操作方法を決定する
- Web ブラウザでのデータの表示方法を制御する

FileMaker Server には、次の 2 つのカスタム Web 公開テクノロジーが備わっています。

- カスタム Web 公開 with PHP：FileMaker Pro データベースへのオブジェクト指向 PHP インターフェースを提供する PHP 用 FileMaker API を使用して、その FileMaker データを PHP Web アプリケーションに統合することができます。PHP Site Assistant を使用すると、完全な PHP Web サイトを生成したり、ユーザ側で PHP Web ページをコード化することができます。
- カスタム Web 公開 with XML and XSLT：
  - XML データ公開を使用して、FileMaker データを他の Web サイトやアプリケーションと交換できます。
  - サーバーによって処理される XSLT スタイルシートを使用すると、FileMaker データのサブセットを、他の Web サイトに統合したり、他のミドルウェアやカスタムアプリケーションに統合することができます。XSLT Site Assistant を使用すると、XSLT スタイルシートを生成したり、ユーザ側でスタイルシートをコード化することができます。

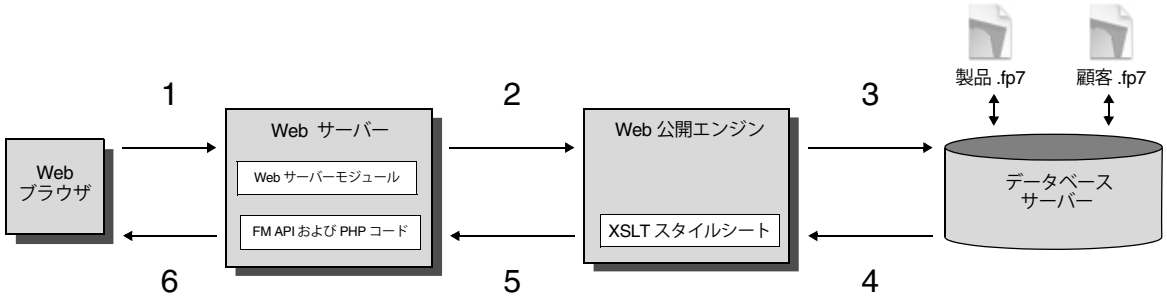
Web 公開エンジンについて

インスタント Web 公開およびカスタム Web 公開をサポートするため、FileMaker Server では、FileMaker Server Web 公開エンジンと呼ばれるソフトウェアコンポーネントのセットが使用されます。Web 公開エンジンは、Web ユーザのブラウザ、Web サーバー、および FileMaker Server の間の通信を処理します。

カスタム Web 公開 with XML and XSLT：Web 公開エンジンは XSLT プロセッサとして機能し、出力を HTML、XML、またはテキスト（vCard など）として Web サーバーに提供します。Web 公開エンジンからの出力は、Web サーバーによって Web ブラウザに提供されます。Web ユーザがカスタム Web 公開ソリューションにアクセスするには、HREF リンクをクリックするか、または Web サーバーのアドレスと FileMaker クエリー文字列リクエストを指定した URL（Uniform Resource Locator）を入力します。この URL で、XML データにアクセスするか、または XSLT スタイルシートを参照できます。Web 公開エンジンは、クエリー文字列リクエストで指定された XML データ、または参照されている XSLT スタイルシートの結果を返します。

カスタム Web 公開 with PHP：Web ユーザがカスタム Web 公開ソリューションにアクセスしている場合、FileMaker Server 上の PHP が Web 公開エンジンに接続し、FileMaker API for PHP を介して応答します。

カスタム Web 公開のための FileMaker Server Web 公開エンジンの使用



Web 公開エンジンのリクエストの処理

1. リクエストが、Web ブラウザまたはアプリケーションから Web サーバーに送信されます。
2. Web サーバーが、FileMaker の Web サーバーモジュールを介して、リクエストを Web 公開エンジンにルーティングします。
3. Web 公開エンジンが、データベースサーバーでホストされているデータベースにデータを要求します。
4. FileMaker Server が、要求された FileMaker データを Web 公開エンジンに送信します。
5. Web 公開エンジンが、FileMaker データを変換してリクエストへの応答を行います。
   - PHP リクエストの場合、Web 公開エンジンは API リクエストに応答します。
   - XML リクエストの場合、Web 公開エンジンは Web サーバーに XML データを直接送信します。
   - XSLT リクエストの場合、Web 公開エンジンは、XSLT スタイルシートを使用して XML データを書式設定または変換し、Web サーバーへの出力を HTML ページ、XML ドキュメント、またはテキストとして生成します。
6. Web サーバーが、Web ブラウザまたはプログラムに出力を送信します。

重要　Web 上にデータを公開する場合は、セキュリティが重要になります。『FileMaker セキュリティガイド』のセキュリティガイドラインを参照してください。このマニュアルは、PDF 形式で www.filemaker.co.jp から入手することができます。

カスタム Web 公開 with PHP

FileMaker API for PHP には、FileMaker データベースへのオブジェクト指向 PHP インターフェースが備わっています。FileMaker API for PHP を使用すると、FileMaker Pro データベースに保存されているロジックおよびデータの両方に対し、Web 上にアクセスして公開したり、他のアプリケーションにエクスポートすることができます。また、API は、FileMaker Pro データベースに保存されているデータの抽出やフィルタを行うために、複雑で複合の検索コマンドをサポートしています。

PHP は元々、手続き型プログラミング言語として設計されており、オブジェクト指向の Web 開発言語として強化されています。PHP には、サイトのページ内でのロジックのタイプのほぼ任意を構築するためのプログラミング言語機能が備わっています。たとえば、条件付きロジック構築を使用して、ページ生成やデータ ルーティング、ワークフローを制御することができます。また、PHP はサイト管理とセキュリティも提供します。

さらに、FileMaker PHP Site Assistant を使用すると、FileMaker Pro データベースにあるデータに正しくアクセスするために必要な前提条件と機能のすべてが含まれる PHP コードを作成することができます。PHP Site Assistant は、Web ユーザがデータベースの検索、レコードの一覧の表示、レコードのブラウズ、レコードのソート、レコードの追加、レコードの編集、レコードの複製、レコードの削除、および集計レポートの表示を行うことができるように、複数ページの Web サイトを生成します。PHP の経験がほとんどない FileMaker 開発者でも、PHP Site Assistant を使用すると、完全な PHP Web サイトを生成することができます。FileMaker の経験がほとんどない PHP 開発者でも、PHP Site Assistant を使用すると、FileMaker API for PHP のオブジェクトとメソッドを理解することができます。

カスタム Web 公開 with XML and XSLT

XML を使用した FileMaker カスタム Web 公開では、FileMaker Server によってホストされている FileMaker Pro データベースに対してクエリーリクエスト送信して、結果のデータの表示、変更、または操作を行うことができます。適切なクエリーコマンドと引数を指定した HTTP リクエストを使用して、FileMaker データを XML ドキュメントとして取得してから、XML データを他のアプリケーションにエクスポートしたり、XML データに XSLT スタイルシートを適用することができます。

XSLT を使用した FileMaker カスタム Web 公開では、Web ブラウザや他のアプリケーション用に XML データを変換、フィルタ、または書式設定できます。次の操作を行うことができます。

- XSLT スタイルシートを使用して、他のアプリケーションやデータベースにて FileMaker XML 文法と他の XML 文法の間でデータを変換する
- データベースのどのフィールドを公開するかをスタイルシートで制御することによってデータをフィルタする
- Web ページにデータを表示する書式を設定したり、Web ユーザがデータを操作する方法を制御する

Web ユーザがいずれかの XSLT スタイルシートを参照する HTTP リクエストと URL を送信すると、Web 公開エンジンは、スタイルシートを使用して FileMaker データベースからデータを取得します。Web 公開エンジンは、スタイルシートを使用して XML データの変換と書式設定を行い、Web ユーザが操作できる最終的な HTML ページを生成します。

また、FileMaker XSLT Site Assistant を使用すると、カスタム Web 公開 XSLT の起点となる基本的な XSLT スタイルシートを作成できます。XSLT Site Assistant は、データベースの検索、一度での 1 レコードのブラウズ、データベース内レコードの一覧表示、レコードのソート、レコードの追加、レコードの編集、レコードの複製、レコードの削除、および集計レポートの表示を行うページ用のスタイルシートを生成します。

PHP と XML および XSLT との比較

以降のセクションでは、ユーザのサイトに最適なソリューションを決定するためのガイドラインの一部について説明します。

PHP を選択する理由

- PHP はオブジェクト指向手続き型スクリプト言語として優れていますが、学習は比較的容易です。トレーニング、開発、およびサポート用に数多くのリソースを使用できます。
- FileMaker API for PHP を使用すると、FileMaker Pro データベースに保存されているロジックおよびデータに対し、Web 上にアクセスして公開したり、他のアプリケーションにエクスポートすることができます。
- PHP では、条件付きロジックを使用して、ページ構築やフローを制御することができます。
- PHP には、サイトのページ上でさまざまなタイプのロジックを構築するためのプログラミング言語機能が備わっています。
- PHP は、最も知られている Web スクリプト言語の 1 つです。
- PHP はオープンソースの言語であり、http://php.net から使用できます。
- PHP を使用すると、さまざまな種類のサードパーティ製コンポーネントにアクセスして、ユーザのソリューションを統合することができます。

メモ カスタム Web 公開 with PHP の詳細については、「FileMaker Server カスタム Web 公開 with PHP」を参照してください。

XML および XSLT を選択する理由

- FileMaker XML リクエスト引数構文は、データベース操作用に設計され、ソリューション開発を簡略化します。
- XML および XSLT は、W3C スタンダードです。
- XML は、Unicode をサポートするコンピュータおよび人間が読み込み可能な形式であり、書き込まれた任意の言語でのデータ通信を可能にします。
- XML は、レコード、一覧、およびツリー構造データの表示に適しています。
- XSLT を使用すると、XML 出力を RSS、RTF、および vCard などの構造化されたテキストドキュメントに変換できます。
- XSLT を使用すると、XML 出力の文法を変換できます。
- テンプレートを使用することで、変数データに条件付き形式を簡単に適用できます。
- カスタム Web 公開および FileMaker Pro データベースからの XML エクスポートには、FMPXMLRESULT ベースのスタイルシートを使用できます。
- FileMaker Server では、FileMaker XSLT スタイルシート処理を行うことで、クライアントサイドの XSLT スタイルシートの使用によって無防備になりうるデータへの不正アクセスを防止します。

第 2 章
カスタム Web 公開 with XML and XSLT について

Web 公開エンジンを使用した動的な Web サイトの作成

Web 公開エンジンは、XML データの公開およびサーバーによって処理される XSLT スタイルシートを使用して、FileMaker Server にカスタム Web 公開機能を提供します。カスタム Web 公開には、次のような多くの利点があります。

- カスタマイズ：Web ユーザが FileMaker データを操作する方法や、Web ブラウザにデータを表示する方法を決定できます。
- データの交換：FileMaker XML を使用することで、FileMaker データを他の Web サイトやアプリケーションと交換できます。
- データの統合：FileMaker XSLT スタイルシートを使用することで、FileMaker データを、他の Web サイトに統合したり、他のミドルウェアやカスタムアプリケーションに統合することができます。Web ブラウザに FileMaker のレイアウト全体を表示する代わりに、データが別の Web サイトに属するかのように表示できます。
- セキュリティ：FileMaker Server の管理者である場合には、サーバーでホストされているすべてのデータベースに対して、インスタント Web 公開、XML を使用した Web 公開、または XSLT を使用した Web 公開を個別に有効または無効にすることができます。FileMaker データベースの所有者である場合は、各データベースに対して、インスタント Web 公開、XML を使用した Web 公開、または XSLT を使用した Web 公開への Web ユーザアクセスを制御できます。
- サーバーサイドのスタイルシート：クライアントサイドのスタイルシートでは、機密性の高いデータベースの情報が不正に検証される可能性がありますが、XSLT スタイルシートはサーバーサイドで処理されるため、これを防ぐことができます。
- 公開されるデータの制御とフィルタ：XSLT スタイルシートを使用することで、公開するデータベース情報のデータやタイプを制御およびフィルタして、データベースの不正使用を防止できます。また、データベース名やフィールド名などのメタデータを隠すこともできます。
- オープンスタンダードへの準拠：カスタム Web 公開ソリューションに対しては、ツール、リソース、および熟練した技術者がより豊富に揃っています。標準的な XML または XSLT の知識がある場合、使用する URL 構文やクエリー引数など、カスタム Web 公開 with XML に特有の詳細事項をいくつか学べば、すぐにソリューションの開発に取りかかることができます。

カスタム Web 公開 with XML について

カスタム Web 公開 with XML では、FileMaker データベースからデータを取得して、そのデータを別の出力形式で簡単に使用できます。適切なクエリーコマンドと引数を指定した HTTP リクエストを使用して、FileMaker データを XML ドキュメントとして取得してから、XML データを他のアプリケーションで使用したり、XML データに XSLT スタイルシートを適用できます。第 5 章「Web 公開エンジンを使用した XML データへのアクセス」を参照してください。

カスタム Web 公開 with XSLT について

カスタム Web 公開 with XSLT では、XML データを変換、フィルタ、または書式設定して、Web ブラウザや他のアプリケーションで使用できます。XSLT スタイルシートを使用して、FileMaker XML 文法と他の XML 文法の間でデータを変換して、他のアプリケーションやデータベースで使用できます。データベースのどのフィールドを公開するかをスタイルシートで制御することにより、データをフィルタできます。また、Web ページにデータを表示する書式を設定したり、Web ユーザがデータを操作する方法を制御できます。第 4 章「カスタム Web 公開 with XSLT の概要」を参照してください。

Web ユーザがいずれかの XSLT スタイルシートを参照する HTTP リクエストと URL を送信すると、Web 公開エンジンは、スタイルシートを使用して FileMaker データベースから動的にデータを取得します。Web 公開エンジンは、スタイルシートを使用して XML データの変換と書式設定を行い、Web ユーザが操作できる最終的な HTML ページを生成します。

XML および XSLT を使用した FileMaker Server カスタム Web 公開の使用の詳細については、www.filemaker.co.jp にアクセスしてください。

XSLT スタイルシートの作成について

FileMaker Server には、XSLT スタイルシートを開発するためのツールが付属します。FileMaker XSLT Site Assistant は、XSLT を使用した FileMaker カスタム Web 公開の起点となる基本的な XSLT スタイルシートを作成する場合に使用できるアプリケーションです。XSLT Site Assistant は、FileMaker XSLT スタイルシートの構築の学習に効果的です。必要に応じて、好みの XSLT スタイルシートオーサリングツールを使用してスタイルシートを変更することができます。27 ページの「FileMaker XSLT Site Assistant を使用した FileMaker XSLT スタイルシートの生成」を参照してください。

メモ FileMaker Server は、World Wide Web Consortium の定義どおりに XSLT 1.0 をサポートします。使用する任意の XSLT オーサリングツールは、スタンダード準拠の XSLT 1.0 を作成する必要があります。

カスタム Web 公開 with XML and XSLT の主な機能

XML および XSLT を使用した FileMaker Server カスタム Web 公開では、多くの重要な機能が提供されています。

- データベースは FileMaker Server 上でホストされ、FileMaker Pro が実行されている必要はありません。
- XSLT スタイルシートをサーバーサイドで処理できます。これは、クライアントサイドでスタイルシートを処理するよりも安全です。
- XSLT スタイルシートで、JavaScript のサーバーサイドでの処理を使用することができます。詳細については、68 ページの「スクリプト言語のサーバーサイドでの処理の使用」を参照してください。
- FileMaker XSLT スタイルシートを使用して、XML データの要求時に使用するクエリーコマンド、引数、および値を静的に定義しておくことで、クエリーコマンドとクエリー引数の不正使用を防止できます。51 ページの「静的に定義されたクエリーコマンドとクエリー引数の使用」を参照してください。
- FileMaker Pro と同様に、データ、レイアウト、およびフィールドへのアクセスは、データベースのアクセス権で定義されているユーザのアカウント設定に基づきます。また、Web 公開エンジンでは、他のセキュリティの強化点もいくつかサポートされています。20 ページの「公開されたデータベースの保護」を参照してください。
- Web ユーザが、複数のステップを使用した複雑なスクリプトを実行することができます。FileMaker は 75 以上のスクリプトステップをカスタム Web 公開でサポートしています。21 ページの「FileMaker スクリプトとカスタム Web 公開」を参照してください。
- FileMaker スクリプトには、引数値を渡すことができます。詳細については、88 ページの「-script.param（スクリプトに引数を渡す）クエリー引数」、89 ページの「-script.prefind.param （検索前にスクリプトに引数を渡す）クエリー引数」および 89 ページの「-script.presort.param （ソート前にスクリプトに引数を渡す）クエリー引数」を参照してください。
- fmresultset XML 文法では、名前でフィールドにアクセスして、relatedset（ポータル）のデータを操作できます。
- XSLT スタイルシートでセッション関数を使用して、サーバーによって維持されるセッションに Web ユーザの情報および処理を保存できます。
- データベースのデータにアクセスするには、レイアウトを指定する必要があります。付録 A「クエリー文字列で使用される有効な名前」を参照してください。
- 各 Web ユーザは、セッションがアクティブである限り維持される固有のグローバルフィールド値を使用できます。グローバルフィールドの詳細については、「FileMaker Pro ヘルプ」を参照してください。カスタム Web 公開でのグローバルフィールドの使用の詳細については、80 ページの「グローバルフィールドを指定するための構文について」を参照してください。

Web 上でデータベースを公開する場合の必要条件

カスタム Web 公開を使用してデータベースを公開するための必要条件

カスタム Web 公開 with XML and XSLT を使用してデータベースを公開するための必要条件は、次のとおりです。

- 次を含む FileMaker Server 展開
  - Microsoft IIS（Windows）または Apache（Mac OS X）のいずれかの Web サーバー
  - カスタム Web 公開用に有効化された FileMaker データベースサーバー
  - インストールおよび設定されている Web 公開エンジン
- FileMaker Server でホストされている 1 つまたは複数の FileMaker Pro データベース
- Web サーバーが実行されているホストの IP アドレスまたはドメイン名
- カスタム Web 公開ソリューションを開発およびテストするための Web ブラウザと Web サーバーへのアクセス

詳細については、『FileMaker Server 入門ガイド』を参照してください。

Web ユーザがカスタム Web 公開ソリューションにアクセスするための必要条件

Web ユーザが カスタム Web 公開 with XML and XSLT ソリューションにアクセスするための必要条件は、次のとおりです。

- Web ブラウザ
- インターネットまたはイントラネット、および Web サーバーへのアクセス
- Web サーバーが実行されているホストの IP アドレスまたはドメイン名

データベースがパスワードで保護されている場合は、データベースアカウントのユーザ名とパスワードの入力

インターネットまたはイントラネットへの接続

インターネットまたはイントラネット上でデータベースを公開する場合、ホストコンピュータで FileMaker Server を起動し、共有するデータベースをホストして利用可能にする必要があります。また、次の点にも注意してください。

- データベースは、インターネットまたはイントラネットへの常時接続を確保したコンピュータで公開してください。インターネットに常時接続していなくても Web 上でデータベースを公開することは可能ですが、Web ユーザはホストするコンピュータがインターネットまたはイントラネットに接続している場合にのみデータベースにアクセスすることができます。
- FileMaker Server 展開の一部である Web サーバー用のホストコンピュータには、静的、つまり固有な専用の IP アドレスまたはドメイン名が設定されている必要があります。ISP（インターネットサービスプロバイダ）に接続してインターネットを使用する場合、IP アドレスは動的に割り当てられる可能性があります。つまり、接続するたびに IP アドレスが変更されることになります。動的な IP アドレスでは、データベースの検索が困難になります。使用できるインターネットへのアクセスの種類がわからない場合は、ISP またはネットワーク管理者にお問い合わせください。

この後の作業を開始するにあたって

ここでは、Web 公開ソリューションの展開を開始するための推奨事項の一部を挙げます。

- FileMaker Server Admin Console を使用してカスタム Web 公開を有効にしていない場合は、有効にします。「FileMaker Server ヘルプ」と『FileMaker Server 入門ガイド』を参照してください。
- 公開する各 FileMaker データベースを FileMaker Pro で開き、データベースで、カスタム Web 公開に対して適切な拡張アクセス権が有効になっていることを確認します。19 ページの「データベースでのカスタム Web 公開の有効化」を参照してください。
- XML を使用して FileMaker データベースのデータにアクセスする方法を学ぶには、第 5 章「Web 公開エンジンを使用した XML データへのアクセス」を参照してください。
- FileMaker XSLT スタイルシートの開発を開始する方法を学ぶには、第 4 章「カスタム Web 公開 with XSLT の概要」を参照してください。

第 3 章
データベースのカスタム Web 公開の準備

データベースでカスタム Web 公開を使用する前に、データベースを準備して不正アクセスから保護する必要があります。

データベースでのカスタム Web 公開の有効化

公開する各データベースでカスタム Web 公開を有効にする必要があります。各データベースで、カスタム Web 公開 with XML またはカスタム Web 公開 with XSLT のいずれかを個別に有効にするか、あるいは両方のテクノロジーを有効にできます。データベースでこれらのテクノロジーの一方または両方を有効にしないと、Web 公開エンジンをサポートするように設定されている FileMaker Server でデータベースがホストされていても、Web ユーザがカスタム Web 公開を使用してデータベースにアクセスすることはできません。

データベースに対してカスタム Web 公開を有効にするには、次の操作を行います。

1. FileMaker Pro で、[完全アクセス] アクセス権セットが割り当てられているアカウントを使用して、公開するデータベースを開きます。または、[拡張アクセス権の管理] アクセス権が割り当てられているアカウントを使用してデータベースを開くこともできます。
2. これらの拡張アクセス権の 1 つまたは両方を、1 つまたは複数のアクセス権セットに割り当てます。
   - カスタム Web 公開 with XML を許可するには、キーワード“fmxml”を使用します。
   - カスタム Web 公開 with XSLT を許可するには、キーワード“fmxslt”を使用します。

   FileMaker Pro 8 からは、キーワード“fmxml”および“fmxslt”は [拡張アクセス権] タブで定義します。
3. 1 つまたは複数のアカウント、あるいは Admin またはゲストアカウントに、カスタム Web 公開拡張アクセス権が含まれるアクセス権セットを割り当てます。

メモ　カスタム Web 公開ソリューション用のアカウント名とパスワードを定義する場合は、表示可能な ASCII 文字（a から z、A から Z、および 0 から 9 など）を使用します。アカウント名とパスワードのセキュリティを高めるには、「!」や「%」などの記号を含めます。ただし、コロンは含めないでください。アカウントの設定の詳細については、「FileMaker Pro ヘルプ」を参照してください。

保護されたデータベースへのアクセス

カスタム Web 公開ソリューションを使用してデータベースにアクセスする場合、Web ユーザに対して、アカウント情報を入力するようメッセージが表示される場合があります。データベースのゲストアカウントが無効になっているか、またはカスタム Web 公開拡張アクセス権が含まれるアクセス権セットが割り当てられていない場合、Web 公開エンジンは、HTTP 基本認証を使用して Web ユーザに認証を要求します。Web ユーザのブラウザによって、カスタム Web 公開拡張アクセス権が割り当てられているアカウントのユーザ名とパスワードをユーザが入力するための HTTP 基本認証のダイアログボックスが表示されます。

次に、Web ユーザがカスタム Web 公開ソリューションを使用してデータベースにアクセスする場合の処理の概要を説明します。

- アカウントにパスワードが割り当てられていない場合、Web ユーザはアカウント名のみを入力します。
- ゲストアカウントが無効な場合、データベースにアクセスするときに、アカウント名とパスワードを入力するようメッセージが表示されます。入力するアカウントでは、カスタム Web 公開拡張アクセス権が有効になっている必要があります。

- ゲストアカウントが有効で、カスタム Web 公開拡張アクセス権が含まれるアクセス権セットが有効な場合、すべての Web ユーザは、自動的に、ゲストアカウントに割り当てられているアクセス権でデータベースを開きます。ゲストアカウントにカスタム Web 公開拡張アクセス権が割り当てられている場合、次のように処理されます。
  - ファイルを開くときに、アカウント名とパスワードを入力するようメッセージは表示されません。
  - すべての Web ユーザは自動的にゲストアカウントでログインし、ゲストアカウントのアクセス権を持ちます。[再ログイン] スクリプトステップを使用すると、ユーザは Web ブラウザからログインアカウントを変更することができます。たとえば、ゲストアカウントから、より多くの機能を使用できる別のアカウントに切り替えることができます。
  - ゲストアカウントのデフォルトのアクセス権セットは、「閲覧のみ」アクセスを提供します。このアカウントのデフォルトのアクセス権（拡張アクセス権を含む）を変更できます。「FileMaker Pro ヘルプ」を参照してください。

メモ デフォルトでは、Web ユーザが Web ブラウザからアカウントのパスワードを変更することはできません。[パスワード変更] スクリプトステップで使用すると、この機能をデータベースに組み込み、Web ユーザがブラウザからパスワードを変更できるようにすることができます。「FileMaker Pro ヘルプ」を参照してください。

公開されたデータベースの保護

カスタム Web 公開 with XML and XSLT を使用する場合、公開されたデータベースにアクセス可能なユーザを制限できます。

- カスタム Web 公開に使用されるデータベースアカウントにパスワードを割り当てます。
- カスタム Web 公開 with XML and XSLT は、公開されたデータベースへのアクセスを許可するアカウントのアクセス権セットでのみ有効にします。
- 個々のデータベースに対して特定のタイプのカスタム Web 公開テクノロジーを有効または無効にするには、拡張アクセス権を設定します。
- Web 公開エンジンで、FileMaker Server Admin Console を使用して、すべてのカスタム Web 公開ソリューションに対して特定のタイプのカスタム Web 公開テクノロジーを有効または無効にします。「FileMaker Server ヘルプ」を参照してください。
- Web 公開エンジンを使用してデータベースにアクセスできる IP アドレスを制限するように Web サーバーを設定します。たとえば、192.168.100.101 という IP アドレスの Web ユーザにのみデータベースへのアクセスを許可するように設定できます。IP アドレスの制限の詳細については、Web サーバーのマニュアルを参照してください。
- Web サーバーと Web ユーザのブラウザの間の通信に、SSL（Secure Sockets Layer）暗号化を使用します。SSL 暗号化は、「暗号」と呼ばれる数式を使用して、サーバーとクライアントの間で交換される情報を判読不可能な情報に変換します。これらの暗号を使用して、「暗号鍵」によって情報を判読可能なデータに再変換します。SSL の有効化と設定の詳細については、Web サーバーのマニュアルを参照してください。

さらに詳しい情報については、『FileMaker ユーザーズガイド』を参照してください。このマニュアルは、PDF 形式で www.filemaker.co.jp から入手することができます。

Web サーバーでのインターネットメディアタイプ（MIME のサポート)

インターネットに対して登録されている最新の MIME（Multipurpose Internet Mail Extensions）タイプがサポートされているかどうかは、Web サーバーによって判断されます。Web 公開エンジンによって、Web サーバーの MIME のサポートが変更されることはありません。詳細については、Web サーバーのマニュアルを参照してください。

Web 上でのオブジェクトフィールドの内容の公開について

イメージファイルなどのオブジェクトフィールドの内容は、FileMaker データベースの内部に保存するか、または相対パスを使用してファイル参照として保存できます。

メモ Web 公開エンジンでは、ムービーファイルのストリーミングはサポートされません。Web ユーザがムービーを表示するには、ムービーファイル全体をダウンロードする必要があります。

データベース内に保存されているオブジェクトフィールドのオブジェクトの公開

FileMaker データベースのオブジェクトフィールドに実際のファイルが保存されている場合は、データベースファイルが FileMaker Server 上で適切にホストされていてアクセス可能であれば、オブジェクトフィールドの内容を操作する必要はありません。34 ページの「XML ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について」および 49 ページの「XSLT ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について」を参照してください。

ファイル参照として保存されているオブジェクトフィールドのオブジェクトの公開

オブジェクトフィールドに、実際のファイルの代わりにファイル参照が保存されている場合、オブジェクトフィールドのオブジェクトを公開するには、次の操作を行います。

メモ すべての QuickTime ムービーは、参照としてオブジェクトフィールドに保存されます。

ファイル参照として保存されているオブジェクトフィールドのオブジェクトを公開するには、次の操作を行います。

1. オブジェクトファイルを、「FileMaker Pro」フォルダ内の「Web」フォルダに保存します。
2. FileMaker Pro で、オブジェクトフィールドにオブジェクトを挿入して、[ファイルの参照データのみ保存] オプションを選択します。
3. 「Web」フォルダ内の参照されているオブジェクトファイルを、Web サーバーソフトウェアのルートフォルダ内の同じ相対パスの場所にコピーまたは移動します。
   - IIS の場合は、「< ドライブ >:¥Inetpub¥wwwroot」にファイルを移動します。
   - Apache の場合は、「/ ライブラリ /WebServer/Documents」にファイルを移動します。

メモ ファイル参照として保存されているオブジェクトの場合、提供するファイルの種類（ムービーなど）の MIME タイプをサポートするように Web サーバーが設定されている必要があります。詳細については、Web サーバーのマニュアルを参照してください。

Web ユーザがオブジェクトフィールドのデータを表示する方法

Web 公開エンジンを使用して Web 上にデータベースを公開する場合、Web ユーザは、次のような限られた方法でオブジェクトフィールドのデータを操作することができます。

- オブジェクトフィールドのサウンドを再生したり、OLE オブジェクトを表示することはできません。
- Web ユーザがオブジェクトフィールドの内容を変更または追加することはできません。Web ユーザがオブジェクトフィールドを使用してデータをデータベースにアップロードすることはできません。
- GIF または JPEG 以外の形式のグラフィックがデータベースに格納されている場合は、Web ブラウザからそのグラフィックデータが要求されたときに、Web 公開エンジンによって一時的な JPEG イメージが作成されます。

FileMaker スクリプトとカスタム Web 公開

FileMaker Pro のスクリプトの管理機能を使用すると、頻繁に実行されるタスクの自動化や、複数のタスクの結合が可能となります。カスタム Web 公開とともに使用すると、Web ユーザは FileMaker スクリプトを使用して、より多くのタスクや一連のタスクを実行できます。

FileMaker は 75 以上のスクリプトステップをカスタム Web 公開でサポートしています。URL のクエリー文字列内、または XSLT スタイルシート内で <?xslt–cwp–query?> 処理命令を使用すると、Web ユーザは、自動化されたさまざまなタスクを実行できます。サポートされていないスクリプトステップを参照するには、FileMaker Pro の [スクリプトの編集] ウインドウで [互換性を表示] から [Web 公開] を選択します。グレー表示されるスクリプトステップは、Web 上ではサポートされません。スクリプトの作成の詳細については、「FileMaker Pro ヘルプ」を参照してください。

スクリプトのヒントと考慮事項

多くのスクリプトステップは Web 上でも同じように動作しますが、動作が異なるものもあります。23 ページの「カスタム Web 公開ソリューションでのスクリプト動作」を参照してください。データベースを共有する前に、Web ブラウザから実行されるスクリプトをすべて評価してください。また、異なるユーザアカウントでログインして、すべてのクライアントに対して正しく動作することを確認します。スクリプト関連のエラーについて、Web 公開エンジンのアプリケーションログファイル（pe_application_log.txt）を確認します。詳細については、76 ページの「Web 公開エンジンのアプリケーションログの使用」を参照してください。

次のヒントおよび考慮事項に注意してください。

- アカウントとアクセス権を使用して、Web ユーザが実行可能なスクリプトのセットを制限します。Web 互換のスクリプトステップのみがスクリプトに含まれることを確認し、Web ブラウザから使用する必要があるスクリプトへのアクセスのみを提供します。
- アクセス権によって制御されたステップの組み合わせを実行するスクリプトの影響を考慮します。たとえば、レコードを削除するステップがスクリプトに含まれていて、Web ユーザがレコードの削除を許可するアカウントでログインしていない場合、このスクリプトでは、[レコード削除] スクリプトステップは実行されません。ただし、スクリプトは引き続き実行される場合があり、予期しない結果になる可能性があります。
- [スクリプトの編集] ウインドウで [スクリプトを完全アクセス権で実行] を選択すると、個々のアクセスが付与されていないタスクをスクリプトで実行することができます。たとえば、アカウントとアクセス権を使用してユーザがレコードを削除できないようにしつつ、スクリプト内にあらかじめ定義された条件下で特定のタイプのレコードを削除するスクリプトの実行を許可することができます。
- サポートされていないステップ（Web 互換ではないステップなど）がスクリプトに含まれる場合は、[ユーザによる強制終了を許可] スクリプトステップを使用して、以降のステップの処理方法を決定します。
  - [ユーザによる強制終了を許可] スクリプトステップオプションが有効（オン）の場合、サポートされていないスクリプトステップが使用されていると、スクリプトの続行は停止されます。
  - [ユーザによる強制終了を許可] がオフの場合、サポートされていないスクリプトステップはスキップされ、スクリプトの実行が続行されます。
  - このスクリプトステップが含まれない場合、スクリプトは、この機能が有効な場合と同様に実行されるため、サポートされていないスクリプトステップが使用されていると、スクリプトは停止します。
- FileMaker Pro クライアントから 1 つのステップで動作する一部のスクリプトでは、追加の [レコード / 検索条件確定] ステップを使用して、データをホストに保存しなければならない場合があります。Web ユーザはホストと直接接続していないので、データが変更されたときに通知されません。たとえば、条件付き値一覧などの機能では、値一覧フィールドに結果を表示するにはデータをホストに保存する必要があるため、Web ユーザに対しては高速に応答しません。
- データ変更は、データをサーバーに保存する（送信する）までブラウザに表示されないので、データを変更するスクリプトにも [レコード / 検索条件確定] ステップを含める必要があります。これには、[切り取り]、[コピー]、[貼り付け] などのスクリプトステップが含まれます。単一ステップの処理の多くは、[レコード / 検索条件確定] ステップに変換する必要があります。Web サーバーから実行されるスクリプトを設計する際は、スクリプトの最後に [レコード / 検索条件確定] ステップを含めて、すべての変更が保存されるようにします。
- クライアントのタイプに基づく条件付きスクリプトを作成するには、Get(アプリケーションバージョン) 関数を使用します。返された値に「Web Publishing Engine」が含まれていれば、現在のユーザがカスタム Web 公開を使用してデータベースにアクセスしていることがわかります。関数の詳細については、「FileMaker Pro ヘルプ」を参照してください。
- XSLT スタイルシートで状態を設定または変更するスクリプトを使用する場合は、FileMaker Server Admin Console を使用して、Web 公開エンジンの XSLT データベースセッションオプションを有効にする必要があります。このオプションが有効でない場合、状態はリクエスト間で維持されません。「FileMaker Server ヘルプ」を参照してください。

カスタム Web 公開ソリューションでのスクリプト動作

次のスクリプトステップは、Web 上と FileMaker Pro で機能が異なります。すべてのスクリプトステップの詳細については、「FileMaker Pro ヘルプ」を参照してください。

スクリプトステップ	カスタム Web 公開ソリューションでの動作
スクリプト実行	ファイルが FileMaker Server 上でホストされていて、他のファイルでカスタム Web 効果が有効になっていない限り、スクリプトを他のファイルで実行することはできません。
アプリケーションを終了	Web ユーザをログオフしてウインドウを閉じますが、Web ブラウザアプリケーションは終了しません。
ユーザによる強制終了を許可	サポートされていないスクリプトステップの処理方法を決定します。スクリプトの続行を中止する場合は有効にし、サポートされていないステップをスキップする場合は無効にします。詳細については、22 ページの「スクリプトのヒントと考慮事項」を参照してください。 Web ユーザはカスタム Web 公開スクリプトを強制終了できませんが、このオプションを使用する場合、サポートされていないスクリプトステップが使用されていると、スクリプトの続行は停止されます。
エラー処理	カスタム Web 公開では常に有効です。Web ユーザはカスタム Web 公開スクリプトを強制終了することはできません。
スクリプト一時停止 / 続行	このスクリプトステップはカスタム Web 公開でサポートされていますが、使用しないでください。一時停止を行うステップが実行されると、スクリプトが一時停止します。一時停止したスクリプトの実行は、続行スクリプトステップが含まれるスクリプトでのみ続行できます。セッションがタイムアウトするまでスクリプトが一時停止された状態のままになっている場合、スクリプトは完了しません。
レコードのソート	カスタム Web 公開で実行するには、指定するソート順を [レコードのソート] スクリプトステップを使用して保存しておく必要があります。
URL を開く	このスクリプトステップは、カスタム Web 公開ソリューションでは効果はありません。
フィールドへ移動	[フィールドへ移動] を使用して Web ブラウザで特定のフィールドをアクティブにすることはできませんが、このスクリプトステップを他のスクリプトステップと組み合わせて使用して、タスクを実行することができます。たとえば、フィールドに移動して内容をコピーし、別のフィールドに移動して値を貼り付けることができます。ブラウザに結果を表示するには、必ず [レコード / 検索条件確定] スクリプトステップを使用してレコードを保存してください。
レコード / 検索条件確定	データベースにレコードを送信します。

スクリプトトリガとカスタム Web 公開ソリューション

FileMaker Pro では、スクリプトならびにユーザの操作 (ユーザによるフィールドのクリックなど) の両方でスクリプトトリガを実行できます。ただし、カスタム Web 公開では、スクリプトでのみ有効にすることが可能です。たとえば、カスタム Web 公開が OnObjectEnter スクリプトトリガをもつフィールドをクリックした場合、トリガは有効になりません。ただし、スクリプトステップによってフィールドへの移動がフォーカスされると、OnObjectEnter スクリプトトリガは実行されます。スクリプトトリガの詳細については、「FileMaker Pro ヘルプ」を参照してください。

第 4 章
カスタム Web 公開 with XSLT の概要

FileMaker XSLT を使用すると、XML データの変換、フィルタ、または書式設定を行って、Web ブラウザ、または他のプログラムやアプリケーションで使用できます。この章では、FileMaker XSLT スタイルシートと、XSLT スタイルシートの作成を開始する場合に役立つ FileMaker XSLT Site Assistant というツールの概要を説明します。FileMaker XSLT スタイルシートの構造の詳細については、第 6 章「FileMaker XSLT スタイルシートの開発」を参照してください。

FileMaker XSLT スタイルシートについて

FileMaker XSLT スタイルシートを使用して、次の処理を行うことができます。

- データベースのどのフィールドを公開するかをスタイルシートで制御することによってデータをフィルタする
- データベース名やフィールド名などのメタデータを隠す
- Web ページにデータを表示する書式を設定したり、Web ユーザがデータを操作する方法を制御する
- データを HTML またはテキスト（vCard やコンマ区切り値など）として出力する
- データを FileMaker XML 文法から別の XML 文法（SVG（Scalable Vector Graphics）など）に変換して、別のデータベースやアプリケーションで使用する
- FileMaker データのサブセットを他の Web サイトに統合したり、FileMaker データベースとは大きく異なる可能性がある他のミドルウェアやアプリケーションを使用して統合する
- 公開されるフィールド名を変更して、データベースデザイン情報の不正使用を防止する

メモ　FileMaker Server のカスタム Web 公開 with XSLT は、XSLT 1.0 の W3C 勧告に基づいています。XSLT 1.0 の詳細については、www.w3.org を参照してください。セッション管理、電子メールの送信、および Cookie とヘッダへのアクセスなどの追加の機能は、FileMaker XSLT 拡張関数によって提供されます。詳細については、54 ページの「FileMaker XSLT 拡張関数と引数の使用」を参照してください。Web 公開エンジンでは、XSL-FO（XSL Formatting Objects）はサポートされていません。

FileMaker XSLT スタイルシートの使用例

次に、FileMaker XSLT スタイルシートを使用可能なさまざまな例の一部を示します。

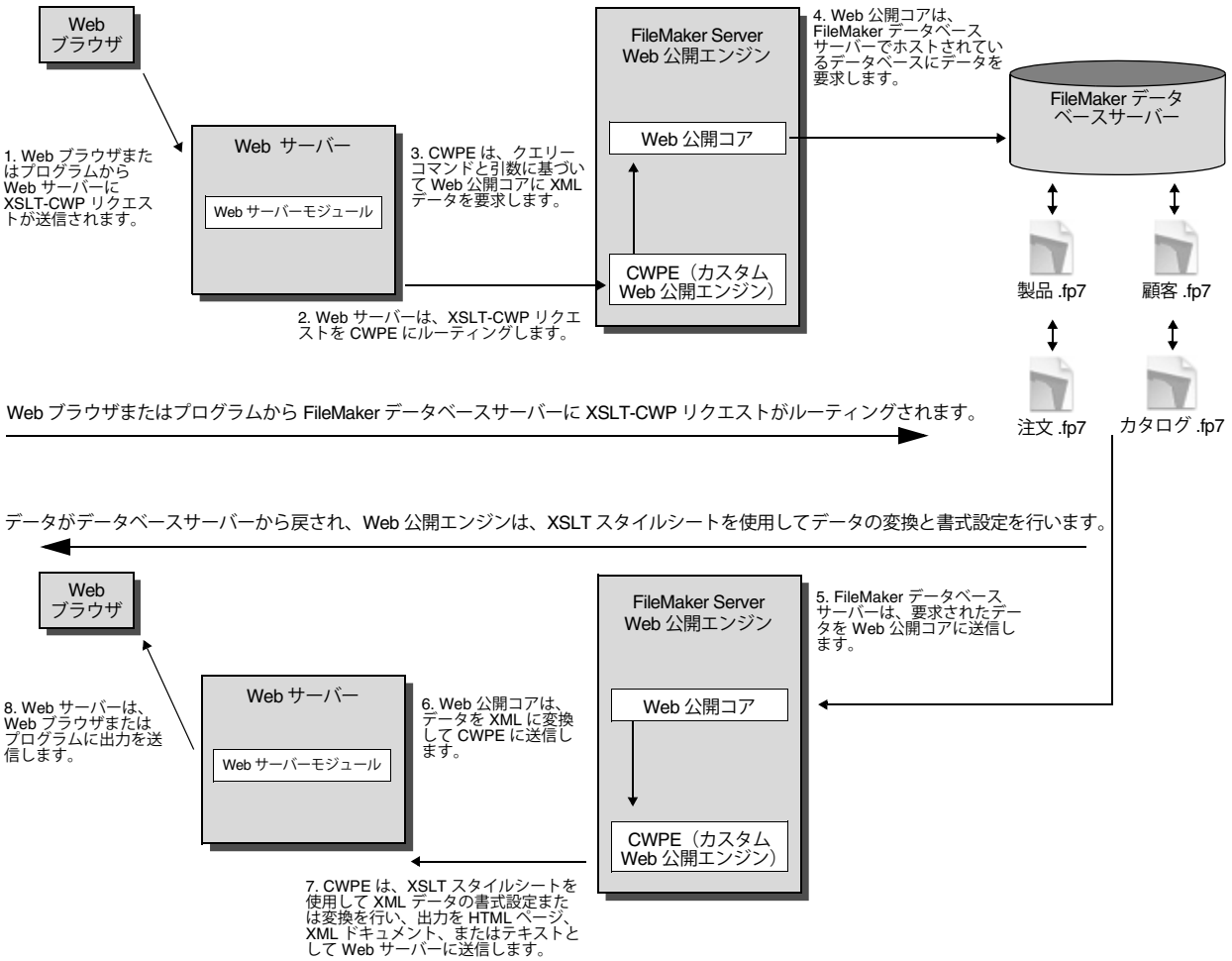
- FileMaker データベースのデータのサブセットが含まれるテーブルを Web ページに挿入して、Web ユーザが参照できます。たとえば、テーブルに名前と住所を含め、電話番号は含めないようにすることができます。不正アクセスを防止するために、Web ページでは、データに対して、FileMaker データベースの実際のフィールド名（「姓」など）ではなく、汎用のラベル（「名前」など）を表示できます。
- FileMaker ポータルのデータを他のデータソースの情報に統合する Web ページやアプリケーションを作成できます。
- FileMaker データベースに保存されている連絡先情報から vCard を作成するボタンを Web ページ上に作成できます。
- FileMaker データベースの XML データを、スプレッドシートやデータベースアプリケーションで開くことができる XML 文法に変換できます。

カスタム Web 公開 with XSLT の使用の開始

標準的な XML と XSLT の知識がある場合、FileMaker の XSLT 拡張関数や、クエリーコマンドと引数など、FileMaker XML と XSLT を使用した公開の詳細事項をいくつか学べば、すぐに Web 公開エンジンを使い始めることができます。XSLT Site Assistant は、スタイルシートの作成およびスタイルシートの構築方法の学習に役立つツールです。その後、好みの XML および XSLT オーサリングツールを使用して、スタイルシートの機能を向上させることができます。

Web 公開エンジンが XML データと XSLT スタイルシートに基づいてページを生成する方法

Web サーバーに XSLT-CWP（XSLT カスタム Web 公開）リクエストが送信されると、Web 公開エンジンは、スタイルシートおよび URL で定義されたクエリーコマンドと引数に基づいて FileMaker データベースに対してクエリーを実行し、XSLT スタイルシートの命令に従ってデータを出力します。



カスタム Web 公開 with XSLT を使用するための一般的な手順

次に、XSLT を使用したカスタム Web 公開を使用するための手順の概要を示します。

1. Admin Cosole で、XSLT 公開が有効になっていることを確認します。「FileMaker Server ヘルプ」を参照してください。
2. 公開する各 FileMaker データベースを FileMaker Pro を使用して開き、データベースで、カスタム Web 公開 with XSLT の fmxslt 拡張アクセス権が有効になっていることを確認します。19 ページの「データベースでのカスタム Web 公開の有効化」を参照してください。

   メモ スタイルシートを開発する際は、エンドユーザに提供するアクセス権セットと同等の FileMaker データベースのアクセス権セットを使用してください。同等のアクセス権セットを使用しなかった場合、開発者は、エンドユーザが使用できない FileMaker データベースのレイアウトや機能にアクセスできることになり、同じ動作を実現できません。

3. FileMaker データベースからの XML データの書式設定と変換を行うために、FileMaker に固有の XSLT 拡張関数、クエリーコマンド、およびクエリー引数が含まれる XSLT スタイルシートを作成します。

   FileMaker XSLT Site Assistant を使用し、サイトの開始点として、1 つまたは複数の基本的な XSLT スタイルシートを作成できます。次のセクション「FileMaker XSLT Site Assistant を使用した FileMaker XSLT スタイルシートの生成」を参照してください。

   また、必要に応じて、独自の XSLT オーサリングツールやテキスト編集ツールを使用して XSLT スタイルシートを変更したり、スタイルシートを最初から開発することができます。第 6 章「FileMaker XSLT スタイルシートの開発」を参照してください。

4. XSLT スタイルシートを「xslt-template-files」フォルダにコピーまたは保存します。
   「xslt-template-files」フォルダは、Web 公開エンジンがインストールされているホスト上の「FileMaker Server」フォルダ内の「Web Publishing」フォルダにあります。

   スタイルシートは、「xslt-template-files」フォルダ内のオプションのフォルダまたはフォルダ階層に保存することもできます。

5. 静的ファイルを Web サーバーに配置します。29 ページの「Web サイトまたはプログラムでの FileMaker XSLT スタイルシートの使用」を参照してください。

6. XSLT スタイルシートを使用する Web サイトやプログラムを作成または変更します。

   たとえば、Web ユーザを XSLT スタイルシートに自動転送するか、または XSLT スタイルシートへのリンクが指定された index.html などの静的ページを Web サイトに使用できます。

7. サイトまたはプログラムのセキュリティメカニズムが設定されていることを確認します。

8. Web ユーザ用に定義されているものと同じアカウントとアクセス権を使用して、XSLT スタイルシートを使用するサイトまたはプログラムをテストします。

9. サイトまたはプログラムを使用可能にし、ユーザに通知します。

FileMaker XSLT Site Assistant を使用した FileMaker XSLT スタイルシートの生成

FileMaker XSLT Site Assistant は、XSLT を使用した FileMaker カスタム Web 公開用の起点となる基本的な XSLT スタイルシートを作成する場合に使用できるアプリケーションです。XSLT Site Assistant は、FileMaker XSLT スタイルシートの構築の学習に効果的です。必要に応じて、独自の XSLT スタイルシートオーサリングツールやテキスト編集ツールを使用してスタイルシートを変更できます。XSLT Site Assistant を使用して既存のスタイルシートを編集または更新することはできませんが、サイト全体で使用する初期スタイルシートを生成したり、既存のサイトに基本的な機能（レコードの削除など）を追加するための単一のスタイルシートを作成することが可能です。

Site Assistant を使用して、カスタム Web 公開経由で FileMaker データベースを操作する場合に役立つあらゆるタイプのページに対して XSLT スタイルシートを生成できます。XSLT Site Assistant で選択するオプションに応じて、ユーザに次の操作を許可するサイトを作成できます。

- 一度に 1 つのレコードをブラウズする
- データベース内のすべてのレコードのリストを表示する
- データベースを検索して、結果をリストで表示する
- レコードのソート
- レコードの追加
- レコードの編集および複製
- レコードの削除
- 集計レポートの表示

また、生成される他の XSLT スタイルシートページへのリンクを含むホームページを生成することもできます（オプション）。

Web ユーザがいずれかの XSLT スタイルシートを参照する HTTP リクエストと URL を送信すると、Web 公開エンジンは、各スタイルシートを使用して FileMaker データベースから動的にデータを取得します。Web 公開エンジンは、スタイルシートを使用して XML データの変換と書式設定を行い、Web ユーザが操作できる最終的な HTML ページを生成します。

メモ XSLT Site Assistant のスタイルシートは、fmresultset XML 文法に基づいて FileMaker XML データを HTML ページに変換するもので、FileMaker XML エクスポートなど、XML データの他の用途には使用できません。

XSLT Site Assistant を使用する前に

XSLT Site Assistant を使用してデータベース用の XSLT スタイルシートを生成する前に、次の作業を行ってください。

- データベースで拡張アクセス権「fmxslt」を設定する。XSLT Site Assistant を実行する際は、Web ユーザに許可するアクセス権セットと同等のアクセス権セットを使用します。19 ページの「データベースでのカスタム Web 公開の有効化」を参照してください。
- FileMaker Server のデータベースサーバーコンポーネントでデータベースを開いてホストする。「FileMaker Server ヘルプ」を参照してください。
- FileMaker Server 展開の Web サーバーコンポーネントが実行されていることを確認する。
- FileMaker Server 展開の Web 公開エンジンコンポーネントが実行されていることを確認する。
- XSLT スタイルシートを使用およびテストするために、Web 公開エンジンで XSLT 公開を有効にする。「FileMaker Server ヘルプ」を参照してください。

XSLT Site Assistant の起動

メモ XSLT Site Assistant を使用するには、Java Runtime Environment 5 または Java Runtime Environment 6 をインストールする必要があります。

XSLT Site Assistant を起動するには、次の操作を行います。

1. ブラウザで [FileMaker Server Web 公開ツール] ページを開きます。

   次の URL へ移動します。

   http://< サーバー >:16000/tools

   < サーバー > は、FileMaker Server が存在しているコンピュータです。

2. [PHP Site Assistant および XSLT Site Assistant ツール] をクリックして、[FileMaker Server Web 公開ツール] ページに移動します。

3. [XSLT Site Assistant の開始] をクリックします。

   FileMaker Server によって、必要な JAR ファイルがローカルコンピュータにインストールされます。手順が完了するまで、処理の進行状況を示すダイアログボックスが表示されます。

4. ファイルのインストール後に、デスクトップに XSLT Site Assistant のアイコンをインストールするかどうかを選択できます（オプション）。[OK] をクリックすると、アイコンをインストールします。

これで XSLT Site Assistant の使用を開始できます。

XSLT Site Assistant の使用

XSLT Site Assistant を使用するための手順ごとの操作の詳細については、「XSLT Site Assistant ヘルプ」を参照してください。XSLT Site Assistant によって生成されたスタイルシートの使用の詳細については、29 ページの「Web サイトまたはプログラムでの FileMaker XSLT スタイルシートの使用」を参照してください。

重要 XSLT Site Assistant を使用する際に、複数のテーブルが含まれるデータベースを選択する場合は、同じテーブルに関連付けられているレイアウトを選択してください。同じテーブルに関連付けられていないレイアウトを選択すると、生成されるサイトは予想外の結果を返します。たとえば、データベースに「製品」テーブルと「顧客」テーブルが含まれる場合に、検索用のページ、レコード編集用のページ、およびレコード追加用のページに使用するレイアウトを選択するときは、これらのレイアウトがすべて同じテーブルに関連付けられていることを確認します。

XSLT Site Assistant によって生成されるスタイルシートについて

XSLT Site Assistant によって生成される XSLT スタイルシートには、FileMaker に固有の複数の処理命令、エレメント、および引数が含まれます。次に、スタイルシートに含まれる処理命令、エレメント、および引数の例をいくつか示します。

- <?xslt-cwp-query params="query string-fragment"?> 処理命令は、使用する XML 文法を指定し、XSLT Site Assistant で選択したデータベースの名前を静的に定義します。51 ページの「静的に定義されたクエリーコマンドとクエリー引数の使用」を参照してください。
- <xsl:param name="request-query"/> エレメントは、リクエストまたは HTML フォームデータ内のクエリー情報にアクセスするために使用されます。たとえば、XSLT Site Assistant のスタイルシートでこのエレメントを使用すると、現在のリクエストのクエリー情報にアクセスして、対象レコードで現在の場所を判断したり、前後のレコードへのリンクを作成することができます。54 ページの「リクエストのクエリー情報へのアクセス」を参照してください。
- <xsl:param name="authenticated-xml-base-uri"/> エレメントは、リクエスト内でさらに多くの XML データが必要になった場合に、リクエスト内の認証済みのベース URI にアクセスするために使用されます。このエレメントは、常にスタイルシートに含まれるわけではありません。55 ページの「認証済みベース URI 引数の使用」を参照してください。

さらに、エラーを定義するための「utilities.xsl」スタイルシート、および XSLT Site Assistant のスタイルシートによって呼び出される一般的な XSLT テンプレートも生成されます。

XSLT Site Assistant のスタイルシートの他のセクションの詳細については、第 6 章「FileMaker XSLT スタイルシートの開発」を参照してください。

Web サイトまたはプログラムでの FileMaker XSLT スタイルシートの使用

Web サイトやプログラムで Web 公開エンジンとともにスタイルシートを使用するための手順は、XSLT Site Assistant を使用して XSLT スタイルシートを生成した場合も、独自のスタイルシートを最初から作成した場合も同じです。

Web サイトまたはプログラムで FileMaker XSLT スタイルシートを使用するには、次の操作を行います。

1. XSLT スタイルシートを「 xslt-template-files」フォルダにコピーまたは保存します。
   「xslt-template-files」フォルダは、Web 公開エンジンがインストールされているホスト上の「 FileMaker Server」フォルダ内の「Web Publishing」フォルダにあります。

   スタイルシートは、「xslt-template-files」フォルダ内のオプションのフォルダまたはフォルダ階層に保存することもできます。

2. XSLT スタイルシートで静的イメージや HTML ファイルなどの静的ファイルが参照されている場合は、静的ファイルを、Web サーバーのルートフォルダ内に元のフォルダ階層で配置します。相対パスが保持されていることを確認します。

   たとえば、HTML タグ <img src="logo.jpg"> を使用して、XSLT スタイルシートで「logo.jpg」という名前のイメージファイルを参照している場合、「logo.jpg」ファイルは、Web サーバー上の次の場所に配置する必要があります。

   < ルートフォルダ >/fmi/xsl/logo.jpg

3. データベースのオブジェクトフィールドに実際のファイルではなくファイル参照が保存されている場合は、レコードを作成または編集するときに、参照されているオブジェクトを FileMaker Pro の「Web」フォルダに保存してから、Web サーバーソフトウェアのルートフォルダ内の同じ相対パスの場所にあるフォルダにコピーまたは移動する必要があります。20 ページの「Web 上でのオブジェクトフィールドの内容の公開について」を参照してください。

   メモ FileMaker データベースのオブジェクトフィールドに実際のファイルが保存されている場合は、データベースファイルが FileMaker Server 上で適切にホストされていてアクセス可能であれば、オブジェクトフィールドの内容を操作する必要はありません。

4. XSLT スタイルシートを要求および処理するには、次の URL 構文を使用します。

< スキーム >://< ホスト >[:< ポート >]/fmi/xsl/< フォルダ >/< スタイルシート >.xsl[?< クエリー文字列 >]

48 ページの「FileMaker XSLT スタイルシートの URL 構文について」を参照してください。

メモ Web サイトに対しては、XSLT スタイルシートをホームページとして含めることをお勧めします。これにより、ユーザがスタイルシートにアクセスするためにクエリー文字列を入力する必要がなくなります。XSLT Site Assistant を使用すると、<?xslt-cwp-query?> 処理命令が使用されているためクエリー文字列の必要のない「home.xsl」を作成できます。たとえば、スタイルシート（「home.xsl」スタイルシートを含む）を「xslt-template-files」フォルダ内の「my_templates」フォルダにコピーした場合は、次の URL を使用してスタイルシートを要求および処理することができます。

http://192.168.123.101/fmi/xsl/my_templates/home.xsl

重要 Web 公開エンジンでは、Web ユーザが「xslt-template-files」フォルダにインストールされている XSLT スタイルシートのソースを表示することは許可されません。Web ユーザがスタイルシートを処理するリクエストを送信した場合、Web 公開エンジンは、スタイルシートによる変換の結果のみを Web ブラウザまたはプログラムに送信します。

XSLT スタイルシートのトラブルシューティング

XSLT スタイルシートの使用に問題がある場合は、次の点を確認します。

- カスタム Web 公開 with XSLT の拡張アクセス権がデータベースで設定されていて、ユーザアカウントに割り当てられている。19 ページの「データベースでのカスタム Web 公開の有効化」を参照してください。
- FileMaker Server のデータベースサーバーコンポーネントでデータベースをホストして開く。「FileMaker Server ヘルプ」を参照してください。
- 使用しているデータベースアカウント名とパスワードが正しい。
- FileMaker Server 展開の Web サーバーコンポーネントが実行されている。
- FileMaker Server 展開の Web 公開エンジンコンポーネントが実行されている。
- Web 公開エンジンで XSLT 公開が有効になっている。
  - ブラウザで [FileMaker Server テクノロジーテスト] ページを開きます。

  http://< サーバー >:16000/test

  < サーバー > は、FileMaker Server が存在しているコンピュータです。
  - [XSLT カスタム Web 公開のテスト] リンクをクリックして、FMServer_Sample テストデータベースにアクセスする XSLT ページを開きます。

詳細については、『FileMaker Server 入門ガイド』および「FileMaker Server ヘルプ」を参照してください。

第 5 章
Web 公開エンジンを使用した XML データへのアクセス

Web 公開エンジンを使用することで、FileMaker データを XML（Extensible Markup Language）形式で取得したり、更新することができます。HTML が World Wide Web 上での通信の標準表示言語になったのと同様に、XML も、構造化データを交換するための標準言語になっています。多くの個人、組織、および企業が、XML を使用して製品情報、取引、在庫データなどの業務データを転送しています。

カスタム Web 公開 with XML の使用

標準的な XML の知識がある場合、使用する URL 構文やクエリー引数など、カスタム Web 公開 with XML に特有の詳細事項をいくつか学べば、すぐに Web 公開エンジンを使い始めることができます。

HTTP URL リクエストを FileMaker のクエリーコマンドと引数とともに使用することで、FileMaker Server によってホストされているデータベースに対してクエリーを実行して、結果のデータを XML 形式でダウンロードできます。たとえば、データベースに対して特定の郵便番号のレコードすべてを検索するクエリーを実行して、結果の XML データをさまざまな方法で使用できます。

また、Web 公開エンジンのサーバーサイド XSLT スタイルシートを使用して、XML データをフィルタしたり、データの書式を HTML または vCard などのテキストに再設定したり、データを SVG（Scalable Vector Graphics）などの他の XML 文法に変換することもできます。第 4 章「カスタム Web 公開 with XSLT の概要」および第 6 章「FileMaker XSLT スタイルシートの開発」を参照してください。

XML の一般情報、XML のその他の使用例、および XML に関する役立つ情報へのリンクについては、FileMaker の Web サイト www.filemaker.co.jp を参照してください。

メモ　Web 公開エンジンによって生成される XML データは、整形式で、XML 1.0 仕様に準拠しています。整形式の XML の要件の詳細については、www.w3.org で入手できる XML の仕様を参照してください。

Web 公開エンジンと FileMaker Pro の XML インポート / エクスポート機能の違い

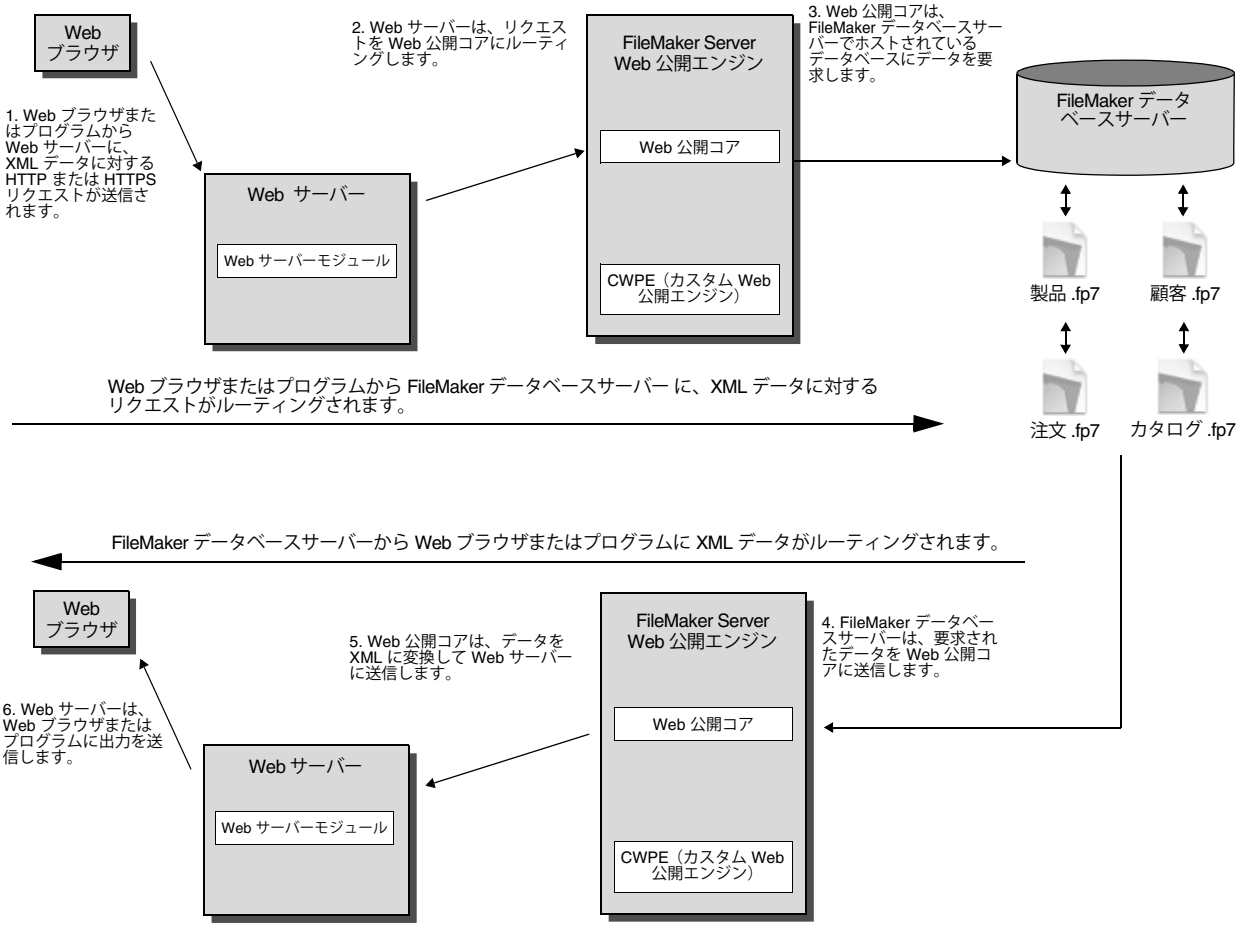
Web 公開エンジンと FileMaker Pro のどちらを使用しても、FileMaker データベースで XML データを使用できます。ただし、これらの 2 つの方法には、次に示すような重要な違いがあります。

- XML データおよび XSLT Web 公開にアクセスするために、Web 公開エンジンでは、fmresultset、FMPXMLRESULT、および FMPXMLLAYOUT 文法がサポートされています。FileMaker Pro では、XML のインポートには FMPXMLRESULT 文法、エクスポートには FMPXMLRESULT または FMPDSORESULT 文法が使用されます。35 ページの「Web 公開エンジンを使用した XML データへのアクセス」を参照してください。
- Web 公開エンジンで XML データにアクセスするには、URL で Web 公開エンジンのクエリー文字列を使用します。FileMaker Pro で XML をインポートおよびエクスポートするには、FileMaker Pro のメニューコマンドまたはスクリプトを使用します。
- Web 公開エンジンはサーバーベースで、FileMaker Server と同じホストにインストールするか、または異なるホストにインストールできます。FileMaker Pro の XML インポートおよびエクスポートの機能は、デスクトップベースです。
- Web 公開エンジンとともに URL リクエストを使用することで、FileMaker データベースから XML データに動的にアクセスできます。FileMaker Pro の XML エクスポート機能では、あらかじめ指定した XML データファイルが生成されます。
- Web 公開エンジンを使用した XML データの操作は、対話型の処理です。FileMaker Pro の XML インポートおよびエクスポートの機能は、バッチ処理です。
- Web 公開エンジンは FileMaker ポータルの XML データにアクセスできますが、FileMaker Pro ではできません。
- Web 公開エンジンはオブジェクトフィールド内のデータにアクセスできますが、FileMaker Pro ではできません。
- Web 公開エンジンは HTTP または HTTPS 上で FileMaker データにリアルタイムにアクセスできますが、FileMaker Pro ではできません。

メモ　FileMaker Pro を使用して XML 形式でデータをインポートおよびエクスポートする操作の詳細については、「FileMaker Pro ヘルプ」を参照してください。

Web 公開エンジンがリクエストから XML データを生成する方法

XML データに対するリクエストが Web サーバーに送信されると、Web 公開エンジンは、FileMaker データベースに対してクエリーを実行し、データを XML ドキュメントとして返します。



Web 公開エンジンから XML データにアクセスするための一般的な手順

次に、Web 公開エンジンを使用して FileMaker データベース内の XML データにアクセスする場合の手順の概要を示します。

1. FileMaker Server Admin Console で、XML 公開 が有効になっていることを確認します。「FileMaker Server ヘルプ」を参照してください。
2. 公開する各 FileMaker データベースを FileMaker Pro で開き、データベースで、XML カスタム Web 公開に対して fmxml 拡張アクセス権が有効になっていることを確認します。19 ページの「データベースでのカスタム Web 公開の有効化」を参照してください。

   ポータル内の XML データにアクセスするには、データベースレイアウトの表示形式を [フォーム形式] または [リスト形式] に設定します。ユーザまたはスクリプトによってデータベースレイアウトの表示形式が [表形式] に変更された場合は、最初の関連レコード（ポータルの最初の行）にのみ XML データとしてアクセスできます。

   XML データは、フィールドオブジェクトがレイアウトに追加された順序に対応する出力です。XML データ順序を画面上に表示されるフィールドの順序（上から下、左から右の順序）と一致させるには、すべてのフィールドを選択し、グループ化してからグループ解除します。この手順によって、画面の順序と一致したレイアウト順序にリセットされます。

3. HTML フォーム、HREF リンク、またはプログラムや Web ページ内のスクリプトを使用して、FileMaker XML 文法、1 つのクエリーコマンド、および 1 つまたは複数の FileMaker クエリー引数を指定した URL の形式で、HTTP または HTTPS リクエストを Web 公開エンジンに送信します。Web ブラウザに URL を入力することもできます。

   URL の指定の詳細については、次のセクション「XML データとオブジェクトにアクセスするための URL 構文について」を参照してください。クエリーコマンドと引数については、43 ページの「FileMaker クエリー文字列を使用した XML データの要求」および付録 A「クエリー文字列で使用される有効な名前」を参照してください。

4. Web 公開エンジンは、URL で指定された文法を使用して、リクエストの結果（データベースからのレコードのセットなど）が含まれる XML データを生成し、プログラムまたは Web ブラウザに返します。

5. Web ブラウザに XML パーサが含まれる場合は、Web ブラウザによってデータが表示されます。

   クライアントサイドのスタイルシートが指定されている場合は、Web ブラウザのパーサによって、該当するスタイルシート命令も適用されます。46 ページの「サーバーサイドおよびクライアントサイドでのスタイルシートの処理の使用」を参照してください。

XML データとオブジェクトにアクセスするための URL 構文について

このセクションでは、Web 公開エンジンを使用して FileMaker データベースの XML データおよびオブジェクトにアクセスするための URL 構文について説明します。XSLT スタイルシートを使用するための URL 構文は、XML データにアクセスするための構文とは異なります。48 ページの「FileMaker XSLT スタイルシートの URL 構文について」および 49 ページの「XSLT ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について」を参照してください。

XML データにアクセスするための URL 構文について

次に、Web 公開エンジンを使用して FileMaker データベースから XML データにアクセスするための URL 構文を示します。

< スキーム >://< ホスト >[:< ポート >]/fmi/xml/<XML 文法 >.xml[?< クエリー文字列 >]

各要素の意味は、次のとおりです。

- < スキーム > には、HTTP または HTTPS プロトコルを指定できます。
- < ホスト > には、Web サーバーがインストールされているホストの IP アドレスまたはドメイン名を指定します。
- < ポート > には、Web サーバーが使用するポートを指定します（オプション）。ポートが指定されていない場合は、プロトコルのデフォルトのポート（HTTP ではポート 80、HTTPS ではポート 443）が使用されます。
- <XML 文法 > には、FileMaker XML 文法の名前を指定します。指定できる値は、fmresultset.xml、FMPXMLRESULT.xml、FMPXMLLAYOUT.xml、または FMPDSORESULT.xml です。36 ページの「fmsresultset 文法の使用」および 38 ページの「他の FileMaker XML 文法の使用」を参照してください。
- < クエリー文字列 > には、FileMaker XML 公開に使用する 1 つのクエリーコマンドと 1 つまたは複数のクエリー引数の組み合わせを指定します（-dbnames コマンドでは引数は必要ありません）。43 ページの「FileMaker クエリー文字列を使用した XML データの要求」および付録 A「クエリー文字列で使用される有効な名前」を参照してください。

メモ クエリーコマンドおよび引数を含め、URL 構文では、クエリー文字列の部分を除いて大文字と小文字が区別されます。URL のほとんどの部分は小文字ですが、FMPXMLRESULT、FMPXMLLAYOUT、および FMPDSORESULT の 3 つの文法名は大文字です。クエリー文字列の大文字と小文字の規則の詳細については、77 ページの「クエリーコマンドと引数の使用のガイドライン」を参照してください。

次に、Web 公開エンジンを使用して XML データにアクセスするための URL の例を 2 つ示します。

http://server.company.com/fmi/xml/fmresultset.xml?-db=products&-lay=sales&-findall

http://192.168.123.101/fmi/xml/FMPXMLRESULT.xml?-db=products&-lay=sales&-findall

XML ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について

XML ソリューションに対して生成された XML ドキュメントでは、オブジェクトへの参照が保存されているフィールドと、実際のオブジェクトがデータベース内に保存されているオブジェクトフィールドで、オブジェクトを参照するために使用される構文が異なります。

- オブジェクトフィールドで実際のオブジェクトがデータベース内に保存されている場合、オブジェクトフィールドの <data> エレメントは、次の相対 URL 構文を使用してオブジェクトを参照します。

  <data>/fmi/xml/cnt/data.< 拡張子 >?< クエリー文字列 ></data>

  < 拡張子 > には、.jpg など、オブジェクトのタイプを識別するファイル拡張子を指定します。このファイル拡張子によって MIME タイプが設定され、Web ブラウザで適切にオブジェクトデータを識別できます。< クエリー文字列 > の詳細については、前のセクション「XML データにアクセスするための URL 構文について」を参照してください。

  例：

  <data>/fmi/xml/cnt/data.jpg?-db=products&-lay=sales&-field=product_image(1)&-recid=2</data>

  メモ オブジェクトフィールドに対して生成された XML では、-field クエリー引数の値は完全修飾フィールド名になります。括弧内の数字は、オブジェクトフィールドの繰り返し数を示し、繰り返しフィールドと非繰り返しフィールドの両方に対して生成されます。78 ページの「完全修飾フィールド名の構文について」を参照してください。

  データベースからオブジェクトデータを取得するには、次の構文を使用します。

  < スキーム >://< ホスト >[:< ポート >]/fmi/xml/cnt/data.< 拡張子 >?< クエリー文字列 >

  < スキーム >、< ホスト >、または < ポート > の詳細については、前のセクション「XML データにアクセスするための URL 構文について」を参照してください。

  例：

  http://www.company.com/fmi/xml/cnt/data.jpg?-db=products&-lay=sales&-field=product_image(1)&-recid=2

- オブジェクトフィールドに実際のオブジェクトではなくファイル参照が保存されている場合、オブジェクトフィールドの <data> エレメントには、オブジェクトを参照する相対パスが含まれます。例：

  <data>/images/logo.jpg</data>

  メモ レコードを作成または編集するときに、参照されているオブジェクトを FileMaker Pro の「Web」フォルダに保存してから、Web サーバーソフトウェアのルートフォルダ内の同じ相対パスの場所にあるフォルダにコピーまたは移動する必要があります。20 ページの「Web 上でのオブジェクトフィールドの内容の公開について」を参照してください。

- オブジェクトフィールドが空の場合は、オブジェクトフィールドの <data> エレメントも空になります。

メモ XML を使用してオブジェクトにアクセスするための構文は、XSLT を使用してオブジェクトにアクセスするための構文とは異なります。49 ページの「XSLT ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について」を参照してください。

URL のテキストエンコードについて

XML データおよびオブジェクトにアクセスするための URL は、UTF-8（Unicode Transformation 8 Bit）形式でエンコードされている必要があります。43 ページの「UTF-8 でエンコードされているデータについて」を参照してください。

たとえば、「info」フィールドの値を「fiancée」に設定するには、次の URL を使用することができます。

http://server.company.com/fmi/xml/fmresultset.xml?-db=members&-lay=relationships&-recid=2
&info= fianc%C3%A9e&-edit

この URL の例で、%C3%A9 は、UTF-8 で表した「é」文字を URL エンコードしたものです。

URL のテキストエンコードの詳細については、www.w3.org で入手できる URL の仕様を参照してください。

Web 公開エンジンを使用した XML データへのアクセス

Web 公開エンジンを通じて XML データにアクセスするには、使用する FileMaker 文法の名前、 1 つの FileMaker クエリーコマンド、および 1 つまたは複数の FileMaker クエリー引数を指定した URL を使用します。Web 公開エンジンによって、次のいずれかのタイプの XML 文法によって書式が設定された XML データがデータベースから生成されます。

- fmresultset：Web 公開エンジンには、この文法を使用することをお勧めします。この文法は、柔軟で XSLT スタイルシートオーサリングに最適化されており、名前によるフィールドアクセスや、関連セット（ポータル）データの操作をより簡単に行うことができます。また、この文法は、グローバル格納オプションや、集計および計算フィールドの識別など、FileMaker の用語や機能とより直接的なつながりを持ちます。この文法は、XML データへのアクセスと XSLT スタイルシートに使用できます。Web 公開を効率的に実行できるよう、この文法は、FMPXMLRESULT 文法よりも詳細になるように設計されています。36 ページの「fmsresultset 文法の使用」を参照してください。
- FMPXMLRESULT および FMPXMLLAYOUT:XML データにアクセスしたり、XSLT スタイルシートを使用するには、FMPXMLRESULT および FMPXMLLAYOUT 文法も Web 公開エンジンとともに使用できます。XML エクスポートとカスタム Web 公開の両方に 1 つのスタイルシートを使用するには、FMPXMLRESULT 文法を使用する必要があります。レイアウト内の値一覧およびフィールド表示情報にアクセスするには、FMPXMLLAYOUT 文法を使用する必要があります。38 ページの「他の FileMaker XML 文法の使用」を参照してください。
- FMPDSORESULT:FMPDSORESULT 文法は、FileMaker Pro で XML をエクスポートするためにサポートされているもので、Web 公開エンジン経由で XML データにアクセスする目的では使用されません。また、FMPDSORESULT 文法は XSLT スタイルシートに対してはサポートされていません。FMPDSORESULT 文法の詳細については、「FileMaker Pro ヘルプ」を参照してください。

URL リクエストで指定した文法に応じて、Web 公開エンジンにより、次の文法の 1 つを使用して XML ドキュメントが生成されます。各 XML ドキュメントには、使用する文法のデフォルトの XML ネームスペース宣言が格納されます。次のセクション「FileMaker XML のネームスペースについて」を参照してください。ドキュメントや Web ページでこれらの文法の 1 つを使用して、FileMaker のデータを XML 形式で表示および操作します。

メモ Web 公開エンジンによって生成される XML データは、UTF-8 形式（Unicode Transformation Format 8）を使用してエンコードされます。43 ページの「UTF-8 でエンコードされているデータについて」を参照してください。

FileMaker XML のネームスペースについて

XML タグが使用されるアプリケーションでタグを区別するには、固有の XML ネームスペースが役立ちます。たとえば、２つの <DATABASE> エレメント（FileMaker XML データと Oracle XML データ用にそれぞれ 1 つずつ）が XML ドキュメントに含まれている場合、各 <DATABASE> エレメントをネームスペースで区別できます。

Web 公開エンジンによって、各文法に対してデフォルトのネームスペースが生成されます。

使用する文法	生成されるデフォルトのネームスペース
fmresultset	xmlns="http://www.filemaker.com/xml/fmresultset"
FMPXMLRESULT	xmlns="http://www.filemaker.com/ fmpxmlresult"
FMPXMLLAYOUT	xmlns="http://www.filemaker.com/fmpxmllayout"

FileMaker データベースのエラーコードについて

最後に実行されたクエリーコマンドの実行時にエラーが発生した場合、Web 公開エンジンは、各 XML ドキュメントの始めに、エラーを表すエラーコードをエラーコードエレメントで囲んで返します。エラーがない場合、値ゼロ（0）が返されます。

使用する文法	使用される構文
fmresultset	<error code="0"></error>
FMPXMLRESULT	<ERRORCODE>0</ERRORCODE>
FMPDSORESULT	<ERRORCODE>0</ERRORCODE>

XML ドキュメントのエラーコードエレメントは、データベースとクエリー文字列に関するエラーを示します。XSLT スタイルシートでは他のタイプのエラーが発生する可能性もあり、異なる方法で処理されます。付録 B「カスタム Web 公開のエラーコード」を参照してください。

FileMaker 文法の文書型定義の取得

FileMaker 文法の DTD（文書型定義）は、HTTP リクエストを使用して取得できます。

使用する文法	使用する HTTP リクエスト
fmresultset	http://< ホスト >[:< ポート]/fmi/xml/fmresultset.dtd
FMPXMLRESULT	http://< ホスト >[:< ポート]/fmi/xml/FMPXMLRESULT.dtd
FMPXMLLAYOUT	http://< ホスト >[:< ポート]/fmi/xml/FMPXMLLAYOUT.dtd
FMPDSORESULT	http://< ホスト >[:< ポート]/fmi/xml/FMPDSORESULT.dtd?-db=< データベース >&-lay=< レイアウト >

fmsresultset 文法の使用

この文法では、XML エレメント名に FileMaker の用語が使用され、フィールドの保存内容がフィールドのタイプから分離されています。また、この文法には、集計、計算、およびグローバルフィールドを識別する機能も含まれています。

fmresultset 文法を使用するには、Web 公開エンジンに XML ドキュメントを要求する URL で、fmresultset 文法の次の名前を指定します。

fmresultset.xml

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&-findall

メモ fmresultset 文法を指定する場合は、小文字を使用してください。

Web 公開エンジンによって、fmresultset 文法を使用して XML ドキュメントが生成されます。この XML ドキュメントで、Web 公開エンジンは、ドキュメントの <?xml...?> 命令直後の 2 行目にある <!DOCTYPE> 命令で、fmresultset 文法の文書型定義を参照します。<!DOCTYPE> 命令によって、fmresultset 文法の DTD をダウンロードするための URL が指定されます。

fmresultset 文法のエレメントの説明

fmresultset 文法は主に、<datasource> エレメント、<metadata> エレメント、および <resultset> エレメントからなります。

<datasource> エレメント

fmresultset 文法では、<datasource> エレメントに、table、layout、date-format、time-format、timestamp-format、total-count、および database 属性が含まれます。

- XML ドキュメントの日付の書式は、<datasource> エレメントの date-format 属性で指定されます。

  MM/dd/yyyy

各要素の意味は、次のとおりです。

- MM - 月を表す 2 桁の値（01 から 12 まで。たとえば 01 は 1 月で 12 は 12 月です）。
- dd - 日を表す 2 桁の値（00 から 31 まで）。
- yyyy - 年を表す 4 桁の値。

- XML ドキュメントの時刻の書式は、<datasource> エレメントの time-format 属性で指定されます。

  HH:mm:ss

各要素の意味は、次のとおりです。

- HH - 時間を表す 2 桁の値（24 時間形式の 00 から 23 まで。）。
- mm - 分を表す 2 桁の値（00 から 59 まで）。
- ss - 秒を表す 2 桁の値（00 から 59 まで）。

- <datasource> エレメントの timestamp-format 属性によって、date-format と time-format ドキュメントの形式が 1 つのタイムスタンプに結合されます。

  MM/dd/yyyy HH:mm:ss

<metadata> エレメント

fmresultset 文法の <metadata> エレメントには、1 つまたは複数の <field-definition> および <relatedset-definition> エレメントが含まれ、各エレメントには、結果セットのフィールドの 1 つの属性が含まれます。

<field-definition > 属性では、次が指定されます。

- フィールドが 「auto-enter」フィールドであるかどうか（「yes」または「no」）
- フィールドが「four-digit-year」フィールドであるかどうか（「yes」または「no」）
- フィールドが 「global field」フィールドであるかどうか（「yes」または「no」）
- 繰り返し値の最大数（max-repeat 属性）
- 入力を許可する最大文字数（max-characters 属性）
- 「not-empty」フィールドであるかどうか（「yes」または「no」）
- 数字フィールドのみについてのものであるかどうか（「yes」または「no」）
- 結果（「text」、「number」、「date」、「time」、「timestamp」、または「container」）
- 「time-of-day 」フィールドであるかどうか（「yes」または「no」）
- タイプ （「normal」、「calculation」、または「summary」）
- フィールド名（必要に応じて完全修飾名）

<relatedset-definition> エレメントは、ポータルを表します。ポータル内の各関連フィールドは、<relatedset-definition> エレメントに含まれる <field-definition> エレメントで表されます。ポータル内に複数の関連フィールドがある場合、関連フィールドのフィールド定義は、単一の <relatedset-definition> エレメント内にまとめられます。

<resultset> エレメント

<resultset> エレメントには、クエリーの結果として返された <record> エレメントと、見つかったレコードの総数の属性が格納されます。各 <record> エレメントは、結果セット内の 1 つのレコードのフィールドデータ（レコードの mod-id および record-id 属性を含む）と、レコード内の 1 つのレコードのデータが含まれる <data> エレメントで構成されます。

ポータル内の各レコードは、<relatedset> エレメント内の <record> エレメントで表されます。<relatedset> エレメントの count 属性にはポータル内のレコードの数が指定され、table 属性にはポータルに関連付けられているテーブルが指定されます。

fmresultset 文法での XML データの例

次に、fmresultset 文法で生成される XML データの例を示します。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE fmresultset PUBLIC "-//FMI//DTD fmresultset//EN" ""http://localhost:16014/fmi/xml/fmresultset.dtd">
 <fmresultset xmlns="http://www.filemaker.com/xml/fmresultset" version="1.0">
 <error code="0" />
 <product build="12/31/2012" name="FileMaker Web Publishing Engine" version="0.0.0.0" />
 <datasource database="art" date-format="MM/dd/yyyy" layout="web3" table="art" time-format="HH:mm:ss" timestamp-format="MM/dd/yyyy
HH:mm:ss" total-count="12" />
 <metadata>
 <field-definition auto-enter="no" four-digit-year="no" global="no" max-repeat="1" name="Title" not-empty="no" numeric-only="no" result="text" time-
of-day="no" type="normal" />
 <field-definition auto-enter="no" four-digit-year="no" global="no" max-repeat="1" name="Artist" not-empty="no" numeric-only="no" result="text" time-
of-day="no" type="normal" />
 <relatedset-definition table="artlocations">
 <field-definition auto-enter="no" four-digit-year="no" global="no" max-repeat="1" name="artlocations::Location" not-empty="no" numeric-only="no"
result="text" time-of-day="no" type="normal" />
 <field-definition auto-enter="no" four-digit-year="no" global="no" max-repeat="1" name="artlocations::Date" not-empty="no" numeric-only="no"
result="date" time-of-day="no" type="normal" />
 </relatedset-definition>
 <field-definition auto-enter="no" four-digit-year="no" global="no" max-repeat="1" name="Style" not-empty="no" numeric-only="no" result="text" time-
of-day="no" type="normal" />
 <field-definition auto-enter="no" four-digit-year="no" global="no" max-repeat="1" name="length" not-empty="no" numeric-only="no" result="number"
time-of-day="no" type="calculation" />
 </metadata>
 <resultset count="1" fetch-size="1">
 <record mod-id="6" record-id="14">
 <field name="Title">
 <data>Spring in Giverny 3</data>
 </field>
 <field name="Artist">
 <data>Claude Monet</data>
 </field>
 <relatedset count="0" table="artlocations" />
 <field name="Style">
 <data />
 </field>
 <field name="length">
 <data>19</data>
 </field>
 </record>
 </resultset>
 </fmresultset>
```

他の FileMaker XML 文法の使用

他の FileMaker XML 文法には、フィールドタイプ、値一覧、およびレイアウトに関する情報が含まれます。FMPXMLRESULT は、fmresultset と機能的に同等です。レイアウト内の値一覧およびフィールド表示情報にアクセスするには、FMPXMLLAYOUT 文法を使用する必要があります。FMPXMLRESULT および FMPXMLLAYOUT 文法は、データ交換用としてより簡潔になっています。

FMPXMLRESULT 文法を使用するには、Web 公開エンジンに XML ドキュメントを要求する URL で、次の文法名を指定します。

FMPXMLRESULT.xml

例：

http://192.168.123.101/fmi/xml/FMPXMLRESULT.xml?-db=employees&-lay=family&-findall

FMPXMLLAYOUT 文法を使用するには、Web 公開エンジンに XML ドキュメントを要求する URL で、次の文法名を -view クエリーコマンドとともに指定します。

FMPXMLLAYOUT.xml

例：

http://192.168.123.101/fmi/xml/FMPXMLLAYOUT.xml?-db=employees&-lay=family&-view

メモ FMPXMLRESULT および FMPXMLLAYOUT 文法を指定する場合は、文法名を大文字で入力してください。

生成された XML ドキュメントで、Web 公開エンジンは、ドキュメントの <?xml...?> 命令直後の 2 行目にある <!DOCTYPE> 命令で指定されている文法の文書型定義を参照します。<!DOCTYPE> 命令によって、文法の DTD をダウンロードするための URL が指定されます。

FMPXMLRESULT 文法のエレメントの説明

FMPXMLRESULT 文法では、<DATABASE> エレメントに、NAME、RECORDS、DATEFORMAT、および TIMEFORMAT 属性が含まれます。

XML ドキュメントの日付の書式は、<DATABASE> エレメントの DATEFORMAT 属性で指定されます。XML ドキュメントの時刻の書式は、<DATABASE> エレメントの TIMEFORMAT 属性で指定されます。FMPXMLRESULT 文法と fmresultset 文法の日付および時刻の書式は同じです。36 ページの「fmresultset 文法のエレメントの説明」の表を参照してください。

FMPXMLRESULT 文法の <METADATA> エレメントには 1 つまたは複数の FIELD エレメントが含まれ、各 <FIELD> エレメントには、結果セットのフィールド / 列のうちの 1 つの情報が含まれます。この情報には、データベースで定義されているとおりのフィールドの名前、フィールドタイプ、空欄のフィールドの許可（Yes）/ 不許可（No）（EMPTYOK 属性）、および繰り返し値の最大数（MAXREPEAT 属性）が含まれます。フィールドタイプに対して有効な値は、TEXT、NUMBER、DATE、TIME、および CONTAINER です。

<RESULTSET> エレメントには、クエリーの結果として返されたすべての <ROW> エレメントと、見つかったレコードの総数の属性が格納されます。各 <ROW> エレメントには、結果セットの 1 つの行に対するフィールド / 列のデータが含まれます。このデータには、行の RECORDID と MODID（86 ページの「-modid （修正 ID）クエリー引数」を参照）、および <COL> エレメントが含まれます。<COL> エレメントには、行内の 1 つのフィールド / 列のデータが含まれ、複数の <DATA> エレメントは、繰り返しフィールドまたはポータルフィールドの値の 1 つを表します。

FMPXMLRESULT 文法での XML データの例

次に、FMPXMLRESULT 文法で生成される XML データの例を示します。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE FMPXMLRESULT PUBLIC "-//FMI//DTD FMPXMLRESULT//EN" ""http://localhost:16014/fmi/xml/FMPXMLRESULT.dtd">
<FMPXMLRESULT xmlns="http://www.filemaker.com/fmpxmlresult">
 <ERRORCODE>0</ERRORCODE>
 <PRODUCT BUILD="12/31/2012" NAME="FileMaker Web Publishing Engine" VERSION="0.0.0.0" />
 <DATABASE DATEFORMAT="MM/dd/yyyy" LAYOUT="web" NAME="art" RECORDS="12" TIMEFORMAT="HH:mm:ss" />
 <METADATA>
  <FIELD EMPTYOK="YES" MAXREPEAT="1" NAME="Title" TYPE="TEXT" />
   <FIELD EMPTYOK="YES" MAXREPEAT="1" NAME="Artist" TYPE="TEXT" />
  <FIELD EMPTYOK="YES" MAXREPEAT="1" NAME="Image" TYPE="CONTAINER" />
 </METADATA>
 <RESULTSET FOUND="1">
 <ROW MODID="6" RECORDID="15">
 <COL>
  <DATA>Spring in Giverny 4</DATA>
 </COL>
 <COL>
  <DATA>Claude Monet</DATA>
 </COL>
 <COL>
  <DATA>/fmi/xml/cnt/data.jpg?-db=art&-lay=web&-recid=15&-field=Image(1)</DATA>
 </COL>
 </ROW>
 </RESULTSET>
</FMPXMLRESULT>
```

<COL> エレメントの順序は、<METADATA> エレメント内の <FIELD> エレメントの順序に一致します。たとえば、<METADATA> エレメントに「Title」および「Artist」フィールドが一覧表示されている場合、「Village Market」および「Camille Pissarro」は、これと同じ順序で <RESULTSET> および <ROW> エレメントに一覧表示されます。

FMPXMLLAYOUT 文法のエレメントの説明

FMPXMLLAYOUT 文法では、レイアウト名、データベース名、およびデータベース側の対応するレイアウトで見つかった各フィールドの <FIELD> エレメントが <LAYOUT> エレメントに含まれます。各 <FIELD> エレメントでフィールドのスタイルタイプが説明され、フィールドの関連した値一覧の VALUELIST 属性がこのエレメントに含まれます。

<VALUELISTS> エレメントには、レイアウトにある各値一覧の <VALUELIST> エレメントが 1 つまたは複数含まれます（各エレメントには、値一覧の名前と、一覧の各値の <VALUE> エレメントが含まれます）。

FileMaker データベースの [値一覧に使用するフィールドの指定] ダイアログボックスで選択したオプションに応じて <VALUE> 要素は、最初のフィールドのみ、２番目のフィールドのみ、または値一覧の両方のフィールドを含む DISPLAY 属性を含みます。たとえば、値一覧内の最初のフィールドがアートスタイルの ID 番号 ("100" など) を格納し、２番目のフィールドがアートスタイルの関連付けられた名前 ("Impressionism" など) を表示する場合、[値一覧に使用するフィールドの指定] ダイアログボックスで各種組み合わせのオプションが選択されたときの DISPLAY 属性の内容についての概要は次のようになります。

- [２番目のフィールドの値も表示] が選択されなかった場合、DISPLAY 属性は、値一覧の最初のフィールドの値のみ含みます。次の XML データの例では DISPLAY 属性はアートスタイルの ID 番号のみ含みます。

```
<VALUELISTS>
  <VALUELIST NAME="style">
   <VALUE DISPLAY="100">100</VALUE>
   <VALUE DISPLAY="101">101</VALUE>
   <VALUE DISPLAY="102">102</VALUE>
  </VALUELIST>
 </VALUELISTS>
```

- [２番目のフィールドの値も表示] と [２番目のフィールドの値のみを表示] の両方が選択された場合、DISPLAY 属性は、２番目のフィールドの値のみ含みます。次の XML データの例では DISPLAY 属性はアートスタイルの名前のみ含みます。

```
<VALUELISTS>
  <VALUELIST NAME="style">
   <VALUE DISPLAY="Impressionism">100</VALUE>
   <VALUE DISPLAY="Cubism">101</VALUE>
   <VALUE DISPLAY="Abstract">102</VALUE>
  </VALUELIST>
 </VALUELISTS>
```

- [２番目のフィールドの値も表示] が選択され、[２番目のフィールドの値のみを表示] が選択されなかった場合、DISPLAY 属性は、値一覧の両方のフィールドの値を含みます。次の XML データの例では DISPLAY 属性はアートスタイルの ID 番号と名前の両方を含みます。

```
<VALUELISTS>
  <VALUELIST NAME="style">
   <VALUE DISPLAY="100 Impressionism">100</VALUE>
   <VALUE DISPLAY="101 Cubism">101</VALUE>
   <VALUE DISPLAY="102 Abstract">102</VALUE>
  </VALUELIST>
 </VALUELISTS>
```

日付、時刻、およびタイムスタンプのフィールドの場合は、そのフィールドタイプに「fm」の書式を使用して、値のデータを書式設定します。「fm」形式では、日付は MM/dd/yyyy、時刻は HH:mm:ss、タイムスタンプは MM/dd/yyyy HH:mm:ss です。65 ページの「日付、時刻、および曜日拡張関数の使用」を参照してください。たとえば、「birthdays」値一覧をレイアウトの「birthdate」フィールドでのポップアップメニューに使用し、その「birthdate」フィールドが日付タイプである場合には、その値一覧の値出力はすべて「fm」日付の書式になります。

メモ レイアウト上で異なるフィールドタイプである２つのフィールドが同じ値一覧 を共有している場合には、その値一覧データの書式は、 1 番目のフィールドのタイプによって決まります。

FMPXMLLAYOUT 文法での XML データの例

次に、FMPXMLLAYOUT 文法で生成される XML データの例を示します。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE FMPXMLLAYOUT PUBLIC "-//FMI//DTD FMPXMLLAYOUT//EN" ""http://localhost:16014/fmi/xml/FMPXMLLAYOUT.dtd">
 <FMPXMLLAYOUT xmlns="http://www.filemaker.com/fmpxmllayout">
 <ERRORCODE>0</ERRORCODE>
 <PRODUCT BUILD="12/31/2012" NAME="FileMaker Web Publishing Engine" VERSION="0.0.0.0" />
 <LAYOUT DATABASE="art" NAME="web2">
 <FIELD NAME="Title">
  <STYLE TYPE="EDITTEXT" VALUELIST="" />
 </FIELD>
 <FIELD NAME="Artist">
  <STYLE TYPE="EDITTEXT" VALUELIST="" />
 </FIELD>
 <FIELD NAME="Image">
  <STYLE TYPE="EDITTEXT" VALUELIST="" />
 </FIELD>
 <FIELD NAME="artlocations::Location">
  <STYLE TYPE="EDITTEXT" VALUELIST="" />
 </FIELD>
 <FIELD NAME="artlocations::Date">
  <STYLE TYPE="EDITTEXT" VALUELIST="" />
 </FIELD>
 <FIELD NAME="Style">
  <STYLE TYPE="POPUPMENU" VALUELIST="style" />
 </FIELD>
 </LAYOUT>
  <VALUELISTS>
   <VALUELIST NAME="style">
    <VALUE DISPLAY="Impressionism">100</VALUE>
    <VALUE DISPLAY="Cubism">101</VALUE>
    <VALUE DISPLAY="Abstract">102</VALUE>
   </VALUELIST>
  </VALUELISTS>
</FMPXMLLAYOUT>
```

UTF-8 でエンコードされているデータについて

Web 公開エンジンによって生成されるすべての XML データは、UTF-8 形式（Unicode Transformation Format 8）を使用してエンコードされます。この形式では、データの ASCII 文字は、標準的な Unicode 形式の 16 ビットから 8 ビットに圧縮されます。XML パーサが Unicode と UTF-8 エンコードをサポートしている必要があります。

UTF-8 エンコードの場合、英語で使用される標準的な ASCII 文字セットは 0 から 127 の値で直接表され、それ以上の値の Unicode 文字については、マルチバイトのエンコードが使用されます。

メモ UTF-8 ファイルをサポートする Web ブラウザまたはテキストエディタプログラムを使用してください。

UTF-8 エンコード形式には次の特徴があります。

- ASCII 文字は、すべて 1 バイトの UTF-8 文字になります。したがって、ASCII で有効な文字列は、UTF-8 文字列としても有効です。
- ASCII 以外の文字（ビットの大きい文字セット）は、マルチバイト文字の一部です。
- UTF-8 文字の 1 バイト目は、その文字に含まれる追加バイトの数を示します。
- マルチバイト文字の 1 バイト目は、2 バイト目以降と容易に区別できるため、データストリームのどの位置でも、文字の開始位置を簡単に判断できます。
- UTF-8 と Unicode は簡単に相互変換できます。
- UTF-8 でエンコードしたテキストは比較的小さくなり、ASCII 文字の比率が大きいテキストの場合、Unicode よりも小さくなります。最も条件が悪い場合でも、UTF-8 文字列は、適合する Unicode 文字列と比較して 50% 程度しか大きくなりません。

FileMaker クエリー文字列を使用した XML データの要求

FileMaker データベースに XML データを要求するには、クエリー文字列で FileMaker クエリーコマンドと引数を使用します。たとえば、URL 内の次のクエリー文字列で -findall クエリーコマンドを使用して、「products」という名前の FileMaker データベースに含まれるすべての製品の一覧を要求できます。

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=products-lay=sales&-findall

クエリー文字列に含めるクエリーコマンドは、-new など、1 つだけにする必要があります。ほとんどのクエリーコマンドでは、対応するさまざまなクエリー引数もクエリー文字列で指定する必要があります。たとえば、-dbnames 以外のすべてのクエリーコマンドでは、クエリー対象のデータベースを指定する -db 引数が必要です。

クエリーコマンドと引数は、URL で使用したり、FileMaker XSLT スタイルシートの <?xslt-cwp-query?> 処理命令で使用することもできます。第 6 章「FileMaker XSLT スタイルシートの開発」を参照してください。

このセクションでは、FileMaker クエリーコマンドと引数の概要を説明します。クエリー文字列でのクエリーコマンドと引数の使用の詳細については、付録 A「クエリー文字列で使用される有効な名前」を参照してください。

メモ Web 公開エンジンでは、FileMaker XSLT スタイルシート専用に定義されている追加のクエリーコマンド（-process）と 3 つのクエリー引数もサポートされています。50 ページの「FileMaker XSLT スタイルシートでのクエリー文字列の使用」を参照してください。

使用するクエリーコマンド名	実行するコマンド
-dbnames	ホストされているすべての Web 共有データベースの名前の取得
-delete	レコード削除
-dup	レコード複製
-edit	レコードの編集
-find	レコードの検索
-findall	すべてのレコードの検索
-findany	ランダムなレコードの検索
-findquery	複雑または複合検索条件を実行する
-layoutnames	ホストされている Web 共有データベースで利用可能なすべてのレイアウトの名前の取得

使用するクエリーコマンド名	実行するコマンド
-new	新規レコードとして追加する
-scriptnames	ホストされている Web 共有データベースで利用可能なすべてのスクリプトの名前の取得
-view	FMPXMLLAYOUT 文法が指定されている場合は、データベースからのレイアウト情報の取得。fmresultset または FMPXMLRESULT 文法が指定されている場合は、XML ドキュメントの <metadata> セクションおよび空のレコードセットの取得。

使用するクエリー引数名	使用するクエリーコマンド
-db（データベース名）	-dbnames および -process（XSLT リクエストのみ）以外のすべてのクエリーコマンドで必要です。
-delete.related	-edit のオプションです。
-field	オブジェクトリクエストの URL でフィールドを指定するために必要です。34 ページの「XML ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について」を参照してください。
フィールド名	-edit では、少なくとも 1 つのフィールド名が必要です。-find ではオプションです。84 ページの「フィールド名（オブジェクトフィールド以外のフィールド名）クエリー引数」を参照してください。
フィールド名 .op（演算子）	–find のオプションです。
-lay（レイアウト名）	-dbnames、-layoutnames、-scriptnames、および -process（XSLT リクエストのみ）以外のすべてのクエリーコマンドで必須です。
-lay.response（XML 応答に対するレイアウトの切り替え）	-dbnames、-layoutnames、-scriptnames、および -process（XSLT リクエストのみ）以外のすべてのクエリーコマンドでのオプションです。
-lop（論理演算子）	-find のオプションです。
-max（最大レコード）	-find および -findall のオプションです。
-modid（修正 ID）	-edit のオプションです。
-query	複合検索条件 -findquery で必須です。
-recid（レコード ID）	–edit、–delete、および –dup で必要です。–find のオプションです。
-relatedsets.filter	-find、-edit、-new、-dup、および -findquery のオプションです。
-relatedsets.max	-find、-edit、-new、-dup、および -findquery のオプションです。
-script（スクリプトの実行）	-find、-findall、-findany、-new、-edit、-delete、-dup、および -view のオプションです。
-script.param（-script によって指定されたスクリプトに引数値を渡します）	-script のオプションです。
-script.prefind（-find、-findany、および -findall の前にスクリプトを実行）	-find、-findany、および -findall のオプションです。
-script.prefind.param（-script.prefind によって指定されたスクリプトに引数値を渡します）	-script.prefind のオプションです。
-script.presort（ソートの前にスクリプト実行）	-find および -findall のオプションです。
-script.presort.param（-script.presort によって指定されたスクリプトに引数値を渡します）	-script.presort のオプションです。
-skip（レコードのスキップ）	-find および -findall のオプションです。
-sortfield.[1-9]（フィールドのソート）	-find および -findall のオプションです。
-sortorder.[1-9]（ソート順）	-find および -findall のオプションです。

使用するクエリー引数名	使用するクエリーコマンド
-stylehref（スタイルシートの HREF）	すべてのクエリーコマンドでオプションです（-styletype に対してスタイルシートの URL を指定します）。
-styletype（スタイルシートの種類）	すべてのクエリーコマンドでオプションです（クライアントサイドのスタイルシートを指定します）。

XML 応答に対するレイアウトの切り替え

–lay クエリー引数には、XML データを要求する場合に使用するレイアウトを指定します。多くの場合、リクエストから生成されるデータの処理には、同じレイアウトが適しています。場合によっては、セキュリティ上の理由から結果の表示に使用するレイアウトには存在しないフィールドが含まれる別のレイアウトを使用して、データを検索できます。フィールド内のデータを検索するには、XML リクエストで指定したレイアウトにそのフィールドが配置されている必要があります。

XML 応答を表示するために、XML リクエストの処理に使用するレイアウトとは異なるレイアウトを指定するには、オプションの –lay.response クエリー引数を使用できます。

たとえば、次のリクエストは、「Budget」レイアウト上の「Salary」フィールドで 100,000 を超える値を検索します。結果のデータは「ExecList」レイアウトを使用して表示されます。これには「Salary」フィールドは含まれません。

```
http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=Budget&Salary=100000&Salary.op=gt&-find
&-lay.response=ExecList
```

XML リクエストの処理方法の理解

XML リクエストの処理と XML ドキュメントの生成を制御するクエリー引数は複数あります。

次に、FileMaker Server と Web 公開エンジンが XML リクエストを処理する順序を示します。

1. –lay クエリー引数を処理します。
2. クエリーで指定されたグローバルフィールド値を指定します（URL の「.global=」部分）。
3. –script.prefind クエリー引数を処理します（指定されている場合）。
4. –find や –new などのクエリーコマンドを処理します。
5. –script.presort クエリー引数を処理します（指定されている場合）。
6. 結果のデータをソートします（ソートが指定されていた場合）。
7. –lay.response クエリー引数を処理して別のレイアウトに切り替えます（指定されている場合）。
8. –script クエリー引数を処理します（指定されている場合）。
9. XML ドキュメントを生成します。

上のいずれかの手順でエラーコードが生成された場合、リクエストの処理は停止し、以降の手順は実行されません。ただし、リクエスト内の前の手順は引き続き実行されます。

たとえば、現在のレコードを削除し、レコードをソートしてからスクリプトを実行するリクエストがあるとします。–sortfield 引数で存在しないフィールドが指定されている場合、このリクエストでは、現在のレコードが削除され、エラーコード 102（「フィールドが見つかりません。」）が返されますが、スクリプトは実行されません。

サーバーサイドおよびクライアントサイドでのスタイルシートの処理の使用

Web 公開エンジンでは、サーバーサイドでの XSLT スタイルシートの処理がサポートされています。また、クライアントサイドでのスタイルシートの処理を指定するクエリー引数を指定することもできます。

スタイルシートを処理するこれら 2 つの方法の違いと、クライアントサイドの処理を使用した場合のセキュリティへの影響を理解することが重要です。サーバーサイドでの処理では、Web ユーザがフィルタされていない XML データにアクセスすることはできないため、サーバーサイドでの処理の方がクライアントサイドでの処理よりも安全です。サーバーサイドでの処理では、データは、データの所有者または XSLT スタイルシートの作成者が適切と判断した形式で提示されます。サーバーサイドの処理では、データベース名、フィールド名、および他の実装の詳細は Web ユーザから隠されます。さらに、サーバーサイドの処理を使用して、静的に定義されたクエリー引数を指定することもでき、データベース名など、権限のないクエリーコマンドとクエリー引数の使用を防止できます。第 4 章「カスタム Web 公開 with XSLT の概要」および第 6 章「FileMaker XSLT スタイルシートの開発」を参照してください。

ソリューションでクライアントサイドでのスタイルシートの処理が必要な場合は、FileMaker クエリー文字列リクエストに –styletype および –stylehref 引数を含めることで、Web 公開エンジンによって各文法で XML スタイルシート処理命令を生成できます。XML ドキュメントの表示には、CSS（Cascading Style Sheet）または XSLT スタイルシートを使用できます。

- –styletype 引数は、TYPE 属性（type=text/css または type=text/xsl）の値を設定するために使用されます。
- –stylehref 引数は、絶対パスを使用してスタイルシートの場所を指定する HREF 属性の値を設定するために使用されます。例：href=/mystylesheet.css または href=/stylesheets/mystylesheet.xsl。スタイルシートの名前には任意の名前を指定できますが、.css または .xsl のいずれかの拡張子を含める必要があります。

次に、クライアントサイドのスタイルシートの処理を生成する FileMaker クエリー文字列の例を示します。

```
http://localhost/fmi/xml/fmresultset.xml?-db=products-lay=sales&-findall&-styletype=text/xsl&-stylehref=/mystylesheet.xsl
```

メモ　この例では、「–stylehref=/document.xsl」の「/」は、スタイルシートが Web サーバーソフトウェアのルートフォルダにあるために使用されています。絶対パスを使用して Web サーバー上での場所を指定するスタイルシートには、URL を使用してください。スタイルシートは、別の Web サーバー上に配置することもできます。

Web 公開エンジンにより、このリクエストに基づいて、次の処理命令が XML ドキュメントに含められます。

```
<?xml–stylesheet type="text/xsl" href="/mystylesheet.xsl"?>
```

クライアントサイドでの処理に使用するスタイルシートは、HREF 属性の URL に絶対パスで指定された場所にある Web サーバーにコピーまたは配置します。

重要　クライアントサイドでの処理に使用するスタイルシートは、「xslt-template-files」フォルダ内には配置しないでください。このフォルダは、サーバーサイドでの XSLT スタイルシートの処理に使用されます。29 ページの「Web サイトまたはプログラムでの FileMaker XSLT スタイルシートの使用」を参照してください。

メモ　一部の Web ブラウザでは、クライアントサイドでの処理はサポートされていません。詳細については、Web ブラウザのマニュアルを参照してください。

XML ドキュメントへのアクセスに関するトラブルシューティング

Web 公開エンジンを使用して XML ドキュメントにアクセスできない場合は、次の点を確認してください。

- XML カスタム Web 公開用にデータベースの拡張アクセス権が設定されていて、ユーザアカウントに割り当てられている。19 ページの「データベースでのカスタム Web 公開の有効化」を参照してください。
- データベースは、FileMaker Server 展開のデータベースサーバーコンポーネントでホストされ、FileMaker Server によって開かれている。「FileMaker Server ヘルプ」を参照してください。
- 使用しているデータベースアカウント名とパスワードが正しい。
- FileMaker Server 展開の Web サーバーコンポーネントが実行されている。
- FileMaker Server 展開の Web 公開エンジンコンポーネントが実行されている。
- Web 公開エンジンコンポーネントで XML 公開が有効になっている。「FileMaker Server ヘルプ」を参照してください。

第 6 章
FileMaker XSLT スタイルシートの開発

この章では、FileMaker XSLT スタイルシートの構造と、FileMaker XSLT 拡張関数の使用方法を説明します。

Web 公開エンジンでの XSLT スタイルシートの使用

Web 公開エンジン経由で FileMaker XML データを要求するための XSLT スタイルシートを開発および使用する場合は、次の点に注意してください。

- Web 公開エンジンで XSLT スタイルシートを使用するには、URL で XSLT スタイルシートの名前を指定する必要があります。スタイルシートが指定されていない場合や、Web 公開エンジンがスタイルシートを見つけたり解釈できない場合は、Web 公開エンジンによってエラーページが表示されます。48 ページの「FileMaker XSLT スタイルシートの URL 構文について」を参照してください。
- スタイルシートを保存するスタイルシートファイル名とフォルダ名は、エンコードが UTF-8 である必要があります。スタイルシートが古い Web ブラウザと互換性がなければならない場合には、名前を ASCII 文字に制限してください。
- 使用する FileMaker XML 文法を、URL 内のクエリー引数、または <?xslt-cwp-query?> 処理命令で静的に定義されたクエリー引数として指定する必要があります。XML 文法が指定されていない場合、Web 公開エンジンによってエラーが表示されます。50 ページの「FileMaker XSLT スタイルシートの XML 文法の指定」を参照してください。
- 要求する FileMaker XML データを識別するクエリー引数を、URL 内、または <?xslt-cwp-query?> 処理命令で静的に定義されたクエリー引数として指定する必要があります。48 ページの「FileMaker XSLT スタイルシートの URL 構文について」および 51 ページの「静的に定義されたクエリーコマンドとクエリー引数の使用」を参照してください。
- -encoding クエリー引数を使用することで、XSLT リクエストのテキストエンコードを指定できます（オプション）。エンコードが指定されていない場合、Web 公開エンジンは、リクエストに対してデフォルトのテキストエンコード設定を使用します。52 ページの「リクエストのテキストエンコードの設定」を参照してください。
- <xsl:output> エレメントの method 属性を使用して、出力方法を指定できます（オプション）。出力方法が指定されていない場合、Web 公開エンジンは、出力として HTML を使用します。また、<xsl:output> エレメントの encoding 属性を使用して、出力ページのエンコードを指定することもできます（オプション）。エンコードが指定されていない場合は、出力ページに使用するデフォルトのテキストエンコード設定が使用されます。52 ページの「出力方法とエンコードの指定」を参照してください。
- fmxslt:send_email() 拡張関数の関数引数を使用して、Web 公開エンジンから送信される電子メールメッセージのテキストエンコードを指定できます（オプション）。60 ページの「Web 公開エンジンからの電子メールメッセージの送信」を参照してください。

リクエストを構築するため、Web 公開エンジンは、最初に、オプションの <?xslt-cwp-query?> 処理命令で静的に定義されたクエリーコマンドとクエリー引数を使用します。静的に定義されたクエリーコマンドと引数は、基本リクエストになります。スタイルシートに <?xslt-cwp-query?> 処理命令は必須ではありませんが、ここで指定した基本リクエストは、URL クエリー文字列で指定された一致するクエリーコマンドや引数よりも優先されます。続いて、Web 公開エンジンは、<?xslt-cwp-query?> 処理命令で定義されて*いない* URL クエリー文字列内のクエリーコマンドや追加の引数を基本リクエストに追加します。Web 公開エンジンは、このリクエストを使用して FileMaker XML データを取得し、Web ブラウザやプログラムに、指定した出力方法または HTML でデータを返します。

FileMaker XSLT 拡張関数リファレンスについて

このリリースには、各 FileMaker XSLT 拡張関数の簡単な説明と例が含まれた「XSLT Reference.fp7」という FileMaker データベースが収録されています。関数リファレンスデータベースは、FileMaker Server 展開にある任意のコンピュータ（マスタまたはワーカー）の以下のディレクトリにあります。

Mac OS

/ ライブラリ /FileMaker Server/Example/XSLT

Windows

< ドライブ >:¥ Program Files ¥ FileMaker ¥ FileMaker Server ¥ Examples ¥ XSLT

ここで、< ドライブ > は、システムを起動する第 1 ドライブです。

FileMaker XSLT Starter Solution について

このリリースには、XSLT ソリューションで使用できるサンプルがある FileMaker XSLT Starter Solution が付属します。XSLT Starter Solution は、FileMaker Server 展開にある任意のコンピュータ（マスタまたはワーカー）の以下のディレクトリにあります。

Mac OS

/ ライブラリ /FileMaker Server/Example/XSLT/Starter Solution

Windows

< ドライブ >:¥ Program Files ¥ FileMaker ¥ FileMaker Server ¥ Examples ¥ XSLT ¥ Starter Solution

ここで、< ドライブ > は、システムを起動する第 1 ドライブです。

FileMaker XSLT スタイルシートの URL 構文について

次に、FileMaker XSLT スタイルシートを Web 公開エンジンで使用するための URL 構文を示します。

< スキーム >://< ホスト >[:< ポート >]/fmi/xsl/[< パス >/]< スタイルシート .xsl>[?< クエリー文字列 >]

各要素の意味は、次のとおりです。

- < スキーム > には、HTTP または HTTPS プロトコルを指定できます。
- < ホスト > には、Web サーバーがインストールされているホストの IP アドレスまたはドメイン名を指定します。
- < ポート > には、Web サーバーが使用するポートを指定します（オプション）。ポートが指定されていない場合は、プロトコルのデフォルトのポート（HTTP ではポート 80、HTTPS ではポート 443）が使用されます。
- < パス > はオプションで、XSLT スタイルシートが保存されている「xslt-template-files」フォルダ内のフォルダを指定します。
- < スタイルシート .xsl> には、XSLT スタイルシートのファイル名を指定します。
- < クエリー文字列 > には、カスタム Web 公開 with XSLT で使用する 1 つのクエリーコマンドと 1 つまたは複数のクエリー引数の組み合わせを指定することができます。50 ページの「FileMaker XSLT スタイルシートでのクエリー文字列の使用」および付録 A「クエリー文字列で使用される有効な名前」を参照してください。指定したスタイルシートに <?xslt-cwp-query?> 処理命令が含まれる場合は、URL クエリー文字内の一致するクエリーコマンドよりも、静的に定義されたクエリーコマンドと引数が優先されます。51 ページの「静的に定義されたクエリーコマンドとクエリー引数の使用」を参照してください。

メモ クエリーコマンドおよび引数を含め、URL 構文では、クエリー文字列の部分を除いて大文字と小文字が区別されます。URL のほとんどの部分は小文字ですが、文法名 FMPXMLRESULT および FMPXMLLAYOUT は大文字です。クエリー文字列の大文字と小文字の規則の詳細については、77 ページの「クエリーコマンドと引数の使用のガイドライン」を参照してください。

次に、FileMaker XSLT スタイルシートを Web 公開エンジンとともに使用するための URL の例を示します。

http://192.168.123.101/fmi/xsl/my_template/my_stylesheet.xsl?-grammar=fmresultset&-db=mydatabase

&-lay=mylayout&-findall

XSLT ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について

XSLT ソリューションに対して生成された XML ドキュメントでは、オブジェクトへの参照が保存されているフィールドと、実際のオブジェクトがデータベース内に保存されているオブジェクトフィールドで、オブジェクトを参照するために使用される構文が異なります。

- オブジェクトフィールドで実際のオブジェクトがデータベース内に保存されている場合、オブジェクトフィールドの <data> エレメントでは、次の URL 構文を使用してオブジェクトが参照されます。

  <data>/fmi/xsl/cnt/data.< 拡張子 >?< クエリー文字列 ></data>

  < 拡張子 > には、.jpg や .mov など、オブジェクトのタイプを識別するファイル拡張子を指定します。< クエリー文字列 > の詳細については、前のセクション「FileMaker XSLT スタイルシートの URL 構文について」を参照してください。

  例：

  <data>/fmi/xsl/cnt/data.jpg?–db=products&–lay=sales&–field=product_image(1)&–recid=2</data>

  メモ オブジェクトフィールドに対して生成された XML では、-field クエリー引数の値は完全修飾フィールド名になります。括弧内の数字は、オブジェクトフィールドの繰り返し数を示し、繰り返しフィールドと非繰り返しフィールドの両方に対して生成されます。78 ページの「完全修飾フィールド名の構文について」を参照してください。

  データベースからオブジェクトデータを取得するには、次の構文を使用します。

  < スキーム >://< ホスト >[:< ポート >]/fmi/xml/cnt/data.< 拡張子 >?< クエリー文字列 >

  < スキーム >、< ホスト >、または < ポート > の詳細については、前のセクション「FileMaker XSLT スタイルシートの URL 構文について」を参照してください。

  例：

  http://www.company.com/fmi/xsl/cnt/data.jpg?–db=products&–lay=sales&–field=product_image(1)&–recid=2

- オブジェクトフィールドに実際のオブジェクトではなくファイル参照が保存されている場合、オブジェクトフィールドの <data> エレメントには、オブジェクトを参照する相対パスが含まれます。たとえば、「logo.jpg」が「FileMaker Pro」フォルダ内の「Web」フォルダにある場合、このオブジェクトフィールドの <data> エレメントは次のようになります。

  <data>/images/logo.jpg</data>

  メモ レコードを作成または編集するときに、参照されているオブジェクトを FileMaker Pro の「Web」フォルダに保存してから、Web サーバーソフトウェアのルートフォルダ内の同じ相対パスの場所にあるフォルダにコピーまたは移動する必要があります。20 ページの「Web 上でのオブジェクトフィールドの内容の公開について」を参照してください。

- オブジェクトフィールドが空の場合は、オブジェクトフィールドの <data> エレメントも空になります。

FileMaker XSLT スタイルシートでのクエリー文字列の使用

URL 内、または FileMaker XSLT スタイルシートの <?xslt-cwp-query?> 処理命令内でクエリー文字列を使用する場合、FileMaker データベースに XML データを要求するために定義されている任意のクエリーコマンドと引数を含めることができます。43 ページの「FileMaker クエリー文字列を使用した XML データの要求」を参照してください。

また、FileMaker XSLT スタイルシート専用に定義されている次のクエリーコマンドと引数を使用することもできます。

使用する XSLT クエリーコマンド または引数名	目的	コメント
-grammar	XSLT-CWP リクエストまたは XSLT スタイルシートの XML 文法を指定する。次のセクション「FileMaker XSLT スタイルシートの XML 文法の指定」を参照してください。	このクエリー引数は、すべての XSLT リクエストで必須です。
-encoding	リクエストのテキストエンコードを指定する。52 ページの「リクエストのテキストエンコードの設定」を参照してください。	このクエリー引数は、すべての XSLT リクエストでオプションです。
-process	データを要求せずにスタイルシートを処理する。53 ページの「FileMaker Server に対してクエリーを実行しない XSLT リクエストの処理」を参照してください。	このクエリーコマンドには、-grammar クエリー引数が必要です。
-token	セッションまたは Cookie を使用せずにページ間で値を渡す。53 ページの「トークンを使用したスタイルシート間での情報の受け渡し」を参照してください。	このクエリー引数は、すべての XSLT リクエストでオプションです。

FileMaker XSLT スタイルシートの XML 文法の指定

カスタム Web 公開 with XSLT で推奨される XML 文法は、fmresultset 文法です。この文法は、XSLT との連携が容易になるように設計されています。36 ページの「fmsresultset 文法の使用」を参照してください。また、古い FMPXMLRESULT または FMPXMLLAYOUT 文法を使用することもできます。レイアウト内の値一覧およびフィールド表示情報にアクセスするには、FMPXMLLAYOUT 文法を使用する必要があります。38 ページの「他の FileMaker XML 文法の使用」を参照してください。カスタム Web 公開 with XSLT では、FMPDSORESULT 文法は使用できません。

FileMaker XSLT スタイルシートの文法を指定するには、URL、または <?xslt-cwp-query?> 処理命令で静的に定義されたクエリー引数で、-grammar クエリーコマンドを使用します。

たとえば、次のような URL があるとします。

```
http://192.168.123.101/fmi/xsl/my_template/my_stylesheet.xsl?-grammar=fmresultset&-db=mydatabase
&-lay=mylayout&-findall
```

たとえば、次のような処理命令があるとします。

```
<?xslt-cwp-query params="-grammar=fmresultset&-db=mydatabase&-lay=mylayout&-findall"?>
```

重要 FileMaker XSLT スタイルシートの XML 文法が指定されていない場合、エラー「QUERY -ER0001」が表示されます。付録 B「カスタム Web 公開のエラーコード」を参照してください。

FileMaker XSLT スタイルシートのネームスペースと接頭語

XSLT タグが使用されるアプリケーションでタグを区別するには、固有の XSLT ネームスペースが役立ちます。スタイルシートで使用する FileMaker XSLT 拡張関数と特定の文法のネームスペースは、すべての FileMaker XSLT スタイルシートの先頭にある <xsl:stylesheet> エレメントで宣言します。

使用する文法	宣言するネームスペース	使用する接頭語
fmresultset XML 文法	xmlns:fmrs="http://www.filemaker.com/xml/fmresultset"	fmrs
FMPXMLRESULT 文法	xmlns:fmp="http://www.filemaker.com/fmpxmlresult"	fmp
FMPXMLLAYOUT 文法	xmlns:fml="http://www.filemaker.com/fmpxmllayout"	fml
クエリー XML 文法	xmlns:fmq="http://www.filemaker.com/xml/query"	fmq
FileMaker XSLT 拡張関数	xmlns:fmxslt="xalan://com.fmi.xslt.ExtensionFunctions"	fmxslt

各 FileMaker XSLT スタイルシートでは、次の必須のネームスペースも宣言する必要があります。

```
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
```

次に、ネームスペース宣言の例を示します。

```
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:fmrs="http://www.filemaker.com/xml/fmresultset"
    xmlns:fml="http://www.filemaker.com/fmpxmllayout"
    xmlns:fmq="http://www.filemaker.com/xml/query"
    xmlns:fmxslt="xalan://com.fmi.xslt.ExtensionFunctions"
    exclude-result-prefixes="xsl fmrs fmq fml fmxslt">
```

静的に定義されたクエリーコマンドとクエリー引数の使用

FileMaker XSLT スタイルシートでは、XML データの要求時に使用するクエリーコマンドとクエリー引数を静的に定義しておくことで、クエリーコマンドとクエリー引数の不正使用を防止できます。これは必須ではありませんが、クエリーコマンドと引数をスタイルシートに静的に定義した場合、これらのクエリーコマンドと引数は、クライアントが URL クエリー文字列で指定する可能性がある、一致するクエリーコマンドまたは引数よりも優先されます。

XSLT Site Assistant によって生成されるスタイルシートでは、静的に定義されたクエリーコマンドと引数が使用されます。ソリューションのセキュリティを高める最も効果的な方法として、静的に定義されたクエリーコマンドと引数を使用することをお勧めします。

クエリーコマンドと引数を静的に定義するには、FileMaker XSLT スタイルシートの先頭で次の処理命令を使用します。

```
<?xslt-cwp-query params="query string-fragment"?>
```

各要素の意味は、次のとおりです。

「query string-fragment」には、名前と値の組が含まれる文字列を次の形式で指定します。

```
名前 = 値 & 名前 2= 値 2....
```

各要素の意味は、次のとおりです。

名前には、クエリーコマンド、クエリー引数、またはデータベースフィールドの名前である文字列を指定します。

値には、任意の長さの文字列値を指定します。クエリー引数およびフィールド名には、「-db=products」など、定義に特定の値を使用します。クエリーコマンドには、-findall などのコマンド名の後に「=」記号や値を指定しないでください。付録 A「クエリー文字列で使用される有効な名前」を参照してください。

クエリー文字列で使用する文字列は、URL エンコードされている必要があります。34 ページの「URL のテキストエンコードについて」を参照してください。<xsl:output> タグの encoding 属性で指定されている文字エンコードと同じ文字エンコードを使用する必要があります。エンコードが指定されていない場合、Web 公開エンジンは、設定されているデフォルトのエンコードを使用します。

名前と値の 2 つの組を区切る区切り文字には、アンパサンド（&）を使用する必要があります。

たとえば、「my_stylesheet.xsl」という名前のスタイルシートで次の処理命令を使用するとします。

```
<?xslt-cwp-query params="-db=products&-lay=sales&-grammar=fmresultset&productname=the%20item&-find"?>
```

この処理命令の例は、「my_stylesheet.xsl」へのすべてのリクエストで、「products」データベースと「sales」レイアウトとともに強制的に fmresultset 文法を使用し、値が「the%20item」に設定されている「productname」フィールドに対して -find リクエストを実行するようにしています。

クライアントが、「my_stylesheet.xsl」を使用する次のリクエストを実行したとします。

```
http://server.company.com/fmi/xsl/my_stylesheet.xsl?-lay=revenue&city=London&-edit
```

この場合、次の XML リクエストが Web 公開エンジンによって処理されることになります。

```
http://server.company.com/fmi/xml/fmresultset.xml?-db=products&-lay=sales&productname=the%20item&city=
London&-find
```

クライアントが入力した -lay=revenue クエリー引数と -edit クエリーコマンドよりも、静的に定義されたクエリーコマンドと引数が優先されます。「city」フィールドは処理命令で静的に定義されていないので、クライアントが入力した「city」フィールドの「London」という値は、Web 公開エンジンによって XML リクエストに含められます。

リクエストのテキストエンコードの設定

Web 公開エンジンは、XSLT リクエストのエンコードを判断できるまで、次の手順をこの順序で実行します。

1. Content-Type リクエストヘッダに charset 属性が設定されているかどうかを確認します。
2. -encoding クエリー引数にエンコードが指定されているかどうかを確認します。この引数は、URL で指定するか、または <?xslt-cwp-query?> 処理命令で静的に定義されたクエリー引数として指定できます。-encoding 引数の値は、リクエスト内の残りの引数に対して使用されるエンコードを示します。次の表は、この引数の有効な値を示します。例：

   http://192.168.123.101/fmi/xsl/template/my_stylesheet.xsl?-db=products-lay=sales&-grammar=fmresultset&-encoding=Shift_JIS&-findall
3. Web 公開エンジンのデフォルトのテキストエンコードオプション [リクエストと出力ページ :] の現在の設定を使用します。Web 公開エンジンを初めてインストールした状態では、リクエストに対するデフォルトのテキストエンコードの初期状態の設定は、Shift_JIS になっています。Web 公開エンジンのテキストエンコードの設定は、Admin Console を使用して変更できます。「FileMaker Server ヘルプ」を参照してください。

Web 公開エンジンによってエンコードが判断された後は、そのエンコードが使用され、エンコードを判断するための以降の手順は実行されません。たとえば、Content-Type リクエストヘッダで charset 属性が設定されている場合、-encoding クエリー引数の値は使用されません。

これらのいずれかの方法で指定するテキストエンコードには、次のエンコードの 1 つを使用する必要があります。

エンコード	説明
US-ASCII	基本 ASCII 文字セット。一般的には、標準テキストの英語の電子メールに使用されます。
ISO-8859-1	Latin 1 文字セット。一般的には、上位 ASCII 文字が必要なローマ字ベースの Web ページや電子メールメッセージに使用されます。
ISO-8859-15	Latin 9 文字セット。Latin 1 文字セットとほぼ同じですが、ユーロ € 記号が追加されています。
ISO-2022-JP	ISO の日本語エンコード。一般的には、日本語の電子メールメッセージに使用されます。
Shift_JIS	日本語エンコード。一般的には、日本語の Web ページに使用されます。
UTF-8	Unicode の 8 ビットのエンコード。主要なブラウザと電子メールクライアントでサポートされるようになったため、電子メールメッセージや Web ページでの UTF-8 の使用は一般的になっています。UTF-8 では Unicode 文字の全範囲がサポートされているため、すべての言語のページを処理できます。

注意

- Web 公開エンジンを初めてインストールした状態では、出力ページに対するデフォルトのテキストエンコードの初期状態の設定は Shift_JIS になっています。次のセクション「出力方法とエンコードの指定」を参照してください。Web 公開エンジンでは、電子メールメッセージに対しては、デフォルトのテキストエンコードの初期状態の設定である ISO-2022-JP が使用されます。これらの設定は、Admin Console を使用して変更できます。
- fmxslt:send_email(String SMTP フィールド , String 本文 , String エンコード) 拡張関数を使用して、電子メールメッセージのエンコードを設定することもできます。60 ページの「Web 公開エンジンからの電子メールメッセージの送信」を参照してください。

出力方法とエンコードの指定

<xsl:output> エレメントの method および encoding 属性を使用することで、出力ページの出力方法とエンコードを指定できます。これらの属性はどちらもオプションです。

method 属性は出力のタイプを指定するもので、「html」、「text」、または「xml」を指定できます。他のタイプの方法はサポートされていません。方法が指定されていない場合は、「html」メソッドが使用されます。

encoding 属性には、出力ページのエンコードを指定します。前のセクションの表に記載されている任意のエンコードを指定できます。エンコードが指定されていない場合は、出力ページに使用するデフォルトのテキストエンコード設定が使用されます。

例：

```
<xsl:output method="html" encoding="Shift_JIS"/>
```

スタイルシートで <xsl:output> エレメントを使用していない場合は、出力ページに使用するデフォルトのテキストエンコードの現在の設定を使用して、HTML ページが出力されます。

XSLT スタイルシートのエンコードについて

スタイルシートの先頭にある XML 宣言の encoding 属性で、リクエストと出力ページのエンコードの他に、XSLT スタイルシートのエンコードも指定する必要があります。52 ページの表に示されている任意のテキストエンコードを使用できます。

たとえば、次の宣言では、スタイルシートのエンコードとして UTF-8 を指定しています。

```
<?xml version="1.0" encoding="UTF-8"?>
```

スタイルシートのエンコードが指定されていない場合、Web 公開エンジンでは、エンコードが UTF-8 であると想定されます。

FileMaker Server に対してクエリーを実行しない XSLT リクエストの処理

スタイルシートで、レコード、フィールド名、レイアウト名など、データベースに固有の情報が必要ない場合は、-process クエリーコマンドを使用して、データベースのデータを必要としない XSLT リクエストを処理できます。このような場合、-process コマンドを使用することで、FileMaker Server の作業負荷を軽減することができます。

たとえば、-process コマンドを使用して、次のような処理を行うことができます。

- データベースの情報が必要ない場合に、静的なページを生成するスタイルシートをロードする
- スタイルシートでデータベースやレイアウトの情報（値一覧など）が必要ない場合に、新しいレコードを作成するスタイルシートをロードする
- データベースのデータを必要としない fmxslt:send_email() などの拡張関数を使用する
- データベースの情報が必要ない場合に、セッションに保存された情報にアクセスする

-process コマンドは、Web 公開エンジンの製品情報が含まれた XML ドキュメントを返します。

-process コマンドに必要な引数は -grammar のみで、fmresultset 文法または FMPXMLRESULT 文法を使用する必要があります。

例：

```
http://192.168.123.101/fmi/xsl/my_template/my_stylesheet.xsl?-grammar=fmresultset&-process
```

トークンを使用したスタイルシート間での情報の受け渡し

-token クエリー引数を、URL 内で、または静的に定義されたクエリーコマンドとして使用すると、セッションや Cookie を使用せずに、スタイルシート間でユーザ定義の情報を渡すことができます。-token クエリー引数は、すべてのクエリーコマンドでオプションです。

ユーザ定義引数の値には、URL エンコードされた任意の文字列を使用できます。

例：

```
http://192.168.123.101/fmi/xsl/template/my_stylesheet.xsl?-db=products&-lay=sales&-grammar=fmresultset
&-token.D100=Pending&-findall
```

91 ページの「-token. [文字列]（XSLT スタイルシート間での値の受け渡し）クエリー引数」を参照してください。

重要 -token クエリー引数は、プライベートデータを渡す目的では使用しないでください。

-token クエリー引数の値を取得するには、<xsl:param name="request-query" /> ステートメントを使用します。54 ページの「リクエストのクエリー情報へのアクセス」を参照してください。

FileMaker XSLT 拡張関数と引数の使用

FileMaker XSLT 拡張関数は、fmxslt ネームスペースに存在するように定義されています。必ず、XSLT スタイルシートの先頭にある <xsl:stylesheet> エレメントに、fmxslt ネームスペースの宣言を含めてください。50 ページの「FileMaker XSLT スタイルシートのネームスペースと接頭語」を参照してください。

FileMaker XSLT 拡張関数は、XPath ステートメント内に関数呼び出しとして指定することによって、XSLT スタイルシートで使用できるように設計されています。XPath ステートメントは、多くの XSLT エレメントで select 属性および test 属性の値として使用されます。

たとえば、User-Agent ヘッダを確認して、使用されているブラウザを判断するとします。このためには、User-Agent ヘッダの値を格納する次のような変数を使用できます。

```
<xsl:variable name="user-agent" select="fmxslt:get_header('User-Agent')"/>
```

値を返す拡張関数では、値は、指定された XSLT タイプで返されます。多くの関数は文字列を返しますが、トラバースできるノードセットを返す関数もあります。

メモ　このセクションでは、FileMaker XSLT 拡張関数と引数について説明し、いくつかの例を示しています。各関数の他の例については、FileMaker XSLT 拡張関数リファレンスを参照してください。48 ページの「FileMaker XSLT 拡張関数リファレンスについて」を参照してください。

Web 公開エンジンによって設定される FileMaker に固有の XSLT 引数について

リクエストを処理する際に、Web 公開エンジンによって、FileMaker に固有の次の XSLT 引数の値が動的に設定されます。<xsl:param> エレメントを使用して、スタイルシートでこれらの引数の値を使用することができます。

FileMaker に固有の XSLT 引数	詳細の参照先
<xsl:param name="request-query"/>	次のセクションの「リクエストのクエリー情報へのアクセス」
<xsl:param name="client-ip"/> <xsl:param name="client-user-name"/> <xsl:param name="client-password"/>	55 ページの「クライアント情報の取得」
<xsl:param name="xml-base-uri"/>	55 ページの「Web 公開エンジンのベース URI 引数の使用」
<xsl:param name="authenticated-xml-base-uri">	55 ページの「認証済みベース URI 引数の使用」

リクエストのクエリー情報へのアクセス

FileMaker XSLT 引数を使用して、URL または HTML フォームデータのリクエストに含まれるクエリー情報にアクセスできます。たとえば、現在のリクエストのクエリー情報にアクセスして、対象レコードの現在の場所を判断したり、前後のレコードへのリンクを作成することができます。

Web 公開エンジン経由で FileMaker XML データを要求するために使用されるすべてのクエリーコマンドとクエリー引数へのアクセスは、次の FileMaker XSLT 引数によって提供されます。

```
<xsl:param name="request-query"/>
```

フィールド名を除き、Web 公開エンジンでは、すべてのクエリーコマンドとクエリー引数の名前は小文字で返されます。フィールド名が大文字の場合は、大文字のまま返されます。

request-query 引数には、XML ドキュメントの一部が次の文法でロードされます。

```
<!DOCTYPE query [
    <!ELEMENT query (parameter)*>
        <!ATTLIST query action CDATA #REQUIRED>
        <!ELEMENT parameter (#PCDATA)>
        <!ATTLIST parameter name CDATA #REQUIRED>
        ]
```

メモ　クエリー情報は、fmq="http://www.filemaker.com/xml/query" ネームスペースに存在するように定義されます。必ず、XSLT スタイルシートの先頭にある <xsl:stylesheet> エレメントに、fmq ネームスペースの宣言を含めてください。50 ページの「FileMaker XSLT スタイルシートのネームスペースと接頭語」を参照してください。

たとえば、次のリクエストのクエリーコマンドとクエリー引数にアクセスするとします。

```
http://192.168.123.101/fmi/xsl/my_stylesheet.xsl?-db=products&-lay=sales&-grammar=fmresultset&-token.1=abc123&-findall
```

<xsl:param name="request-query" /> ステートメントをテンプレートセクションの前に含めた場合、Web 公開エンジンによって、次の XML ドキュメントが引数に格納されます。

```
<query action="my_stylesheet.xsl" xmlns="http://www.filemaker.com/xml/query">
    <parameter name="-db">products</parameter>
    <parameter name="-lay">sales</parameter>
    <parameter name="-grammar">fmresultset</parameter>
    <parameter name="-token.1">abc123</parameter>
    <parameter name="-findall"></parameter>
</query>
```

その後、XPath 式を使用することによって、request-query 引数を使用して、URL で渡されたトークンの値にアクセスできます。例：

```
$request-query/fmq:query/fmq:parameter[@name = '–token.1']
```

クライアント情報の取得

次の FileMaker XSLT 引数を使用して、Web 公開エンジンから Web クライアントの IP アドレス、ユーザ名、およびパスワードを取得できます。

```
<xsl:param name="client-ip"/>
<xsl:param name="client-user-name"/>
<xsl:param name="client-password">
```

これらの引数ステートメントは、XSLT スタイルシートで最も上にある <xsl:template> エレメントの前に含めます。

これらの引数により、パスワードで保護された追加の XML ドキュメントがスタイルシートのプログラムによってロードされるときに、Web ユーザの証明書が提供されます。56 ページの「追加のドキュメントのロード」を参照してください。Web ユーザは、最初に HTTP 基本認証のダイアログボックスでユーザ名とパスワードを入力する必要があります。19 ページの「保護されたデータベースへのアクセス」を参照してください。

これら 3 つの FileMaker XSLT 引数の使用に関する詳細と他の例については、『FileMaker XSLT 拡張関数リファレンス』を参照してください。

Web 公開エンジンのベース URI 引数の使用

Web 公開エンジンでは、ベース URI（Uniform Resource Identifier）引数は、Web 公開エンジンがインストールされているホストとポートになるように定義されています。ベース URI により、FileMaker データベースの XML データへのリクエストを Web 公開エンジンホストに対して相対的に解決できます。

Web 公開エンジンのベース URI にアクセスするには、XSLT スタイルシートで最も上にある <xsl:template> エレメントの前に、次のステートメントを含めます。

```
<xsl:param name="xml-base-uri"/>
```

その後、FileMaker XML データへの追加のリクエストを実行する必要がある場合は、$xml-base-uri 変数を使用して、現在のスタイルシートのベース URI を使用できます。たとえば、次の追加の XML データに対する要求で、ベース URI を使用することができます。

```
<xsl:variable name="layout_information" select="document(concat($xml-base-uri,'/fmi/xml/FMPXMLLAYOUT.xml?-db=products&-lay=sales&-view'))" />
```

認証済みベース URI 引数の使用

authenticated-xml-base-uri 引数では、client-user-name および client-password 引数の機能と xml-base-uri 引数が組み合わされます。

```
<xsl:param name="authenticated-xml-base-uri"/>
```

この引数は、現在処理中の元のリクエストで指定されているものと同じユーザ名とパスワードが必要な、パスワードで保護された追加の XML ドキュメントをロードする場合に使用します。例については、次のセクション「追加のドキュメントのロード」を参照してください。

この引数ステートメントは、XSLT スタイルシートで最も上にある <xsl:template> エレメントの前に含めます。

client-user-name および client-password 引数の値が空白でない場合、authenticated-xml-base-uri 引数の値は次のようになります。

```
http://username:password@hostname:port
```

client-user-name および client-password 引数の値が空白の場合、authenticated-xml-base-uri 引数の値は、xml-base-uri 引数の値と同じになります。

追加のドキュメントのロード

XSLT スタイルシートの処理中に追加の XML ドキュメントをロードするには、XML ドキュメントへの URI とともに、XSLT の標準の document() 関数を使用します。document() 関数は、要求された XML を、<xsl:variable> エレメントに格納できるノードセットとして返します。

FileMaker データベースのデータが含まれる XML ドキュメントをロードするには、document() 関数を FileMaker クエリーコマンドおよび引数とともに使用します。例：

```
<xsl:variable name="other-data" select="document(concat($xml-base-uri,'/fmi/xml/FMPXMLLAYOUT.xml?-db=products&-lay=sales&-view'))" />
```

現在処理中の元のリクエストで指定されているものと同じユーザ名とパスワードが必要な、パスワードで保護された追加の XML ドキュメントをロードするには、authenticated-xml-base-uri 引数を使用します。この引数を使用することで、document() 関数に渡される URI の一部として同じユーザ名とパスワードが指定されます。

例：

```
<xsl:variable name="other-data" select="document(concat($authenticated-xml-base-uri,'/fmi/xml/FMPXMLLAYOUT.xml?-db=products&-lay=sales&-view'))"/>
```

親リクエストで指定されているものとは異なるユーザ名とパスワードが必要な、パスワードで保護された XML ドキュメントをロードするには、次の構文を使用して、document() 関数に渡される URI の一部としてユーザ名とパスワードを指定します。

```
http://username:password@hostname/path?querystring
```

FileMaker データベースに基づかない XML ドキュメントをロードするには、FileMaker クエリーコマンドまたは引数を設定せずに document() 関数を使用します。例：

```
<xsl:variable name="other-data" select="document('http://server.company.com/data.xml')" />
```

document() 関数を相対 URL とともに使用した場合、Web 公開エンジンは、スタイルシートが保存されている場所に対して相対的な場所にある XML ドキュメントをローカルファイルシステムからロードしようとします。たとえば、「xslt-template-files」フォルダの「mystylesheets」フォルダ内にあるスタイルシートに、相対 URL を使用した次の document() 関数が含まれているとします。

```
<xsl:variable name="mydoc" select="document('mystylesheets/mydoc.xml')" />
```

この場合、Web 公開エンジンは、ローカルファイルシステムの「xslt-template-files」フォルダ内にある「mystylesheets」フォルダから「mydoc.xml 」をロードしようとします。

メモ Web 公開エンジンでは、Web 公開エンジンのベース URI を使用してドキュメントをロードする場合は、HTTP のみがサポートされています。外部サーバーからドキュメントをロードする場合は、HTTP と HTTPS の両方がサポートされています。

スタイルシートでのデータベースのレイアウト情報の使用

FMPXMLLAYOUT 文法を使用してレイアウト情報を要求し、XSLT の document() 関数を使用して変数にロードすることで、FileMaker データベースのレイアウト情報をスタイルシートに組み込むことができます。

```
<xsl:variable name="layout" select="document(concat($xml-base-uri,'/fmi/xml/FMPXMLLAYOUT.xml?-view'))" />
```

たとえば、「Color」という名前のフィールドに対してプルダウンメニューを作成するとします。このフィールドには、FileMaker データベースのレイアウトに定義されている「shirts」という名前の 2 フィールド値一覧が入力されています。2 フィールド値一覧内の最初のフィールドが「Color」の ID 番号（"100" など）を格納し、2 番目のフィールドが「color」の関連付けられた名前（"Light Green" など）を格納するとします。ここで document() 関数を使用して、このレイアウト情報を、DISPLAY 属性と共に XSLT の変数にロードし、2 フィールド値一覧内の 2 つ目のフィールドの値を表示する方法を示します。

```
<xsl:variable name="layout" select="document(concat($xml-base-uri,'/fmi/xml/FMPXMLLAYOUT.xml?-db=products
&-lay=sales&-view'))" />
<select size="1">
    <xsl:attribute name="name">Color</xsl:attribute>
    <option value=""> Select One...</option>
    <xsl:for-each select="$layout/fml:FMPXMLLAYOUT/fml:VALUELISTS/fml:VALUELIST[@NAME = 'shirts']/fml:VALUE">
            <option>
                        <xsl:attribute name="value"><xsl:value-of select="."/></xsl:attribute>
                        <xsl:value-of select="@DISPLAY"/>
        </option>
      </xsl:for-each>
</select>
```

コンテンツバッファの使用

コンテンツバッファが無効の場合、Web 公開エンジンは、XSLT 変換の結果を直接クライアントに戻します。明示的に有効にしない限り、コンテンツバッファは常に無効です。コンテンツバッファを有効にすると、変換されたコンテンツは、変換全体が完了するまで、Web 公開エンジンによって保存されています。

ヘッダを操作する XSLT スタイルシートでは、コンテンツバッファが必要です。ヘッダは応答の本文より先に書き込まれるため、追加されたヘッダ情報を含めることができるように、本文をバッファする必要があります。

XSLT 変換の結果をバッファする必要がある FileMaker 拡張関数は、次の 4 つです。

- fmxslt:create_session(): 58 ページの「セッション拡張関数の使用」を参照してください。
- fmxslt:set_header(): 61 ページの「ヘッダ関数の使用」を参照してください。
- fmxslt:set_status_code (): 61 ページの「ヘッダ関数の使用」を参照してください。
- fmxslt:set_cookie(): 62 ページの「Cookie 拡張関数の使用」を参照してください。

これらの FileMaker 拡張関数を正常に機能させるには、リクエストに対する最上位のドキュメントに次の XSLT 処理命令を含める必要があります。

```
<?xslt-cwp-buffer buffer-content="true"?>
```

重要 他のスタイルシートが含まれるベーススタイルシートを使用している場合、<?xslt-cwp-buffer?> 処理命令は、ベーススタイルシートに含める必要があります。ベーススタイルシートに含まれるスタイルシートでこの命令を使用しても、無視されます。

この処理命令を使用して応答をバッファすると、Web 公開エンジンが応答の長さを判断して、応答に Content-Length ヘッダを設定できるという利点があります。応答をバッファすると、Web 公開エンジンのパフォーマンスが低下する場合があります。

Web 公開エンジンセッションを使用したリクエスト間での情報の保存

Web 公開エンジンのサーバーサイドセッションを使用して、リクエスト間で任意のタイプの情報を追跡および保存できます。セッションを使用すると、持続的な任意の情報をリクエスト間で使用することによって状態を維持できる Web アプリケーションを作成できます。たとえば、最初のフォームページで入力されたユーザクライアント情報をセッションに保存しておき、以降のページで値を入力するために使用できます。

デフォルトでは、Web 公開エンジンは、Cookie を使用してセッション ID を保存します。Cookie が許可されていないクライアントに対応するには、fmxslt:session_encode_url() 関数を使用して、セッション ID を URL に追加できます。すべての状況において互換性を保証するために、ページに書き込む URL は、fmxslt:session_encode_url() 関数を使用してすべてエンコードすることをお勧めします。この関数では、jsessionid というセミコロン区切りの引数が URL に追加されます。この引数は、該当するクライアントの親セッションの ID です。

たとえば、次のリンクがページに配置されているとします。

```
<a href="my_stylesheet.xsl?-db=products&-lay=sales&-grammar=fmresultset&-findall">hyperlinked text</a>
```

この場合、ページ上のすべてのリンクを次のようにエンコードすることをお勧めします。

```
<a href="{fmxslt:session_encode_url('my_stylesheet.xsl?-db=products&-lay=sales&-grammar=fmresultset
&-findall')}">hyperlinked text</a>
```

クライアントで Cookie が許可されていない場合は、次のリンクがページに含められます。

```
<a href="my_stylesheet.xsl;jsessionid=<session id>?-db=products&-lay=sales&-grammar=fmresultset&-findall">
hyperlinked text</a>
```

クライアントで Cookie が許可されていることが Web 公開エンジンによって検出された場合、fmxslt:session_encode_url() 関数は、URL ではなく Cookie にセッション ID を保存します。

メモ　セッション情報は、Web 公開エンジンの再起動後には維持されません。

セッション拡張関数の使用

セッション変数を操作するには、次のセッション関数を使用します。セッションオブジェクトには、文字列、数字、論理値、またはノードセットを保存できます。ノードセットを使用すると、XML のデータ構造体を作成して、リクエスト間でセッションオブジェクトに保存できます。

セッション拡張関数	戻り値のデータ	説明
fmxslt:session_exists(String セッション - 名)	論理値	指定した名前のセッションが存在するかどうかを確認します。
fmxslt:create_session(String セッション - 名)	論理値	指定したセッション名とデフォルトのタイムアウトでセッションを作成します。デフォルトのタイムアウトは、Admin Console を使用して設定します。「FileMaker Server ヘルプ」を参照してください。 メモ　この関数には、<?xslt-cwp-buffer?> 処理命令が必要です。57 ページの「コンテンツバッファの使用」を参照してください。
fmxslt:invalidate_session(String セッション-名)	論理値	セッションをただちに強制的にタイムアウトさせます。
fmxslt:set_session_timeout(String セッション-名 , Number タイムアウト)	論理値	セッションのタイムアウトを設定します（秒単位）。セッションのデフォルトのタイムアウトは、Admin Console を使用して設定します。
fmxslt:session_encode_url(String URL)	文字列	クライアントで Cookie がサポートされていない場合は、セッション ID を含めて URL をエンコードします。その他の場合は、入力 URL を返します。

セッション拡張関数	戻り値のデータ	説明
fmxslt:set_session_object(String セッション-名 , String 名前 , Object 値)	XSLT オブジェクト（数字、文字列、論理値、またはノードセット）	セッション中に XSLT オブジェクト（数字、文字列、論理値、またはノードセット）を保存しておき、後で fmxslt:get_session_object () 関数を使用して取得できます。この関数は、指定したセッションオブジェクト名ですでに保存されているオブジェクトも返します。指定した名前で保存されているオブジェクトがない場合は、ヌルオブジェクトを返します。 メモ set_session_object() 拡張関数は、文字列値のみを格納し、文字列として渡された任意のオブジェクトを解釈します。オブジェクトを文字列に変換できない場合には、セッションには値が格納されず、拡張関数エラーコードが 10100（原因不明のセッションエラー）に設定されます。空の文字列を使用してセッションオブジェクトを設定しようとする場合にも、エラーコード 10100（原因不明のセッションエラー）が返されます。セッション変数を消去するには、remove_session_object() 関数を使用してセッションから変数を削除します。
fmxslt:get_session_object(String セッション - 名 , String 名前)	XSLTオブジェクト	セッションから XSLT オブジェクトを取得します。
fmxslt:remove_session_object(String セッション-名 , String 名前)	XSLTオブジェクト	セッションから XSLT オブジェクトを取得して削除します。

次に、セッションを作成して、お気に入りの色をセッションに保存する例を示します。

```
<xsl:variable name="session">
    <xsl:choose>
        <xsl:when test="not (fmxslt:session_exists(string($session-name)))">
                    <xsl:value-of select="fmxslt:create_session(string($session-name))"/>
        </xsl:when>
        <xsl:otherwise>true</xsl:otherwise>
    </xsl:choose>
</xsl:variable>
<xsl:variable name="favorite-color" select="fmxslt:set_session_object(string($session-name), 'favorite-color', string($color))"/>
```

重要

- セッションの完了後にユーザをデータベースから確実にログアウトするには、fmxslt:invalidate_session () 関数を使用して、セッションをただちに強制的にタイムアウトさせます。
- 状態を設定または変更するグローバルフィールドやスクリプトを使用する場合は、Admin Console を使用して、Web 公開エンジンの XSLT データベースセッションオプションを有効にする必要があります。このオプションが有効でない場合、グローバルフィールドの値や状態は、リクエスト間で維持されません。「FileMaker Server ヘルプ」を参照してください。
- Web 公開エンジンのセッションを使用して別のデータベースファイルに切り替えると、切り替えた場合、グローバルフィールドの値は保持されません。Web 公開エンジンによって、2 番目のファイルを開く前に 1 番目のファイルが閉じます。代わりに、1 番目のデータベースファイルにあるレイアウトを使用することによって、2 番目のデータベースファイルのデータを使用することが可能になります。

Web 公開エンジンからの電子メールメッセージの送信

Web 公開エンジンを使用して電子メールメッセージを生成することができます。これは、カスタム Web ソリューションで便利です。Web 公開エンジンから電子メールメッセージを送信するには、XSLT スタイルシートで、次の 3 つの fmxslt:send_email () 拡張関数の 1 つを使用します。これらの関数を使用して、1 つまたは複数の別個のメッセージを送信できます。fmxslt:send_email () 関数は、Web 公開エンジンのサーバーサイドの XSLT スタイルシートに含まれるため、クライアントが Web 公開エンジンを使用して不正な電子メールメッセージを送信することはできません。

電子メール拡張関数	戻り値のデータ	説明
fmxslt:send_email(String SMTP フィールド , String 本文)	論理値	電子メールメッセージに使用する Web 公開エンジンのデフォルトのテキストエンコードを使用して、Web 公開エンジンから任意の長さのテキストの電子メールメッセージを送信します。
fmxslt:send_email(String SMTP フィールド , String 本文 , String エンコード)	論理値	任意の長さのテキストの電子メールメッセージを、US-ASCII、ISO-8859-1、ISO-8859-15、ISO-2022-JP、Shift_JIS、UTF-8 のいずれかのテキストエンコードを使用して送信します。これらのエンコードの詳細については、52 ページの「リクエストのテキストエンコードの設定」を参照してください。
fmxslt:send_email(String SMTP フィールド , String XSLT ファイル , Node XML, boolean イメージの包含)	論理値	スタイルシートの <xsl:output> エレメントの encoding 属性によって指定されているエンコードを使用して、HTML ベースの電子メールメッセージを送信します。スタイルシートの <xsl:output> エレメントの encoding 属性によって指定されているエンコードを使用して、HTML ベースの電子メールメッセージを送信します。<xsl:output> エレメントに encoding 属性が含まれない場合は、電子メールメッセージに使用する Web 公開エンジンのデフォルトのテキストエンコードが使用されます。

注意

- これら 3 つの形式の各 fmxslt:send_email () 関数では、SMTP フィールド引数には、次の形式を使用するアドレスとトピックの情報が含まれる、URL エンコードされた任意の長さの文字列を指定します。この形式は、RFC 2368 の mailto URL スキームに基づきます。

  ユーザ名 @ ホスト ? 名前 1 = 値 1 & 値 2 = 値 2 ...

  ユーザ名 @ ホストには、受信者を指定します。名前 / 値の組を任意の順序で指定でき、次のように定義されます。

  - from= ユーザ名 @ ホスト (記述するのは 1 回だけにする必要があります)。from フィールドは必ず指定する必要があります。
  - to= ユーザ名 @ ホスト。追加の受信者には、この名前 / 値の組を使用します。
  - reply-to= ユーザ名 @ ホスト (1 回だけ記述できます)。
  - cc= ユーザ名 @ ホスト。
  - bcc= ユーザ名 @ ホスト。
  - subject= 文字列 (1 回だけ記述できます)。

  複数の from、reply-to、または subject フィールドが指定されている場合、電子メールメッセージは送信されず、関数によって値 false() が返され、該当するエラーコードが設定されます。
- 入力されたすべての電子メールアドレスの構文は、Web 公開エンジンによって確認されます。構文は次の形式でなければなりません。

  ユーザ @ ホスト .tld または " ダブルクォーテーションで囲まれた ID"< ユーザ @ ホスト .tld>

  tld には、com や net などの最上位のドメインを指定します。いずれかのフィールドに無効な電子メールアドレスが含まれる場合、電子メールメッセージは送信されず、該当するエラーステータスコードが設定されます。
- subject などの SMTP フィールド引数の個々の値は、URL エンコードされた文字列である必要があります。たとえば、「&」文字は「&」として、スペースは「%20」として指定する必要があります。SMTP フィールド引数の文字列全体が XML エンコードされている必要があります (このセクションの最後にある例を参照してください)。
- これらの各関数に対して、電子メールメッセージが正常に送信された場合は true() の値が返され、その他の場合は false() が返されます。

- 日本語の電子メールメッセージには、Web 公開エンジンによって、デフォルトのテキストエンコードの初期状態の設定である Shift_JIS が使用されます。この設定は、Admin Console を使用して変更できます。「FileMaker Server ヘルプ」を参照してください。
- fmxslt:send_email(String SMTP フィールド , String XSLT ファイル , Node XML, boolean イメージの包含) 関数は、この関数で指定した電子メール用スタイルシートによって処理された XML データで構成される電子メールメッセージを送信します。
  - XSLT ファイル引数には、リクエストに使用する主な処理用スタイルシートファイルに対して相対的な URL を入力して電子メール用スタイルシートの名前を指定します。
  - XML 引数には、電子メール用スタイルシートとともに使用する XML データの親ノードを指定します。ブラウザに表示されている XML データと同じ XML データを使用して電子メールメッセージを送信するには、単にドキュメントのルートの XPath 「/」を指定します。その他の場合は、最初に document() 関数でドキュメントをロードしてから、そのドキュメントを fmxslt:send_email() 関数に渡すことで、異なる XML ドキュメントを使用できます。
  - イメージの包含引数には、電子メールメッセージの HTML で指定されているすべてのイメージを Web 公開エンジンで添付ファイルとして含める場合は、論理値 true() を指定します。この引数では、FileMaker データベース内にあるイメージと、データベース内にない他の場所のイメージの両方が含められます。イメージの URL は、Web 公開エンジンによって、添付ファイルを参照するように変更されます。イメージファイルが多い場合や、大きい場合は、パフォーマンスが低下する可能性があります。false () を指定した場合、イメージの URL は Web 公開エンジンによって変更されません。絶対 URL の場合、電子メールクライアントは、Web サーバーからイメージをロードしようとします。

次に、<xsl:if> エレメントの内部など、XPath ステートメント内で、fmxslt:send_email(String SMTP フィールド , String XSLT ファイル , Node XML, boolean イメージの包含) 関数を使用する場合の例を示します。

```
fmxslt:send_email('tom_jones@company.com?subject=project%20status&amp;from=john_smith@company.com
&amp;cc=jane_doe@company.com','my_mail_template.xsl', /, true())
```

SMTP サーバーに接続するように Web 公開エンジンを設定する場合の詳細については、「FileMaker Server ヘルプ」を参照してください。

ヘッダ関数の使用

fmxslt:get_header() 関数を使用して、HTTP リクエストおよび応答のヘッダから情報を読み取り、fmxslt:set_header() 関数を使用して、これらのヘッダに情報を書き込むことができます。これらの関数は、クライアントがヘッダ情報を使用して Web サーバーから情報を取得できる場合や、他の理由で HTTP ヘッダを設定する必要がある場合に便利です。

ヘッダ拡張関数	戻り値のデータ	説明
fmxslt:get_header(String 名前)	文字列	指定したヘッダの値を返します。
fmxslt:set_header(String 名 , String 値)	void	指定したヘッダの値を設定します。
fmxslt:set_status_code(Number ステータスコード)	void	HTTP ステータスコードを設定します。

注意

- fmxslt:get_header() と fmxslt:set_header() 関数で使用する名前、および fmxslt:set_header() 関数の値には、任意の長さの文字列を指定できます。
- fmxslt:set_header() 関数および fmxslt:set_status_code() 関数には、<?xslt-cwp-buffer?> 処理命令が必要です。57 ページの「コンテンツバッファの使用」を参照してください。

次の例に、ヘッダの値を設定する方法を示します。ここでは、スタイルシートを使用して vCard を出力するとします。この場合、ブラウザがスタイルシートのページをロードするときに、.xsl ファイルがブラウザによって vCard ではなくスタイルシートとして解釈されるという問題が発生する可能性があります。Content-Disposition というヘッダを使用すると、拡張子 .vcf の付いた添付ファイルが存在することを指定できます。

```
<xsl:value-of select="fmxslt:set_header('Content-Disposition','attachment;filename=test.vcf')" />
```

Cookie 拡張関数の使用

Cookie 拡張関数を使用して、クライアントの Web ブラウザに保存される Cookie を取得または設定できます。

Cookie 拡張関数	戻り値のデータ	説明
fmxslt:get_cookie(String 名前)	ノードセット	指定した Cookie 名を持つ COOKIES ノードリストを返します。
fmxslt:get_cookies()	ノードセット	クライアントによって提供された Cookie がすべて含まれる COOKIES ノードセットを返します。
fmxslt:set_cookie(String 名 , String 値)	void	指定した Cookie を、指定した値でクライアントのブラウザに保存します。
fmxslt:set_cookie(String 名 , String 値 , Number 有効期限 , String パス , String ドメイン)	void	指定した Cookie を、Cookie で使用可能なすべての値でクライアントのブラウザに保存します。有効期限引数には、Cookie の有効期限が切れるまでの秒数を指定します。

注意

- fmxslt:get_cookie() および fmxslt:get_cookies() 関数は、次の構造のノードセットを返します。

```
<!ELEMENT cookies (cookie)*>
    <!ATTLIST cookie xmlns CDATA #FIXED "http://www.filemaker.com/xml/cookie">
<!ELEMENT cookie (#PCDATA)>
    <!ATTLIST cookie name CDATA #REQUIRED>
```

- Cookie ノードセットの XML ネームスペースは、"http://www.filemaker.com/xml/cookie" です。このネームスペースを宣言し、ネームスペースには接頭語を指定する必要があります。
- fmxslt:set_cookie 関数のすべての引数値が有効である必要があります。有効でない場合、Web ブラウザでは、fmxslt:set_cookie 関数のリクエストは無視されます。
- すべての Cookie 関数では、文字列引数は任意の長さにできます。
- fmxslt:set_cookie() 関数の両方の形式には、<?xslt-cwp-buffer?> 処理命令が必要です。57 ページの「コンテンツバッファの使用」を参照してください。

例：get_cookie

次の例では、「preferences」という名前の Cookie とその値を抽出します。

```
<xsl:variable name="pref_cookie" select="fmxslt:get_cookie('preferences')"/>
<xsl:value-of select="concat('Cookie Name = ', $pref_cookie/fmc:cookies/fmc:cookie/@name)"/> <br/>
<xsl:value-of select="concat('Cookie Value = ', $pref_cookie/fmc:cookies/fmc:cookie)"/>
```

例：set_cookie

次に、すべての値を使用して Cookie を設定する方法の例を示します。

```
<xsl:variable name="storing_cookie" select="fmxslt:set_cookie ('text1', 'text2', 1800, 'my_text', 'my.company.com')" />
```

文字列操作拡張関数の使用

文字列操作関数を使用して、任意の長さの文字列のエンコードを変更できます。

文字列操作拡張関数	戻り値のデータ	説明
fmxslt:break_encode(String 値)	文字列	改行などが HTML エンコードされた文字列を返します。「&」（アンパサンド）などの文字は「&」などに、行送りや改行などの復帰改行文字は にそれぞれ置き換えられます。この関数は、<xsl:value-of> および <xsl:text> エレメントの disable-output-escaping 属性が「yes」に設定されている（disable-output-escaping="yes"）場合にのみ動作します。 メモ fmxslt:break_encode() 関数が適用される文字列に行送りまたは改行を含めるには、文字列内でエスケープ文字「
」（行送り）または「
」（改行）を使用する必要があります。テキストエディタで Enter キー（Window）または return キー（Mac OS）を押して文字列に行送りまたは改行を含めることはできません。
fmxslt:html_encode(String 値)	文字列	HTML エンコードされた文字列を返します。「&」（アンパサンド）などの文字は、「&」などに置き換えられます。
fmxslt:url_encode(String 値)	文字列	URL エンコードされた文字列を返します。URL エンコードは、インターネット上、特に URL で文字を転送するために使用されます。たとえば、URL エンコードされた形式の「&」（アンパサンド）は、「%26」になります。予約済みの文字が href で使用されている場合は、この関数を使用して、文字列を URL エンコードします。
fmxslt:url_encode(String 値 , String エンコード)	文字列	encoding 引数に指定した文字エンコードを使用して、URL エンコードされた文字列を返します。US-ASCII、ISO-8859-1、ISO-8859-15、ISO-2022-JP、Shift_JIS、または UTF-8 を指定できます。 この関数は、Web サーバーで、現在のスタイルシートで使用されている文字コードと異なる文字エンコードが使用されることがわかっている場合に使用します。たとえば、Web サイトの最初のページは UTF-8 で表示されていても、ユーザがリンクをクリックして日本語のページに移動する場合があります。リクエストに日本語の文字が含まれていて、日本語のページで Shift_JIS エンコードが使用されている場合は、文字列を Shift_JIS でエンコードすることをお勧めします。
fmxslt:url_decode(String 値)	文字列	すでにエンコードされている URL 文字列から URL デコードされた文字列を返します。
fmxslt:url_decode(String 値 , String エンコード)	文字列	encoding 引数に指定した文字エンコードを使用して、URL エンコードされた文字列を返します。US-ASCII、ISO-8859-1、ISO-8859-15、ISO-2022-JP、Shift_JIS、または UTF-8 を指定できます。 この関数は、URL エンコードされた文字列を正しくデコードするために、文字列で使用されている文字エンコードを指定する必要がある場合に使用します。たとえば、Web サイトで ISO-8859-1 が使用されていても、ユーザは、異なる文字エンコードを使用してフォームを送信できます。

Perl 5 の正規表現を使用した文字列の比較

fmxslt:regex_contains() 拡張関数を使用すると、Perl 5 の正規表現を使って文字列を比較できます。正規表現による比較は高度な種類のテキスト照合で、文字列が指定したパターンに一致するかどうかを判断できます。この関数の構文は次の通りです。

fmxslt:regex_contains(String input, String pattern)

input には文字列を、pattern には Perl 5 の正規表現をそれぞれ指定します。Perl 5 の正規表現の構文の詳細については、www.perldoc.com を参照してください。fmxslt:regex_contains() 関数は、論理値を返します。

この関数は、標準の XSLT によって提供されている機能よりも高度な文字列操作が必要な場合に便利です。たとえば、Perl 5 の正規表現で文字列を比較することで、フィールドの値に有効な電話番号や電子メールアドレスが含まれているかどうかを判断できます。

次に、この関数を使用して、フィールドの値に正しい形式の電子メールアドレスが含まれているかどうかを判断する場合の例を示します。

```
<xsl:variable name="email" select="'foo@bar.com'"/>
<xsl:if test="fmxslt:regex_contains($email,'^\w+[\w-\.]*\@\w+((-\w+)|(\w*))\.[a-z]{2,3}$')">Valid Email</xsl:if>
```

Web 公開エンジンがこのパターンを解析できない場合は、エラーステータスがエラーコード 10311 に設定されます。101 ページの「FileMaker XSLT 拡張関数のエラーコード番号」を参照してください。

チェックボックスとして書式設定されたフィールドの値の確認

次の拡張関数を使用して、チェックボックス形式の値一覧の特定の値が FileMaker データベースのフィールドに保存されているかどうかを判断します。

```
fmxslt:contains_checkbox_value(String valueString, String valueListEntry)
```

valueString にはフィールドを指定する XPath を、valueListEntry には確認する値をそれぞれ指定します。

指定した値がフィールドに保存されている場合、この論理関数は true() を返します。その他の場合は、false() を返します。この関数を使用して、HTML フォームの checked 属性を設定し、チェックボックスを選択された状態で表示するかどうかを決定できます。

たとえば、FileMaker データベースのレイアウトのフィールドに、次のチェックボックスオプションが設定されているとします。

[ ] Red
[ ] Blue
[ ] Green
[ ] Small
[ ] Medium
[ ] Large

ユーザが [Red] のみを選択している場合、フィールドには文字列「Red」が含まれています。フィールドに「Blue」が含まれるかどうかを判断するために、次の関数呼び出しを使用できます。

```
fmxslt:contains_checkbox_value(< フィールド値のノード >,'Blue')
```

< フィールド値のノード > には、チェックボックスフィールドの <data> エレメントへの XPath を指定します。この例では、この関数は「false」を返します。

この関数の一般的な応用は、Web ページにチェックボックスの値一覧を表示して、データベースで選択されているチェックボックスを Web ページでも選択する場合です。たとえば、次の HTML および XSLT ステートメントを使用して、「color_size」という名前の値一覧を使用する「style」というフィールドに対して、チェックボックスのセットを作成します。

```
<xsl:variable name="field-value" select="fmrs:field[@name='style']/fmrs:data" />
<xsl:for-each select="$valuelists[@NAME = 'color_size']/fml:VALUE">
    <input type="checkbox">
        <xsl:attribute name="name">style</xsl:attribute>
        <xsl:attribute name="value"><xsl:value-of select="."/></xsl:attribute>
        <xsl:if test="fmxslt:contains_checkbox_value($field-value,.)">
                <xsl:attribute name="checked">checked</xsl:attribute>
        </xsl:if>
    </input><xsl:value-of select="."/><br/>
</xsl:for-each>
```

この例の HTML および XSLT ステートメントでは、次のチェックボックス、[Red] および [Medium] が選択された状態で Web ページ上に出力されます。

[x] Red
[ ] Blue
[ ] Green
[ ] Small
[x] Medium
[ ] Large

日付、時刻、および曜日拡張関数の使用

現在の日付、時刻、または曜日を取得したり、２つの日付、時刻、または曜日を比較する拡張関数を使用することができます。

次の表の関数では、ロケールに関係なく「fm」書式が使用されます。「fm」形式では、日付は MM/dd/yyyy、時刻は HH:mm:ss、タイムスタンプは MM/dd/yyyy HH:mm:ss です。

出力された値を異なる形式または優先形式に並べ替えるには、計算関数または JavaScript を使用します。

日付、時刻、および曜日拡張関数	戻り値のデータ	説明
fmxslt:get_date()	文字列	現在の日付を「fm」書式で返します。
fmxslt:get_date(String 書式)	文字列	現在の日付を指定した書式で返します。書式引数には、“short”、“long”、または “fm” を入力します。
fmxslt:get_time()	文字列	現在の時刻を「fm」書式で返します。
fmxslt:get_time(String 書式)	文字列	現在の時刻を指定した書式で返します。書式引数には、“short”、“long”、または “fm” を入力します。
fmxslt:get_day()	文字列	現在の曜日を短い書式で返します。
fmxslt:get_day(String 書式)	文字列	現在の曜日を指定した書式で返します。書式引数には、“short” または “long” を入力します。
fmxslt:get_fm_date_format()	文字列	「fm」日付書式の形式文字列「MM/dd/yyyy」
fmxslt:get_short_date_format()	文字列	短い日付書式の形式文字列「M/d/yy」
fmxslt:get_long_date_format()	文字列	長い日付書式の形式文字列「MMM d, yyyy」
fmxslt:get_fm_time_format()	文字列	「fm」時刻書式の形式文字列「HH:mm:ss」
fmxslt:get_fm_timestamp_format()	文字列	「fm」タイムスタンプ書式の形式文字列「MM/dd/yyyy HH:mm:ss」
fmxslt:get_short_time_format()	文字列	短い時刻書式の形式文字列「h:mm a」
fmxslt:get_long_time_format()	文字列	長い時刻書式の形式文字列「h:mm:ss a z」
fmxslt:get_short_day_format()	文字列	短い曜日書式の形式文字列「EEE」
fmxslt:get_long_day_format()	文字列	長い曜日書式の形式文字列「EEEE」
fmxslt:compare_date(String 日付１ , String 日付２)	数字	この関数は、２つの日付の値を比較します。日付１が日付２より前の場合は負の数を返し、日付１が日付２より後の場合は正の数を返します。日付１と日付２が同一の場合は０を返します。両方の日付を「fm」日付書式で指定する必要があります。

日付、時刻、および曜日拡張関数	戻り値のデータ	説明
fmxslt:compare_time(String 時刻１ , String 時刻２)	数字	この関数は、２つの時刻の値を比較します。時刻１が時刻２より前の場合は負の数を返し、時刻１が時刻２より後の場合は正の数を返します。時刻１と時刻２が同一の場合は０を返します。両方の時刻を「fm」時刻書式で指定する必要があります。
fmxslt:compare_day(String 曜日１ , String 曜日２)	数字	この関数は、２つの曜日の値を比較します。曜日１が曜日２より前の場合は負の数を返し、曜日１が曜日２より後の場合は正の数を返します。曜日１と曜日２が同一の場合は０を返します。両方の曜日を短い曜日書式で指定する必要があります。

次の表に示す関数は、日付と時刻の書式を指定するカスタム日付形式文字列を使用します。次のセクション「日付と時刻の形式文字列について」を参照してください。

日付、時刻、および曜日拡張関数	戻り値のデータ	説明
fmxslt:get_datetime(String 日付書式)	文字列	日付および時刻の形式文字列を使用して、現在の日付と時刻を返します。
fmxslt:convert_datetime(String 古い書式 , String 新しい書式 , String 日付)	文字列	指定した古い書式の指定した日付を、指定した新しい書式に従った文字列に変換します。古い書式と新しい書式の文字列は、日付と時刻の形式文字列を使用して指定する必要があります。
fmxslt:compare_datetime(String 日付書式１ , String 日付書式２ , String 日付１ , String 日付２)	数字	この関数は、日付１と日付２を各日付書式に従ってデコードすることによって、これらの２つの日付を比較します。日付１が日付２より前の場合は負の数を返し、日付１が日付２より後の場合は正の数を返します。日付１と日付２が同一の場合は０を返します。日付書式１と日付書式２の文字列は、日付と時刻の形式文字列を使用して指定する必要があります。

日付と時刻の形式文字列について

日付と時刻の書式は、日付と時刻のパターン文字列で指定します。日付と時刻のパターン文字列内では、クォートで囲まれていない「A」から「Z」および「a」から「z」の文字は、日付または時刻の文字列の構成部分を表すパターン文字として解釈されます。

次のパターン文字が定義されています（「A」から「Z」および「a」から「z」の他の文字もすべて予約されています）。

文字	日付または時刻の構成要素	表示	例
G	年号指定子	テキスト	西暦
y	年	年	1996; 96
M	年を基準とした月	月	7 月 ; 7 ; 07
w	年を基準とした週	数字	27
W	月を基準とした週	数字	2
D	年を基準とした日	数字	189
d	月を基準とした日	数字	10
F	月を基準とした週の日	数字	2
E	曜日	テキスト	火曜日 ; 火
a	AM/PM のマーカ	テキスト	午後
H	日を基準とした時間（0 から 23）	数字	0
k	日を基準とした時間（1 から 24）	数字	24
K	AM/PM での時間（0 から 11）	数字	0
h	AM/PM での時間（1 から 12）	数字	12

文字	日付または時刻の構成要素	表示	例
m	時を基準とした分	数字	30
s	分を基準とした秒	数字	55
S	ミリ秒	数字	978
z	時間帯	一般的な時間帯	太平洋夏時間 ; PST
Z	時間帯	RFC 822 の時間帯	-0800

パターン文字の数によって正確な表現が決まるため、パターン文字は、通常は繰り返し使用されます。

- テキスト：書式設定時には、パターン文字の数字が 4 つ以上の場合、完全な書式が使用されます。パターン文字が 3 つ以下の場合、使用可能であれば、短い書式または短縮形の書式が使用されます。解析時には、パターン文字の数に応じて両方の書式を使用できます。
- 数字：書式設定時には、パターン文字の数は最小の桁数になり、それよりも短い数字には、この量になるまでゼロが追加されます。解析時には、パターン文字の数は、連続する 2 つのフィールドを分離するために必要な場合以外は無視されます。
- 年：書式設定時には、パターン文字の数が 2 つの場合、年は 2 桁に切り詰められます。その他の場合は、数字として解釈されます。

  解析時には、パターン文字の数が 2 より多い場合、年は、桁数にかかわらず文字どおりに解釈されます。したがって、「yyyy/MM/dd」というパターンを使用すると、「12/01/11」は、西暦 12 年 1 月 11 日に解釈されます。

  省略形の年のパターン（「y」または「yy」）を使用した解析時には、省略形の年は、日付書式のインスタンスが作成された時点の 80 年前または 20 年後に日付を調整することによって、特定の世紀に対して相対的に解釈されます。たとえば、「yy/MM/dd/」というパターンと、1997 年 1 月 1 日に作成された日付書式のインスタンスを使用した場合、文字列「12/01/11」は 2012 年 1 月 11 日に解釈されますが、文字列「64/05/04」は 1964 年 5 月 4 日に解釈されます。解析時には、ちょうど 2 桁で構成される文字列のみがデフォルトの世紀に解析されます。 1 桁の文字列、 3 桁以上の文字列、または一部が数字ではない 2 桁の文字列（例：「-1」）など、他の数値文字列は、文字どおりに解釈されます。したがって、「3/01/02」または「003/01/02」は、同じパターンを使用して、西暦 3 年 1 月 2 日と解釈されます。同様に、「-3/01/02」は、紀元前 4 年 1 月 2 日と解釈されます。
- 月：パターン文字の数が 3 つ以上の場合、月はテキストとして解釈されます。その他の場合は、数字として解釈されます。
- 一般的な時間帯：時間帯は、名前がある場合はテキストとして解釈されます。GMT からのオフセットの値を表す時間帯に対しては、次の構文が使用されます。
  - GMT からのオフセットの時間：GMT 記号 時 : 分
  - 記号：+ または -
  - 時：数字または数字 数字
  - 分：数字 数字
  - 数字：0 1 2 3 4 5 6 7 8 9 のいずれか

  時は 0 から 23 の数字、分は 00 から 59 の数字でなければなりません。書式はロケールに依存せず、数字は Unicode 標準の Basic Latin ブロックから使用する必要があります。

  解析時には、RFC 822 の時間帯も使用できます。
- RFC 822 の時間帯：書式設定時には、RFC 822 の 4 桁の時間帯書式が使用されます。
  - RFC822 の時間帯：記号 2 桁の時 分
  - 2 桁の時：数字 数字

  2 桁の時は 00 から 23 でなければなりません。他の定義は、一般的な時間帯用です。

  解析時には、一般的な時間帯も使用できます。

次の例は、日本のロケールにおいて日付と時刻のパターンがどのように解釈されるかを示します。指定されている日付と時刻は、米国の太平洋標準時の時間帯のローカル時間 2001-07-04 12:08:56 です。

日付と時刻のパターン	結果
"yyyy.MM.dd G 'at' HH:mm:ss z"	2001.07.04 西暦 at 12:08:56 PDT
""EEE, MMM d, ' 'yy"	水 , 7 4, '01
"h:mm a"	12:08 午後
"hh 'o' 'clock' a, zzzz"	12 o'clock 午後 , 太平洋夏時間
"K:mm a, z"	0:08 午後 , PDT
"yyyyy.MMMMM.dd GGG hh:mm aaa"	02001.7 月 .04 西暦 12:08 午後
"EEE, d MMM yyyy HH:mm:ss Z"	水 , 4 7 2001 12:08:56 -0700
"yyMMddHHmmssZ"	010704120856-0700

拡張関数のエラーステータスのチェック

XSLT スタイルシート内で次の拡張関数を使用して、直前に呼び出された FileMaker XSLT 拡張関数の現在のエラーステータスをチェックして、ページの処理中に発生したエラーを処理できます。

fmxslt:check_error_status()

fmxslt:check_error_status() 関数が呼び出されると、Web 公開エンジンは、直前に呼び出された関数の現在のエラーコードの値を数字タイプとして返し、エラーステータスを 0（「エラーなし」）に設定します。エラーコードの値の詳細については、101 ページの「FileMaker XSLT 拡張関数のエラーコード番号」を参照してください。

ログの使用

標準の XSLT <xsl:message> エレメントを使用して、Web 公開エンジンのアプリケーションログファイルにログエントリを書き込むことができます。76 ページの「Web 公開エンジンのアプリケーションログの使用」を参照してください。

スクリプト言語のサーバーサイドでの処理の使用

Web 公開エンジンに埋め込まれている下層の XSLT トランスフォーマは、スクリプト言語のサーバーサイドでの処理をサポートします。そのため、JavaScript を使用して、XSLT スタイルシートから直接呼び出すことができるユーザ独自の拡張関数を開発することができます。

この機能を有効にするため、次の 2 つの Java ライブラリがインストールされています。

- bsf.jar - このライブラリを使用すると、XSLT トランスフォーマをスクリプト言語に接続できます。
- js.jsr - このライブラリは、Mozilla プロジェクトからの JavaScript の完全実装です。

これらのライブラリを使用して、ユーザの XSLT スタイルシートコードの内部にユーザ独自の拡張関数を作成できます。これらの拡張関数によって、任意のスクリプトのロジックを実装でき、これらの拡張関数は論理関数を作成するには XSLT および XPath よりも容易な管理が可能です。

XSLT トランスフォーマの拡張サポートの詳細については、次の Apache Xalan Extensions の Web サイトを参照してください。

http://xml.apache.org/xalan-j/extensions.html

拡張関数の定義

スタイルシートの内部に拡張関数を定義するには、次の操作を行います。

1. 拡張用のネームスペースを定義します。

   xalan ネームスペースを追加して、XSLT トランスフォーマに対し、拡張コンポーネントをサポートするように指示します。これにより、ユーザ独自の拡張関数ネームスペースに名前が与えられます。次の例では、拡張関数ネームスペースの接頭語として fmp-ex を使用します。

```
<xsl:stylesheet version="1.0"
xmlns:xsl=http://www.w3.org/1999/XSL/Transform
xmlns:xalan=http://xml.apache.org/xslt
xmlns:fmp-ex="ext1"
exclude-result-prefixes="xsl xalan fmp-ex">
```

2. 拡張コンポーネントと拡張関数を定義します。このとき、拡張関数を実際に実装するコードを使用します。

```
<xalan:component prefix="fmp-ex" functions="getValueColor">
<xalan:script lang="javascript">
    function getValueColor(value) {
                if (value > 0)
                    return ("#009900");
                else
                    return ("#CC0000");
    }
</xalan:script>
</xalan:component>
```

   この例では、入力値に基づいて色の値が返されます。入力値が 0 よりも大きい場合には、返される色は緑（"#009900"）です。それ以外の入力値が 0 より小さい場合には、返される色は赤（"#CC0000"）です。

   メモ <xalan:component> エレメントは、<xsl:stylesheet> エレメントの子である必要があります。

3. スタイルシートの内部で拡張関数を使用します。

   次の例で、XPath ステートメントを使用して拡張関数を呼び出す方法を示します。

   この 1 番目の例では、フォントの色を緑（"#009900"）に設定します。

```
<font color=”{fmp-ex:getValueColor(50)}”> 値は 50 です </font>
```

   この 2 番目の例では、フォントの色を赤（"#CC0000"）に設定します。

```
<font color=”{fmp-ex:getValueColor(-500)}”> 値は -500 です </font>
```

拡張関数の例

上記の処理で使用される簡単な JavaScript 関数は、<xsl:choose> ステートメントを使用して実装できたと思われます。スクリプト拡張を使用する本当の利点は、XSLT や XPath だけでは実装できない関数を作成できることです。

たとえば、ユーザ所属の企業用の内部ポータルサイトを構築し、そのポータルページで現在の株価情報を載せるとします。XML ストックフィードを使用できますが、これらへのアクセスには一般的に、商用ライセンスが必要です。ただし、Yahoo! Web サイトからコンマ区切り値（CSV）ドキュメント形式で株価データをダウンロードすることができます。XPath document() 関数では、XML ソースからコンテンツをインポートできますが、CSV コンテンツを XML に変換する必要があります。 1 つの解決策として、JavaScript を使用して CSV の株価情報をダウンロードし、そのファイルを解析してデータを抽出する方法があります。

次の URL は、Yahoo! Web サイトから CSV ファイルとして株価情報を取得するための構文を示します。

```
http://quote.yahoo.com/d/quotes.csv?s=< ティッカ >&f=l1gh&e=.csv
```

ここで、< ティッカ > は、データを取得しようとする株式コードを示します。

返されるデータは、次のようなコンマ区切りの 3 つの数字です。

```
31.79,31.17,32.12
```

ここで 1 番目の値は最後の取引価格であり、 2 番目の値はその日の最低価格であり、 3 番目の値はその日の最高価格です。

次の例では、現在の株価情報を Yahoo! Web サイトから取得してこれを XPath 関数を介して使用可能にさせる JavaScript XSLT 拡張関数を示します。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet
    exclude-result-prefixes="xsl fmxslt fmrs xalan fmp-ex"
    version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:fmrs="http://www.filemaker.com/xml/fmresultset"
    xmlns:fmxslt="xalan://com.fmi.xslt.ExtensionFunctions"
    xmlns:xalan="http://xml.apache.org/xslt"
    xmlns:fmp-ex="ext1"
>

<?xslt-cwp-query params="-grammar=fmresultset&-process" ?>
<xsl:output method="html"/>
<xalan:component prefix="fmp-ex" functions="include get_quote" >
<xalan:script lang="javascript">
    function include(url) {
        var dest = new java.net.URL(url);
        var dis = new java.io.DataInputStream(dest.openStream());
        var res = "";
        while ((line = dis.readLine()) != null)
        {
                    res += line + java.lang.System.getProperty("line.separator");
        }
        dis.close();
        return res;
    }
    function get_quote(ticker) {
        url = "http://quote.yahoo.com/d/quotes.csv?s=″+
                    "+ticker+"&amp;f=l1gh&amp;e=.csv";
        csv_file = include(url);
        var str_tokenizer = new java.util.StringTokenizer(csv_file, ',');
        // the first token is the last trade price
        var last = str_tokenizer.nextToken();
        return last;
    }
</xalan:script>
</xalan:component>

<xsl:template match="/fmrs:fmresultset">
    <html>
        <body>
        <font size="2" face="verdana, arial">
        Apple 株価 :<xsl:value-of select="fmp-ex:get_quote('AAPL')"/>
        </font>
        </body>
    </html>
</xsl:template>

</xsl:stylesheet>
```

Web 公開エンジンがこのスタイルシートを処理するときには、Yahoo! Web サイト からの株価情報が必要です。get_quote() 関数によって、株価情報のデータが解析され、そのデータがスタイルシートに返されます。変換された出力が、ブラウザに表示されます。

第 7 章
サイトのステージング、テスト、および監視

この章では、カスタム Web 公開サイトを運用環境に展開する前にステージングおよびテストを行う手順について説明します。テスト中または展開後にログファイルを使用してサイトを監視する手順についても説明します。

カスタム Web 公開サイトのステージング

サイトを正しくテストする前に、必要なファイルを、ステージングサーバーの正しい場所に移動またはコピーする必要があります。

サイトをステージングして、テスト用に準備するには、次の操作を行います。

1. 第 3 章「データベースのカスタム Web 公開の準備」で説明するすべての手順を行います。
2. XSLT および XML が有効になっており、FileMaker Server Admin Console で正しく設定されていることを確認します。

   メモ　手順については、「FileMaker Server ヘルプ」を参照してください。
3. Web サーバーおよび Web 公開エンジンが実行されていることを確認します。
4. Web 公開エンジンが存在しているコンピュータに、XSLT スタイルシートをコピーまたは移動します。

   Web 公開エンジンマシンの次の場所に、XSLT スタイルシートをコピーまたは移動します。

   - Apache（Mac OS）：/ ライブラリ /FileMaker Server/Web Publishing/xslt-template-files
   - IIS（Windows）：< ドライブ >:¥ Program Files ¥ FileMaker ¥ FileMaker Server ¥ Web Publishing ¥ xslt-template-files

   ここで、< ドライブ > は、システムを起動する第 1 ドライブです。

   メモ　スタイルシートは、「xslt-template-files」フォルダ内のオプションのフォルダまたはフォルダ階層に保存することもできます。
5. Web サーバーマシンに、参照されているオブジェクトをコピーまたは移動します。

   データベースオブジェクトフィールドに実際のファイルではなくファイル参照が保存されている場合、レコードを作成または編集するときに、その参照されているオブジェクトが FileMaker Pro の「Web」フォルダに保存されている必要があります。サイトをステージングするには、参照されているオブジェクトを、Web サーバーソフトウェアのルートフォルダ内の同じ相対パスの場所にコピーまたは移動します。

   メモ　データベースファイルが FileMaker Server 展開のデータベースサーバーコンポーネントに適切にホストされていてアクセス可能であり、FileMaker データベースのオブジェクトフィールドに実際のファイルが保存されている場合には、オブジェクトフィールドの内容を移動する必要はありません。

6. 次の URL 構文を使用して XSLT スタイルシートを要求および処理し、表示される HTML を生成します。

< スキーム >://< ホスト >[:< ポート >]/fmi/xsl/< パス >/< スタイルシート >.xsl[?< クエリー文字列 >]

各要素の意味は、次のとおりです。

- < スキーム > は、HTTP または HTTPS プロトコルです。
- < ホスト > には、Web サーバーがインストールされているホストコンピュータの IP アドレスまたはドメイン名を指定します。
- < ポート > には、Web サーバーのポートを指定します（オプション）。ポートが指定されていない場合は、プロトコルのデフォルトのポート（HTTP ではポート 80、HTTPS ではポート 443）とされます。
- < パス > はオプションで、XSLT スタイルシートが保存されている「xslt-template-files」フォルダ内のフォルダを指定します。
- < スタイルシート > は、スタイルシートの名前と拡張子 .xsl です。
- < クエリー文字列 > には、カスタム Web 公開 with XSLT で使用する 1 つのクエリーコマンドと 1 つまたは複数のクエリー引数の組み合わせを指定することができます。

指定したスタイルシートに <?xslt-cwp-query?> 処理命令が含まれる場合は、URL クエリー文字内の一致するクエリーコマンドよりも、静的に定義されているクエリーコマンドと引数が優先されます。XSLT Site Assistant によって生成される「home.xsl」スタイルシートを参照する場合は、クエリー文字列を含める必要はありません。クエリー文字列の詳細については、付録 A「クエリー文字列で使用される有効な名前」を参照してください。

クエリーコマンドおよび引数を含め、URL 構文では、クエリー文字列の部分を除いて大文字と小文字が区別されます。大部分の URL は小文字です。たとえば、スタイルシート（「home.xsl」スタイルシートを含む）を「xslt-template-files」フォルダ内の「my_templates」フォルダにコピーした場合は、次の URL を使用してスタイルシートを要求および処理することができます。

http://192.168.123.101/fmi/xsl/my_templates/home.xsl

メモ Web 公開エンジンでは、Web ユーザが「xslt-template-files」フォルダにインストールされている XSLT スタイルシートのソースを表示することは許可されません。Web ユーザがスタイルシートを処理するリクエストを送信した場合、Web 公開エンジンは、XSLT Site Assistant のスタイルシートの結果である HTML ページのみを Web ブラウザに送信します。

カスタム Web 公開サイトのテスト

カスタム Web 公開サイトが使用可能であることをユーザに通知する前に、そのサイトが意図どおりに表示され、機能することを確認してください。

- レコードの検索、追加、削除、およびソートなどの機能を異なるアカウントとアクセス権セットでテストする。
- 異なるアカウントでログインして、さまざまなアクセス権セットが意図したとおりに動作することを確認する。権限のないユーザがデータにアクセスしたり、データを変更することができないようにしてください。
- すべてのスクリプトをチェックして、結果が意図したとおりであることを確認する。Web で安全に使用できるスクリプトの設計の詳細については、21 ページの「FileMaker スクリプトとカスタム Web 公開」を参照してください。
- 異なるオペレーティングシステムや Web ブラウザを使ってサイトをテストする。

メモ ネットワークに接続されていない場合でも、Web サーバー、Web 公開エンジン、および FileMaker Server が 1 つのコンピュータ展開を使用してインストールされていれば、URL に“http://localhost/”または“http://127.0.0.1/”と入力することで、カスタム Web 公開サイトをテストできます。URL 構文の詳細については、32 ページの「Web 公開エンジンから XML データにアクセスするための一般的な手順」および 48 ページの「FileMaker XSLT スタイルシートの URL 構文について」を参照してください。

XML 出力をテストするためのスタイルシートの例

次に、XML 出力をテストする場合に役立つ XSLT スタイルシートの例を 2 つ示します。

- 次のスタイルシートの例では、要求された XML データを、変換を行わずに出力します。このスタイルシートは、Web 公開エンジンによって使用される実際の XML データを表示する場合に便利です。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:fmrs="http://www.filemaker.com/xml/fmresultset">
    <xsl:output method="xml"/>
    <xsl:template match="/">
        <xsl:copy-of select="."/>
    </xsl:template>
</xsl:stylesheet>
```

- スタイルシートをデバッグする場合は、次の例の HTML <textarea> タグを使用して、スタイルシートによってアクセスされる XML ソースドキュメントを、スクロールするテキスト領域に表示することができます。同じページで、変換された XSLT 結果を変換前の XML ソースドキュメントに照らして比較できます。

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0"
    xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:fmrs="http://www.filemaker.com/xml/fmresultset">
    <xsl:output method="html"/>
<html>
    <body>
    <xsl:template match="/fmrs:fmresultset">
        <textarea rows="20" cols="100">
                    <xsl:copy-of select="."/>
        </textarea><br/>
    </xsl:template>
    </body>
</html>
</xsl:stylesheet>
```

サイトの監視

次のタイプのログファイルを使用して、カスタム Web 公開サイトを監視し、サイトにアクセスした Web ユーザに関する情報を収集することができます。

- Web サーバーのアクセスログとエラーログ
- Web 公開エンジンのアプリケーションログ
- Web サーバーモジュールのエラーログ
- Web 公開コアの内部アクセスログ

Web サーバーのアクセスログとエラーログの使用

Apache（Mac OS）：Apache Web サーバーでは、アクセスログファイルとエラーログファイルが生成されます。Apache アクセスログファイルは、デフォルトでは W3C Common Logfile Format の形式で、Web サーバーへのすべての着信 HTTP リクエストの記録です。Apache エラーログは、HTTP リクエストの処理に関係する問題の記録です。これらのログファイルの詳細については、Apache Web サーバーのマニュアルを参照してください。

IIS（Windows）：Microsoft IIS Web サーバーではアクセスログファイルが生成され、エラーは、ログファイルに書き込まれるのではなく、Windows イベント ビューアに表示されます。アクセスログファイルは、デフォルトでは W3C Extended Log File Format の形式で、Web サーバーへのすべての着信 HTTP リクエストの記録です。アクセスログには W3C Common Logfile Format を使用することもできます。詳細については、Microsoft IIS Web サーバーのマニュアルを参照してください。

W3C Common Logfile Format および W3C Extended Log File Format の詳細については、World Wide Web Consortium の Web サイト www.w3.org を参照してください。

Web 公開エンジンのアプリケーションログの使用

Web 公開エンジンでは、Web 公開エンジンのエラー、スクリプト、およびユーザログ情報の記録であるアプリケーションログファイルがデフォルトで生成されます。

- エラーログ情報には、Web 公開エンジンで発生した一般的ではないエラーが記述されます。Web ユーザに通知された一般的なエラー（たとえば、「データベースが開いていません」など）は記録されません。
- スクリプトログ情報には、Web ユーザがスクリプトを実行したときに生成されたエラーが記録されます。たとえば、スクリプトステップが Web 互換でない場合に、スキップされたスクリプトステップの一覧が記述されます。
- ユーザログメッセージには、XSLT スタイルシートの XSLT <xsl:message> エレメントによって生成されたメッセージが含まれます。Web ユーザが XSLT スタイルシートにアクセスすると、<xsl:message> エレメント内に含めた情報がアプリケーションログファイルに記録されます。第 6 章「FileMaker XSLT スタイルシートの開発」を参照してください。

アプリケーションログは「pe_application_log.txt」という名前で、Web 公開エンジンホスト上の「FileMaker Server」フォルダ内の「Logs」フォルダにあります。

「pe_application_log.txt」ファイルは、Web 公開エンジンのログオプションで次のいずれかが有効な場合に生成されます。

有効になっているログオプション	「pe_application_log.txt」で記録される情報
[エラーログ :]	Web 公開エンジンで発生した一般的ではないエラー。Web ユーザに通知された一般的なエラー（たとえば、「データベースが開いていません」など）は記録されません。
[スクリプトログ :]	Web ユーザがスクリプトを実行したときに生成されたエラー。たとえば、スクリプトステップが Web 互換でない場合に、スキップされたスクリプトステップの一覧が記述されます。
[ユーザログ :]	Web ユーザがカスタム Web 公開ソリューションにアクセスしたときに生成されるメッセージ。

ログオプションのこれらの 3 つのオプションは、すべてデフォルトで有効になっています。Admin Console を 使用してこれらのオプションを設定する場合の詳細については、「FileMaker Server ヘルプ」を参照してください。

メモ アプリケーションログ内の項目は自動的には削除されないため、時間の経過とともにファイルのサイズが非常に大きくなる場合があります。ホストコンピュータ上のハードディスクのスペースを節約するために、定期的にアプリケーションログファイルのアーカイブを作成してください。

Web サーバーモジュールのエラーログの使用

Web サーバーが Web 公開エンジンに接続できない場合は、Web サーバーモジュールによって、処理に関するエラーを記録するログファイルが生成されます。このファイルは「web_server_module_log.txt」という名前で、Web 公開エンジンホスト上の「FileMaker Server」フォルダ内の「Logs」フォルダにあります。

Web 公開コアの内部アクセスログの使用

Web 公開エンジンの Web 公開コアソフトウェアコンポーネントでは、Web 公開コアへの各アクセスの記録である 2 つの内部アクセスログファイルが生成されます。

- 「wpc_access_log.txt」アクセスログは、XML を生成したり、FileMaker Server インスタント Web 公開を使用するためのすべてのエンドユーザリクエストの記録です。これらのリクエストは、Web サーバーから Web 公開コアに直接ルーティングされます。
- 「pe_internal_access_log.txt」アクセスログは、XSLT リクエストの処理中に Web 公開エンジンの XSLT-CWP ソフトウェアコンピュータによって実行されるすべての内部 XML リクエストの記録です。これらのリクエストは、Web 公開エンジン内で、XSLT-CWP ソフトウェアコンポーネントから Web 公開コアソフトウェアコンポーネントに内部的にルーティングされます。

これらのファイルは、Web 公開エンジンホスト上の「FileMaker Server」フォルダ内の「Logs」フォルダにあります。

内部アクセスログは、Web 公開エンジンのアクセスログオプションが有効な場合に生成されます。デフォルトの設定は [有効] です。アクセスログオプションを設定する場合の詳細については、「FileMaker Server ヘルプ」を参照してください。

付録 A
クエリー文字列で使用される有効な名前

この付録では、Web 公開エンジンを使用して FileMaker データにアクセスする場合にクエリー文字列で使用できる、クエリーコマンドと引数の有効な名前を説明します。

クエリーコマンドと引数について

次に、すべてのクエリーコマンド名とクエリー引数名の一覧を示します。

クエリーコマンド名	クエリー引数名
-dbnames (81 ページを参照)	-db (83 ページを参照)
-delete (81 ページを参照)	-encoding (XSLT のみ) (84 ページを参照)
-dup (81 ページを参照)	-field (84 ページを参照)
-edit (81 ページを参照)	フィールド名（84 ページを参照）
-find、-findall、-findany (81 ページを参照)	フィールド名 .op（85 ページを参照）
-findquery (82 ページを参照)	-grammar (XSLT only) (85 ページを参照)
-layoutnames (82 ページを参照)	-lay (85 ページを参照)
-new (82 ページを参照)	-lay.response (86 ページを参照)
-process (XSLT のみ) (83 ページを参照)	-lop (86 ページを参照)
-scriptnames (83 ページを参照)	-max (86 ページを参照)
-view (83 ページを参照)	-modid (86 ページを参照)
	-query (87 ページを参照)
	-recid (87 ページを参照)
	-relatedsets.filter (88 ページを参照)
	-relatedsets.max (88 ページを参照)
	-script (88 ページを参照)
	-script.param (88 ページを参照)
	-script.prefind (89 ページを参照)
	-script.prefind.param (89 ページを参照)
	-script.presort (89 ページを参照)
	-script.presort.param (89 ページを参照)
	-skip (90 ページを参照)
	-sortfield.[1-9] (90 ページを参照)
	-sortorder.[1-9] (90 ページを参照)
	-stylehref (91 ページを参照)
	-styletype (91 ページを参照)
	-token.[string] (XSLT のみ) (91 ページを参照)

重要 -dbnames、-layoutnames、-scriptnames、および -process（XSLT リクエストのみ）以外のすべてのクエリーコマンドで、レイアウトを指定するための -lay クエリー引数が必須になっています。

クエリーコマンドと引数の使用のガイドライン

クエリー文字列でクエリーコマンドと引数を使用する場合は、次の点に注意してください。

- クエリー文字列に含めるクエリーコマンドは、1 つだけにする必要があります。クエリーコマンドをまったく指定しないことも、2 つ以上のクエリーコマンドを指定することもできません。たとえば、新しいレコードを追加するためにクエリー文字列に -new を含めることができますが、同じ文字列に -new と -edit を含めることはできません。
- ほとんどのクエリーコマンドでは、対応するさまざまなクエリー引数をクエリー文字列で指定する必要があります。たとえば、-dbnames および -process 以外のすべてのクエリーコマンドでは、クエリー対象のデータベースを指定する -db 引数が必要です。必要な引数については、43 ページの「FileMaker クエリー文字列を使用した XML データの要求」の表を参照してください。

- クエリー引数とフィールド名には、-db=employees など、使用する特定の値を指定します。クエリーコマンドには、-findall などのコマンド名の後に「=」記号や値を指定しないでください。
- クエリーコマンドと引数の名前は、-delete や -lay のように小文字で指定する必要があります。
- クエリー文字列で使用されるデータベース名、レイアウト名、およびフィールド名では、大文字と小文字は区別されません。たとえば、MyLayout を指定するために -lay=mylayout を使用できます。

  メモ クエリー文字列の外部の XSLT ステートメントで使用されるフィールドとデータベースの名前では大文字と小文字が区別され、データベースで使用されている実際の名前に完全に一致する必要があります。たとえば、次のステートメントがあるとします。

  ```
  <xsl:value-of select="fmrs:field[@name='LastName']"/>
  ```

  この場合、フィールド参照「LastName」は、データベースの「LastName」フィールドの名前に完全に一致する必要があります。

- フィールド名にはピリオドを入れることができますが、以下の例外があります。
  - ピリオドは、数字の前に置くことはできません。たとえば、「myfield.9」のフィールド名は無効です。
  - ピリオドは、文字列「op」（２文字の「op」）の前に置くことはできません。たとえば、「myfield.op」のフィールド名は無効です。
  - ピリオドは、文字列「global」（「global」という文字）の前に置くことはできません。たとえば、「myfield.global」のフィールド名は無効です。

  これらの例外のいずれかが含まれるフィールド名（「text.field」など）に、HTTP クエリーを使用して XML または XSLT でアクセスすることはできません。これらの構造は、以下の「完全修飾フィールド名の構文について」で記述されているとおり、レコード ID に予約されています。
- -find コマンドでは、フィールドの値の大文字と小文字は区別されません。たとえば、Field1=Blue または Field1=blue を使用することができます。-new および -edit コマンドでは、フィールドの値に使用した大文字と小文字は保持され、クエリー文字列で指定したとおりにデータベースに保存されます。たとえば、LastName=Doe などの大文字と小文字は保持されます。

FileMaker クエリー文字列リファレンスについて

このリリースには、FileMaker の各クエリーコマンドとクエリー引数の簡単な説明と例が含まれた「クエリー文字列リファレンス .fp7」という FileMaker データベースが収録されています。これは、FileMaker Server 展開にある任意のコンピュータ（マスタまたはワーカー）の以下のディレクトリにあります。

Mac:

/ ライブラリ /FileMaker Server/Examples/XSLT

Windows:

< ドライブ >:¥ Program Files ¥ FileMaker ¥ FileMaker Server ¥ Examples ¥ XSLT

場所 : < ドライブ > は、システムを起動する第 1 ドライブです。

完全修飾フィールド名の構文について

完全修飾フィールド名により、フィールドのインスタンスが正確に識別されます。一般的な名前を使用したフィールドは別のテーブルに基づく可能性もあるため、場合によっては、エラーを回避するために完全修飾名を使用する必要があります。

次に、完全修飾フィールド名を指定するための構文を示します。

```
table-name::field-name(repetition-number).record-id
```

各要素の意味は、次のとおりです。

- table-name には、フィールドが含まれるテーブルの名前を指定します。テーブル名は、フィールドが、クエリー文字列で指定されているレイアウトの基本テーブルにない場合にのみ必要です。

- field-name(repetition-number) には、繰り返しフィールドの特定の値を指定します。これは、繰り返しフィールドに対してのみ必要です。繰り返し数は数字の 1 から始まります。たとえば、フィールド名 (2) は、繰り返しフィールドの 2 番目の値を参照します。繰り返しフィールドに対して繰り返し数を指定しなかった場合は、繰り返しフィールドの最初の値が使用されます。繰り返し数は、繰り返しフィールドが含まれる -new および -edit クエリーコマンドでは必要ですが、-find コマンドでは必要ありません。
- record-id には、レコード ID を指定します。クエリー文字列を使用して、ポータルフィールドにレコードを追加したり、ポータルフィールド内のレコードを編集する場合にのみ必要です。次のセクション「ポータルへのレコードの追加」および「ポータル内のレコードの編集」を参照してください。record-id は、ポータルフィールドが含まれる -new および -edit クエリーコマンドでは必要ですが、-find コマンドでは必要ありません。

メモ フィールドにアクセスできるようにするには、フィールドが、クエリー文字列で指定するレイアウト上に配置されている必要があります。

ポータルフィールドでのクエリーコマンドの使用

以下の各セクションでは、ポータルフィールドでのクエリーコマンドの操作方法について説明します。

ポータルへのレコードの追加

親レコードを追加すると同時にポータルに新しいレコードを追加するには、-new クエリーコマンドを使用して、リクエストのクエリー文字列で次のように指定します。

- 関連するポータルフィールドに対して完全修飾フィールド名を使用する
- 関連するポータルフィールドの名前の後に、レコード ID として 0 を指定する
- 関連するポータルフィールドを指定する前に、親レコードの少なくとも 1 つのフィールドを指定する
- 親レコードの照合フィールド（キーフィールド）のデータを指定する

たとえば、次の URL では、「Employees」に John Doe の新しい親レコードを追加すると同時に、ポータルに Jane の新しい関連レコードを追加します。関連テーブルの名前は「Dependents」で、ポータル内の関連フィールドの名前は「Names」です。照合フィールド「ID」には、従業員の ID 番号が保存されています。

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&FirstName=John&LastName=Doe
&ID=9756&Dependents::Names.0=Jane&-new

メモ 1 つのリクエストでポータルに追加できる関連レコードは 1 つだけです。

ポータル内のレコードの編集

ポータル内の 1 つまたは複数のレコードを編集するには、-edit コマンドとレコード ID を使用して、編集するポータルレコードが含まれる親レコードを指定します。そのレコード ID を完全修飾フィールド名で使用して、編集する特定のポータルレコードを指定します。レコード ID は、XML データの <relatedset> エレメント内にある <record> エレメントの record-id 属性から判断できます。36 ページの「fmsresultset 文法の使用」を参照してください。

たとえば、次の URL では、親レコードのレコード ID が 1001 であるポータル内のレコードを編集します。「Dependents」は関連テーブルの名前、「Names」はポータル内の関連フィールドの名前、「Names.2」の「2」はポータルレコードのレコード ID です。

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&-recid=1001
&Dependents::Names.2=Kevin&-edit

次に、1 つのリクエストを使用して、親レコード経由で複数のポータルレコードを編集する方法の例を示します。

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&-recid=1001
&Dependents::Names.2=Kevin&Dependents::Names.5=Susan&-edit

-edit コマンドを使用してポータルのレコード ID として 0 を指定し、既存の親レコードに対してポータル内で新しい関連レコードを追加することもできます。例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&-recid=1001
&Dependents::Names.0=Timothy&-ledit

ポータルレコードの削除

ポータルレコードを削除するには、-delete コマンドではなく -edit コマンドで -delete.related 引数を使用します。

たとえば、次の URL では、employees から「1001」レコードが削除されます。

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&-recid=1001&-delete

ただし、次の URL では、「Dependents」という関連テーブルから、親レコード「1001」とともに、レコード ID が「3」であるポータルレコードが削除されます。

http://192.168.123.101/fmi/xml/fmresultset.xml?ñdb=employees&ñlay=family&ñrecid=1001
&ñdelete.related=Dependents.3&ñedit

83 ページの「-delete.related （ポータルレコードを削除）クエリー引数」を参照してください。

ポータルレコードのクエリーを実行

関連レコードが多くあるソリューションでは、ポータルレコードのクエリーを実行してソートすると、時間がかかる可能性があります。関連セットで表示するレコードと行の数を制限するには、-relatedsets.filter 引数および -relatedsets.max 引数を使用してリクエストを検索します。詳細については、88 ページの「-relatedsets.filter （ポータルレコードのフィルタ）クエリー引数」および 88 ページの「-relatedsets.max （ポータルレコードの制限）クエリー引数」を参照してください。

グローバルフィールドを指定するための構文について

次に、グローバルフィールドを指定するための構文を示します。

テーブル名 :: フィールド名 (繰り返し数).global

global により、フィールドは、グローバル格納を使用するものと識別されます。テーブル名およびフィールド名（繰り返し数）の詳細については、78 ページの「完全修飾フィールド名の構文について」を参照してください。グローバルフィールドの詳細については、FileMaker Pro ヘルプを参照してください。

クエリー文字列内でグローバルフィールドを識別するには、.global の構文を使用する必要があります。Web 公開エンジンでは、グローバルフィールドの引数値は、クエリーコマンドを実行する前、またはクエリー文字列内の他の引数値を設定する前に設定されます。直接の XML リクエスト、およびセッションを使用しない XSLT スタイルシートによって実行されるリクエストでは、グローバル値は、リクエストの実行後ただちに使用期限が切れます。セッションを使用する XSLT スタイルシートによって実行されるリクエストでは、スタイルシートで定義されたセッションの期間中、または他のリクエストで変更されるまで、グローバル値は維持されます。

.global 構文を使用してクエリー文字列でグローバルフィールドを識別しない場合、Web 公開エンジンは、最初にグローバルフィールドの値を設定せずに、クエリー文字列の残りの部分を使用してグローバルフィールドを評価します。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments
&Country.global=USA&-recid=1&-edit

重要 XSLT スタイルシートでグローバルフィールドを使用する場合は、Admin Console を使用して、Web 公開エンジンの XSLT データベースセッションオプションを有効にする必要があります。このオプションが有効でない場合、グローバルフィールドの値は、リクエスト間で維持されません。「FileMaker Server ヘルプ」を参照してください。

クエリーコマンドリファレンス

このセクションでは、XML および XSLT のリクエストで使用可能なクエリーコマンドについて説明します。

メモ XSLT リクエストの場合のみ、次のすべてのクエリーコマンドで -grammar クエリー引数が必要です。

-dbnames （データベース名）クエリーコマンド

FileMaker Server でホストされていて、カスタム Web 公開 with XML または XSLT が有効なすべてのデータベースの名前を取得します。

必須のクエリー引数：（なし）

例：

データベース名を取得する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-dbnames

-delete （レコード削除）クエリーコマンド

-recid 引数で指定されているレコードを削除します。

必須のクエリー引数：-db、-lay、-recid

オプションのクエリー引数：-script

例：

レコードを削除する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-recid=4&-delete

-dup （レコード複製）クエリーコマンド

-recid で指定されているレコードを複製します。

必須のクエリー引数：-db、-lay、-recid

オプションのクエリー引数：-script

例：

指定したレコードを複製する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-recid=14&-dup

-edit （レコード編集）クエリーコマンド

任意のフィールド名 / 値の組の内容をフィールドに入れて、-recid 引数で指定されているレコードを更新します。-recid 引数は編集されるレコードを示します。

必須のクエリー引数：-db、-lay、-recid、 1 つまたは複数のフィールド名

オプションのクエリー引数：-modid、-script

メモ ポータル内のレコードの編集の詳細については、79 ページの「ポータル内のレコードの編集」を参照してください。

例：

レコードを編集する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-recid=13&Country=USA&-edit

-find、-findall、または -findany （レコードの検索）クエリーコマンド

定義された条件を使用して検索リクエストを送信します。

必須のクエリー引数：-db、-lay

オプションのクエリー引数：-recid、-lop、-op、-max、-skip、-sortorder、-sortfield、-script、-script.prefind、-script.presort、フィールド名

例：

レコードをフィールド名で検索する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&Country=USA&-find

メモ 1 回のリクエストでのフィールド名の複数回指定はサポートされていません。FileMaker Server では、すべての値を解析しますが、解析された最後の値のみが使用されます。

レコードをレコード ID で検索する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&-recid=427&-find

データベース内のすべてのレコードを検索する場合には、-findall を使用します：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&-findall

ランダムなレコードを検索する場合には、-findany を使用します：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=family&-findany

-findquery （複合検索）クエリーコマンド

複数の検索レコードおよびレコード除外リクエストを使用して、検索リクエストを送信します。

必須のクエリー引数：-db、-lay、-query

オプションのクエリー引数：-max、-skip、-sortorder、-sortfield、-script、-script.prefind、-script.presort

例：

「Fluffy」という名前でない猫または犬のレコードの検索

http://host/fmi/xml/fmresultset.xml?-db=vetclinic&-lay=animals&-query=(q1);(q2);!(q3)&-q1=typeofanimal&-q1.value=Cat&-q2=typeofanimal&-q2.value=Dog&-q3=name&-q3.value=Fluffy&-findquery

複合検索での -findquery コマンドの使用

-findquery ステートメントは、以下の順序での 4 つのパートからなります。

- -query 引数
- クエリー識別子宣言とリクエスト処理からなるクエリーリクエスト宣言
- 各識別子における検索フィールドと値の定義
- 構文全体の最後にある -findquery コマンド

-query 引数の使用法の詳細については、87 ページの「-query （複合検索条件）クエリー引数」を参照してください。

-layoutnames （レイアウト名）クエリーコマンド

FileMaker Server でホストされていて、カスタム Web 公開 with XML または XSLT が有効な指定されたデータベースで使用可能なレイアウトすべての名前を取得します。

必須のクエリー引数：-db

例：

使用可能なレイアウトの名前を取得する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-layoutnames

-new （新規レコード）クエリーコマンド

新規レコードを作成し、そのレコードに任意のフィールド名 / 値の組の内容を入れます。

必須のクエリー引数：-db、-lay

オプションのクエリー引数： 1 つまたは複数のフィールド名、-script

メモ ポータルに新しいレコードを含める場合の詳細については、79 ページの「ポータルへのレコードの追加」を参照してください。

例：

新規レコードを追加する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&Country=Australia&-new

-process （XSLT スタイルシートの処理）

データベースのデータベースを要求せずに、XSLT スタイルシートを処理します。このクエリーコマンドは、XSLT スタイルシートとともにのみ使用できます。

必須のクエリー引数：-grammar。fmresultset または FMPXMLRESULT 文法を使用する必要があります。

例：

http://192.168.123.101/fmi/xsl/my_template/my_stylesheet.xsl?-grammar=fmresultset&-process

53 ページの「FileMaker Server に対してクエリーを実行しない XSLT リクエストの処理」を参照してください。

-scriptnames （スクリプト名）クエリーコマンド

FileMaker Server でホストされていて、カスタム Web 公開 with XML または XSLT が有効な指定されたデータベースで使用可能なスクリプトすべての名前を取得します。

必須のクエリー引数：-db

例：

すべてのスクリプトの名前を取得する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-scriptnames

-view （レイアウト情報の表示）クエリーコマンド

FMPXMLLAYOUT 文法が指定されている場合は、データベースからレイアウト情報を取得して、FMPXMLLAYOUT 文法で表示します。データ文法（fmresultset または FMPXMLRESULT）が指定されている場合は、XML ドキュメントの metadata セクションおよび空のレコードセットを取得します。

必須のクエリー引数：-db、-lay

オプションのクエリー引数：-script

例：

レイアウト情報を取得する場合：

http://192.168.123.101/fmi/xml/FMPXMLLAYOUT.xml?-db=employees&-lay=departments&-view

メタデータ情報を取得する場合：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-view

クエリー引数リファレンス

このセクションでは、XML および XSLT リクエストで使用可能なクエリー引数について説明します。XSLT リクエストでのみ使用可能な引数の詳細については、50 ページの「FileMaker XSLT スタイルシートでのクエリー文字列の使用」を参照してください。

-db （データベース名）クエリー引数

クエリーコマンドを適用するデータベースを指定します。

値：データベースの名前（ファイル拡張子がある場合は、拡張子を含まない名前）

メモ クエリー引数で -db 引数にデータベースの名前を指定する場合は、ファイル拡張子を含めないでください。実際のデータベースファイル名にはオプションで拡張子を含めることができますが、-db 引数の値として拡張子は使用できません。

必須：-dbnames および -process 以外のすべてのクエリーコマンド

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-findall

-delete.related （ポータルレコードを削除）クエリー引数

ポータルフィールドからレコードを削除します。

オプション：-edit クエリーコマンド

必須：関連テーブル名とレコード ID

例：

次の例では、「jobtable」という関連テーブルから、親レコード「7」とともに、レコード ID が「20」であるポータルレコードが削除されます。

http://host/fmi/xml/fmresultset.xml?-db=career&-lay=applications&-recid=7&-delete.related=jobtable.20&-edit

-encoding （XSLT リクエストのエンコード）クエリー引数

XSLT リクエストのテキストエンコードを指定します。このクエリーコマンドは、カスタム Web 公開 with XSLT のリクエストでのみ使用できます。

値：US-ASCII、ISO-8859-1、ISO-8859-15、ISO-2022-JP、Shift_JIS、または UTF-8

オプション：XSLT リクエストのすべてのクエリーコマンド

例：

http://192.168.123.101/fmi/xsl/my_template/my_stylesheet.xsl?-db=employees&-lay=departments
&-grammar=fmresultset&-encoding=Shift_JIS&-findall

52 ページの「リクエストのテキストエンコードの設定」を参照してください。

-field （オブジェクトフィールド名）クエリー引数

オブジェクトフィールドの名前を指定します。

必須：オブジェクトフィールドのデータに対するリクエスト

34 ページの「XML ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について」および 49 ページの「XSLT ソリューション内の FileMaker オブジェクトにアクセスするための URL 構文について」を参照してください。

フィールド名（オブジェクトフィールド以外のフィールド名）クエリー引数

フィールド名は、-find クエリーコマンドの条件の制御とレコードの内容の変更に使用されます。クエリーコマンドや引数にオブジェクトフィールド以外のフィールドの値を指定する必要がある場合は、名前 / 値の組の名前の部分としてハイフン（-）文字を付けずにフィールド名を使用します。

名前：FileMaker データベース内のフィールドの名前。フィールドが、クエリー文字列で指定されたレイアウトの基本テーブルにない場合、フィールド名は完全修飾されている必要があります。フィールド名にはピリオドを入れることができますが、以下の例外があります。

- ピリオドは、数字の前に置くことはできません。たとえば、「myfield.9」のフィールド名は無効です。
- ピリオドは、文字列「op」（2 文字の「op」）の前に置くことはできません。たとえば、「myfield.op」のフィールド名は無効です。
- ピリオドは、文字列「global」（「global」という文字）の前に置くことはできません。たとえば、「myfield.global」のフィールド名は無効です。

これらの例外のいずれかが含まれるフィールド名（「text.field」など）に、HTTP クエリを使用して XML または XSLT でアクセスすることはできません。これらの構造は、78 ページの「完全修飾フィールド名の構文について」で記述されているとおり、レコード ID に予約されています。

値：-new および -edit クエリーコマンドでは、現在のレコード内のフィールドに保存する値を指定します。-find クエリーコマンドでは、フィールドで検索する値を指定します。日付、時刻、およびタイムスタンプのフィールドの値を指定する場合、そのフィールドタイプに「fm」の書式を使用して、値を指定する必要があります。「fm」形式では、日付は MM/dd/yyyy、時刻は HH:mm:ss、タイムスタンプは MM/dd/yyyy HH:mm:ss です。

必須：-edit クエリーコマンド

オプション：-new および -find クエリーコマンド

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-op=eq&FirstName=Sam
&-max=1&-find

メモ　1 回のリクエストでのフィールド名の複数回指定はサポートされていません。FileMaker Server では、すべての値を解析しますが、解析された最後の値のみが使用されます。

フィールド名 .op（比較演算子）クエリー引数

演算子の前に指定したフィールド名に適用する比較演算子を指定します。比較演算子は、-find クエリーコマンドとともに使用します。

値：使用する演算子。デフォルトの演算子は「begins with」です。次に、有効な演算子を示します。

キーワード	FileMaker Pro の演算子
eq	= 値
cn	* 値 *
bw	値 *
ew	* 値
gt	> 値
gte	>= 値
lt	< 値
lte	<= 値
neq	除外 , 値

オプション：-find クエリーコマンド

必須：フィールド名と値

次に、比較演算子を指定するための構文を示します。

テーブル名 :: フィールド名 = 値 & テーブル名 :: フィールド名 .op= 演算子記号

各要素の意味は、次のとおりです。

- テーブル名には、フィールドが含まれるテーブルを指定します。フィールドが、クエリー文字列で指定されているレイアウトの基本テーブルにない場合にのみ必要です。
- 演算子記号には、cn など、前の表に示されているキーワードの 1 つを指定します。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&name=Tim&name.op=cn&-find

bw キーワードを指定して、FileMaker Pro の任意の検索演算子を使用できます。たとえば、範囲の演算子（...）を使用して値の範囲を検索するには、bw キーワードを指定して、検索条件の前に「...」の文字を配置します。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&IDnum=915...925&IDnum.op=bw
&-find

テキストの検索に使用できる演算子の詳細については、「FileMaker Pro ヘルプ」を参照してください。

-grammar （XSLT スタイルシートの文法）クエリー引数

XSLT スタイルシートに対して使用する文法を指定します。このクエリーコマンドは、カスタム Web 公開 with XSLT のリクエストでのみ使用できます。

値：fmresultset、FMPXMLRESULT、または FMPXMLLAYOUT

必須 : すべての XSLT リクエスト

例：

http://192.168.123.101/fmi/xsl/my_template/my_stylesheet.xsl?-grammar=fmresultset&-db=mydatabase
&-lay=mylayout&-findall

50 ページの「FileMaker XSLT スタイルシートの XML 文法の指定」を参照してください。

-lay （レイアウト）クエリー引数

使用するデータベースのレイアウトを指定します。

値：レイアウトの名前

必須：-dbnames 以外のすべてのクエリーコマンド -layoutnames、-scriptnames および -process (XSLT リクエストのみ)。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-view

-lay.response （応答のレイアウトの切り替え）クエリー引数

リクエストを処理する際には -lay 引数で指定されているレイアウトを使用し、XML 応答を処理する際には -lay.response 引数で指定されているレイアウトに切り替えるよう指定します。

-lay.response 引数が含まれていない場合は、リクエストの処理時も、応答の処理時も、-lay 引数で指定されているレイアウトが使用されます。

-lay.response 引数は、XML リクエストに対して、または XSLT スタイルシートリクエスト内のいずれかで使用できます。

値：レイアウトの名前

オプション：-dbnames、-layoutnames、-scriptnames、および -process（XSLT リクエストのみ）以外のすべてのクエリーコマンドで必須です。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=Budget&Salary=100000&Salary.op=gt&-find
-lay.response=ExecList

-lop （論理演算子）クエリー引数

-find クエリーコマンドに含まれる複数の検索条件を「and」または「or」のいずれの検索として組み合わせるかを指定します。

値：「and」または「or」（小文字で指定する必要があります）。-lop クエリー引数が含まれない場合、-find クエリーコマンドでは「and」の値が使用されます。

オプション：-find クエリーコマンド

メモ -findquery クエリーコマンドによるサポートはありません。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&Last+Name=Smith
&Birthdate=2/5/1972&-lop=and&-find

-max （最大レコード）クエリー引数

返されるレコードの最大数を指定します。

値：数字。すべてのレコードを返すには、値 all を使用します。値「all」は、小文字で指定する必要があります。-max が指定されていない場合は、すべてのレコードが返されます。

オプション：-find、-findall、および -findquery クエリーコマンド

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-max=10&-findall
http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-max=all&-findall

-modid （修正 ID）クエリー引数

修正 ID は、レコードの現在のバージョンを指定する増加するカウンタです。-edit クエリーコマンドを使用する際に修正 ID を指定することで、確実にレコードの現在のバージョンを編集できます。指定した修正 ID の値がデータベースの現在の修正 ID の値に一致しない場合、-edit クエリーコマンドは使用できず、エラーコードが返されます。

値：修正 ID。修正 ID は、FileMaker データベースのレコードの現在のバージョンを指定する固有の ID です。

オプション：-edit クエリーコマンド

必須：-recid 引数

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-recid=22&-modid=6
&last_name=Jones&-edit

-query （複合検索条件）クエリー引数

複合検索条件における、クエリー名と検索基準を指定します。82 ページの「-findquery （複合検索）クエリーコマンド」を参照してください。

値：クエリー式。

必須：-findquery クエリーコマンド

複合検索条件用の構文は、次のようになります。

-query=< リクエスト宣言 >< リクエスト定義 >&-findquery

各要素の意味は、次のとおりです。

< リクエスト宣言 > は、２つ以上のリクエスト宣言です。

- 各リクエスト宣言は、コンマで区切られた 1 つまたは複数のクエリー識別子からなり、カッコで囲まれます。クエリー識別子では、「q」という文字の後ろに数字が付きます。例：q1
- カッコで囲まれ、複数のクエリーは、対象レコードを絞り込む論理式 AND での検索として動作します。たとえば、(q1, q2) で返されるレコードは、q1 および q2 と一致します。
- FileMaker Pro の場合のように、各リクエストは検索リクエストまたは除外リクエストのいずれかにできます。検索リクエストを行うと、一致するレコードが対象レコードに追加されます。デフォルトは、検索リクエストです。除外リクエストの場合、リクエストの前に感嘆符（!）が付きます。

  例：(q1);!(q2)

  この例では、q1 が検索リクエストで、感嘆符が前に付いている q2 は除外リクエストです。
- リクエストは、セミコロンで区切られます。複数のリクエストは、対象レコードを拡大する論理式 OR での検索として動作します。たとえば、(q1);(q2) で返されるレコードは、q1 or q2 と一致します。除外リクエストは、検索条件からレコードを削除するため論理式 OR 検索としては動作しません。
- リクエストは、指定された順序で実行されます。対象レコードには、複合検索条件全体の結果が含まれます。

< リクエスト定義 > は、各リクエスト宣言におけるリクエスト定義です。各リクエスト定義は、検索フィールドと値の定義からなります。マイナス（-）記号は、リクエスト定義の開始です。

構文：

-< クエリー ID>=< フィールド名 > & -< クエリー ID>.value=< 値 >

例：

-q1=typeofanimal&-q1.value=Cat

-q2=name&-q2.value=Fluffy

例：

「Fluffy」という名前でない「Gray」の猫の検索。

http://host/fmi/xml/fmresultset.xml?-db=petclinic&-lay=Patients&-query=(q1, q2);!(q3)
&-q1=typeofanimal&-q1.value=Cat&-q2=color&-q2.value=Gray&-q3=name&-q3.value=Fluffy&-findquery

-recid （レコード ID）クエリー引数

処理するレコードを指定します。主に -edit および -delete クエリーコマンドで使用されます。
-view コマンドによって使用され、関連する値一覧データを FMPXMLLAYOUT 文法で取得します。

値：レコード ID。レコード ID は、FileMaker データベースのレコードの固有の識別子です。

必須：-edit、-delete、および -dup クエリーコマンド

オプション：-find クエリー、および -view コマンド

例 1 ：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-recid=22&-delete

例 2 ：

http://localhost/fmi/xml/FMPXMLLAYOUT.xml?-db=test&-lay=empty&-view&-recid=9

-relatedsets.filter （ポータルレコードのフィルタ）クエリー引数

ポータルフィールドを使用するクエリーで返される行をフィルタするかどうかを指定します。

値：layout または none

- この引数が指定されていない場合、デフォルトの値は「none」です。
- クエリーで「layout」が指定されている場合、FileMaker Pro の [ポータル設定] ダイアログボックスで指定された設定が優先されます。レコードは、[ポータル設定] ダイアログボックスで定義されたソートに基づいてソートされ、レコード セットは、指定された最初の行から開始するようにフィルタされます。
- [ポータル設定] ダイアログボックスで [垂直スクロールバーを表示] 設定が有効になっている場合、-relatedsets.max オプションを使用すると、クエリーに応答して返すように最大数の行を指定できます。
- [垂直スクロールバーを表示] 設定が無効になっている場合、FileMaker Pro の [ポータル設定] ダイアログボックスの [行数] 設定によって、表示する行数が決定されます。
- -relatedsets.filter が「none」に設定されている場合、Web 公開エンジンによって、ポータル内のすべての行、および事前にソートされていないポータルレコードが返されます。

オプション：-find、-edit、-new、-dup、および -findquery

例：

http://localhost/fmi/xml/fmresultset.xml?-db=FMPHP_Sample&-lay=English&-relatedsets.filter=none&-findany

http://localhost/fmi/xml/fmresultset.xml?-db=FMPHP_Sample&-lay=English&relatedsets.filter=layout&-relatedsets.max=all&-findany

http://localhost/fmi/xml/fmresultset.xml?-db=FMPHP_Sample&-lay=English&-relatedsets.filter=layout&-relatedsets.max=10&-findany

-relatedsets.max （ポータルレコードの制限）クエリー引数

このクエリーの結果で返す最大数の行を指定します。

値：整数、または all

- クエリーで整数が指定される場合、Web 公開エンジンによって、最初の行を返した後の行数が返されます。
- クエリーで all が指定される場合、Web 公開エンジンによって、すべての関連レコードが返されます。
- クエリーで -relatedsets.max 引数が指定されない場合、-relatedsets.filter 引数で指定された値によって、行数が決定されます。88 ページの「-relatedsets.filter （ポータルレコードのフィルタ）クエリー引数」を参照してください。

オプション：-find、-edit、-new、-dup、および -findquery°

例：

http://localhost/fmi/xml/fmresultset.xml?-db=FMPHP_Sample&-lay=English&relatedsets.filter=layout&-relatedsets.max=all&-findany

http://localhost/fmi/xml/fmresultset.xml?-db=FMPHP_Sample&-lay=English&-relatedsets.filter=layout&-relatedsets.max=10&-findany

-script （スクリプト）クエリー引数

クエリーコマンドとソートの実行後に実行する FileMaker スクリプトを指定します。45 ページの「XML リクエストの処理方法の理解」を参照してください。

値：スクリプト名

オプション：-dbnames、-layoutnames、-process、および -scriptnames

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-script=myscript&-findall

-script.param（スクリプトに引数を渡す）クエリー引数

-script によって指定された FileMaker スクリプトに、引数を渡します。

値： 1 つのテキスト引数。

- 複数の引数内に渡すには、それらの引数を区切る文字列を作成し、スクリプトが個々の引数を解析するようにします。たとえば、「|」文字が「param1%7Cparam2%7Cparam3」のように URL エンコードされたリストとして、「param1|param2|param3」を渡します。

- テキストでない値をテキスト引数として処理するには、スクリプトをテキスト値に変換します。たとえば、テキスト値を数字に変換する場合には、スクリプトに次を含めることができます。
  GetAsNumber(Get(スクリプト引数))
- クエリーに -script がなく -script.param が含まれている場合、-script.param は無視されます。
- クエリーに複数の -script.param が含まれている場合、Web 公開エンジンによって、解析される最後の値が使用されます。

オプション：-script

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-script=myscript
&-script.param=Smith%7CChatterjee%7CSu&-findall

-script.prefind （検索前のスクリプト）クエリー引数

-find クエリーコマンドの処理時に、レコードの検索とソート（指定されている場合）の前に実行する FileMaker スクリプトを指定します。

値：スクリプト名

オプション：-dbnames、-layoutnames、-process、および -scriptnames

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-script.prefind=myscript&-findall

-script.prefind.param （検索前にスクリプトに引数を渡す）クエリー引数

-script.prefind によって指定された FileMaker スクリプトに、引数を渡します。

値： 1 つのテキスト引数。

- 複数の引数内に渡すには、それらの引数を区切る文字列を作成し、スクリプトが個々の引数を解析するようにします。たとえば、「|」文字が「param1%7Cparam2%7Cparam3」のように URL エンコードされたリストとして、「param1|param2|param3」を渡します。
- テキストでない値をテキスト引数として処理するには、スクリプトをテキスト値に変換します。たとえば、テキスト値を数字に変換する場合には、スクリプトに次を含めることができます。
  GetAsNumber(Get(スクリプト引数))
- クエリーに -script.prefind がなく -script.prefind.param が含まれている場合、-script.prefind.param は無視されます。
- クエリーに複数の -script.prefind.param が含まれている場合、Web 公開エンジンによって、解析される最後の値が使用されます。

オプション：-script.prefind

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-script.prefind=myscript
&-script.prefind.param=payroll&-findall

-script.presort （ソート前のスクリプト）クエリー引数

-find クエリーコマンドの処理時に、レコードの検索（指定されている場合）の後、レコードの検索の前に実行する FileMaker スクリプトを指定します。

オプション：-dbnames、-layoutnames、-process、および -scriptnames

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-script.presort=myscript
&-sortfield.1=dept&-sortfield.2=rating&-findall

-script.presort.param （ソート前にスクリプトに引数を渡す）クエリー引数

-script.presort によって指定された FileMaker スクリプトに、引数を渡します。

値： 1 つのテキスト引数。

- 複数の引数内に渡すには、それらの引数を区切る文字列を作成し、スクリプトが個々の引数を解析するようにします。たとえば、「|」文字が「param1%7Cparam2%7Cparam3」のように URL エンコードされたリストとして、「param1|param2|param3」を渡します。
- テキストでない値をテキスト引数として処理するには、スクリプトをテキスト値に変換します。たとえば、テキスト値を数字に変換する場合には、スクリプトに次を含めることができます。
  GetAsNumber(Get(スクリプト引数))
- クエリーに -script.presort がなく -script.presort.param が含まれている場合、-script.presort.param は無視されます。
- クエリーに複数の -script.presort.param が含まれている場合、Web 公開エンジンによって、解析される最後の値が使用されます。

オプション：-script.presort

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-script.presort=myscript&-script.presort.param=18%7C65&-sortfield.1=dept&-sortfield.2=rating&-findall

-skip （レコードのスキップ）クエリー引数

対象レコード内のスキップするレコードの数を指定します。

値：数字。値が対象レコード内のレコード数より大きい場合、レコードは表示されません。デフォルト値は 0 です。

オプション：-find クエリーコマンド

次の例では、結果の最初の 10 レコードがスキップされ、11 番目から 15 番目のレコードが返されます。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-skip=10&-max=5&-findall

-sortfield （ソートフィールド）クエリー引数

ソートに使用するフィールドを指定します。

値：フィールド名

オプション：-find または -findall クエリーコマンド

-sortfield クエリー引数を使用して、複数のフィールドソートを複数回実行できます。次に、ソートフィールドの優先順位を指定するための構文を示します。

-sortfield. 優先順位番号 = 完全修飾フィールド名

-sortfield. 優先順位番号クエリー引数の優先順位番号には、複数のソートフィールドを使用する場合の優先順位を指定する、 1 から始まる数字を指定します。

次の例では、最初に「dept」フィールドでソートされ、続いて「rating」フィールドでソートされます。-sortorder クエリー引数が指定されていないため、両方のフィールドは昇順でソートされます。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=performance&-sortfield.1=dept&-sortfield.2=rating&-findall

-sortorder （ソート順）クエリー引数

ソート順を指定します。

値：ソート順。次に、有効なソート順を示します。< 値一覧名 > には、「Custom」などの値一覧名を指定します。

キーワード	FileMaker Pro の演算子
昇順	a から z、-10 から 10 の昇順のソート
降順	z から a、10 から -10 の降順のソート
< 値一覧名 >	レイアウト上のフィールドに割り当てられた、指定した値一覧を使用したソート

オプション：-find または -findall クエリーコマンド

必須：-sortfield クエリー引数

-sortorder クエリー引数を -sortfield クエリー引数とともに使用して、複数のソートフィールドのソート順を指定できます。次に、ソートフィールドのソート順を指定するための構文を示します。

-sortorder. 優先順位番号 = ソート方法

各要素の意味は、次のとおりです。

- -sortorder. 優先順位番号引数の優先順位番号には、-sortorder クエリー引数の適用先の -sortfield クエリー引数を指定する 1 から 9 の数字を指定します。
- ソート方法には、ascend など、前の表に示されているソート順を指定するためのキーワードの 1 つを指定します。

次の例では、最も優先順位の高いソートフィールド（「dept」）のソート順は ascend で、2 番目に優先順位の高いソートフィールド（「rating」）のソート順は descend になっています。-sortorder.2 の優先順位番号 2 により、クエリー引数 -sortorder.2=descend が -sortfield.2=rating クエリー引数に適用されるように指定されています。

例：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=performance&-sortfield.1=dept
&-sortorder.1=ascend&-sortfield.2=rating&-sortorder.2=descend&-findall

メモ ソートフィールドに対して -sortorder クエリー引数が指定されていない場合は、デフォルトの昇順ソートが使用されます。

-stylehref （スタイル href）クエリー引数

XML ドキュメントでクライアントサイドの CSS（Cascading Style Sheet）または XSLT スタイルシートを使用できるように、出力ドキュメント内に XML スタイルシート処理命令を生成して、href 属性の値（href=/mystylesheet.css または href=/stylesheets/mystylesheet.xsl）を設定します。-stylehref 引数の値には、絶対パスを使用する必要があります。スタイルシートの名前には任意の名前を指定できますが、.css または .xsl のいずれかの拡張子を含める必要があります。46 ページの「サーバーサイドおよびクライアントサイドでのスタイルシートの処理の使用」を参照してください。この引数は -styletype 引数とともに使用します。

オプション：すべてのクエリーコマンド

必須：-styletype 引数

例（「mystylesheet.xsl」が Web サーバーソフトウェアのルートフォルダにあると想定しています）：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-styletype=text/xsl
&-stylehref=/mystylesheet.xsl&-findall

-styletype （スタイルタイプ）クエリー引数

XML ドキュメントでクライアントサイドの CSS（Cascading Style Sheet）または XSLT スタイルシートを使用できるように、出力ドキュメント内に XML スタイルシート処理命令を生成して、type 属性の値（type=text/css または type=text/xsl）を設定します。46 ページの「サーバーサイドおよびクライアントサイドでのスタイルシートの処理の使用」を参照してください。この引数は -stylehref 引数とともに使用します。

オプション：すべてのクエリーコマンド

必須：-stylehref 引数

例（「mystylesheet.css」が Web サーバーソフトウェアのルートフォルダにあると想定しています）：

http://192.168.123.101/fmi/xml/fmresultset.xml?-db=employees&-lay=departments&-styletype=text/css
&-stylehref=/mystylesheet.css&-findall

-token. [文字列] （XSLT スタイルシート間での値の受け渡し）クエリー引数

セッションまたは Cookie を使用せずに、XSLT スタイルシート間でユーザ定義の任意の情報を渡します。このクエリー引数は、カスタム Web 公開 with XSLT のリクエストでのみ使用できます。

-token.[文字列] の文字列：0 から 9 の数字、a から z の小文字の英字、または A から Z の大文字の英字を含む、任意の長さの任意の英数字（空白を除く）の文字列を指定します。

ユーザ定義引数の値：URL エンコードされた任意の文字列

オプション：すべての XSLT リクエスト

例：

http://192.168.123.101/fmi/xsl/template/my_stylesheet.xsl?-db=employees&-lay=departments
&-grammar=fmresultset&-token.D100=Active&-findall

53 ページの「トークンを使用したスタイルシート間での情報の受け渡し」を参照してください。

付録 B
カスタム Web 公開のエラーコード

Web 公開エンジンでは、カスタム Web 公開で発生する可能性がある次の 3 種類のエラーコードがサポートされています。

- データベースおよびクエリー文字列のエラー。XML データリクエストが発生すると、常に、公開されているデータベースのエラーコードが Web 公開エンジンによって生成されます。次のセクション「FileMaker データベースのエラーコード番号」を参照してください。
- Web 公開エンジンのエラー。Web 公開エンジンが開発モードの場合に、Web 公開エンジン自体でエラーが発生したときは、特定のエラーページが生成されます。実作業モードの場合は、一般的なテキストメッセージが表示されます。99 ページの「Web 公開エンジンのエラーコード番号」を参照してください。
- FileMaker XSLT 拡張関数のエラー。XSLT スタイルシート内で fmxslt:check_error_status() 拡張関数を使用して、拡張関数が呼び出されたときに拡張関数のエラーステータスをチェックできます。101 ページの「FileMaker XSLT 拡張関数のエラーコード番号」を参照してください。

更新されたエラーコードについては、テクニカルサポートインフォメーションを参照してください。

FileMaker データベースのエラーコード番号

データが要求されると、常に XML 形式で公開されているデータベースのエラーコードが Web 公開エンジンによって生成されます。このタイプのエラーコードの値は、XML ドキュメントの先頭に、fmresultset 文法の場合は <error code> エレメント、FMPXMLRESULT または FMPDSORESULT 文法の場合は <ERRORCODE> エレメントでそれぞれ挿入されます。エラーコード 0 は、エラーが発生していないことを示します。

次に、fmresultset 文法のデータベースエラーコードの例を示します。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE fmresultset PUBLIC "-//FMI//DTD fmresultset//EN" "/fmi/xml/fmresultset.dtd">
<fmresultset xmlns="http://www.filemaker.com/xml/fmresultset" version="1.0">
    <error code="0"></error>
```

次に、FMPXMLRESULT 文法のデータベースエラーコードの例を示します。

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE FMPXMLRESULT PUBLIC "-//FMI//DTD FMPXMLRESULT//EN" "/fmi/xml/FMPXMLRESULT.dtd">
<fmpxmlresult xmlns="http://www.filemaker.com/fmpxmlresult">
    <ERRORCODE>0</ERRORCODE>
```

<error code> または <ERRORCODE> エレメントの値をチェックして適切に処理することは、カスタム Web 公開ソリューションの開発者の責任です。Web 公開エンジンによってデータベースエラーが処理されることはありません。

エラー番号	説明
-1	原因不明のエラー
0	エラーなし
1	ユーザによるキャンセル
2	メモリエラー
3	コマンドが使用できません（たとえば誤ったオペレーティングシステム、誤ったモードなど）
4	コマンドが見つかりません
5	コマンドが無効です（たとえば、[フィールド設定] スクリプトステップに計算式が指定されていない場合など）
6	ファイルが読み取り専用です
7	メモリ不足
8	空白の結果

エラー番号	説明
9	アクセス権が不十分です
10	要求されたデータが見つかりません
11	名前が有効ではありません
12	名前がすでに存在します
13	ファイルまたはオブジェクトが使用中です
14	範囲外
15	0で割ることができません
16	処理に失敗したため、再試行が必要です（たとえば、ユーザクエリーなど）
17	外国語の文字セットの UTF-16 への変換に失敗しました
18	続行するには、クライアントはアカウント情報を指定する必要があります
19	文字列に A から Z、a から z、0 から 9（ASCII）以外の文字が含まれています
20	コマンドまたは操作がスクリプトトリガによってキャンセルされました
100	ファイルが見つかりません
101	レコードが見つかりません
102	フィールドが見つかりません
103	リレーションシップが見つかりません
104	スクリプトが見つかりません
105	レイアウトが見つかりません
106	テーブルが見つかりません
107	索引が見つかりません
108	値一覧が見つかりません
109	アクセス権セットが見つかりません
110	関連テーブルが見つかりません
111	フィールドの繰り返しが無効です
112	ウインドウが見つかりません
113	関数が見つかりません
114	ファイル参照が見つかりません
115	メニューセットが見つかりません
116	レイアウトオブジェクトが見つかりません
117	データソースが見つかりません
130	ファイルが損傷しているか見つからないため、再インストールする必要があります
131	言語パックファイルが見つかりません（テンプレートなど）
200	レコードアクセスが拒否されました
201	フィールドを変更できません
202	フィールドアクセスが拒否されました
203	ファイルに印刷するレコードがないか、入力したパスワードでは印刷できません
204	ソート優先順位に指定されたフィールドにアクセスできません
205	ユーザに新規レコードを作成するアクセス権がありません。既存のデータはインポートしたデータで上書きされます
206	ユーザにパスワードの変更アクセス権がないか、変更可能なファイルではありません

エラー番号	説明
207	ユーザにデータベーススキーマを変更する十分なアクセス権がないか、変更可能なファイルではありません
208	パスワードに十分な文字が含まれていません
209	既存のパスワードと新規パスワードを同一にすることはできません
210	ユーザアカウントが非アクティブです
211	パスワードが期限切れです
212	ユーザアカウントまたはパスワードが無効です。再試行してください
213	ユーザアカウントまたはパスワードが存在しません
214	ログイン試行回数が多すぎます
215	管理者権限は複製できません
216	ゲストアカウントは複製できません
217	ユーザに管理者アカウントを変更する十分なアクセス権がありません
300	ファイルがロックされているか、使用中です
301	別のユーザがレコードを使用中です
302	別のユーザがテーブルを使用中です
303	別のユーザがデータベーススキーマを使用中です
304	別のユーザがレイアウトを使用中です
306	レコード修正 ID が一致しません
400	検索条件が空です
401	検索条件に一致するレコードがありません
402	選択したフィールドはルックアップの照合フィールドではありません
403	評価版の FileMaker Pro に設定されている最大レコード数の制限を超過しています
404	ソート優先順位が無効です
405	指定したレコード数が除外可能なレコード数を超過しています
406	全置換またはシリアル番号の再入力に指定された条件が無効です
407	片方または両方の照合フィールドが欠けています（無効なリレーションシップ）
408	指定されたフィールドのデータが不適切なため、この処理を実行できません
409	インポート順が無効です
410	エスクポート順が無効です
412	ファイルの修復に、誤ったバージョンの FileMaker Pro が使用されました
413	指定されたフィールドのフィールドタイプが不適切です
414	レイアウトに結果を表示できません
415	1 つまたは複数の必要な関連レコードが使用できません
416	データソーステーブルからプライマリキーが必要です
417	データベースが、サポートされているデータソースではありません
500	日付の値が入力値の制限を満たしていません
501	時刻の値が入力値の制限を満たしていません
502	数字の値が入力値の制限を満たしていません
503	フィールドの値が入力値の制限オプションに指定されている範囲内に入っていません
504	フィールドの値が入力値の制限オプションで要求されているようにユニークな値になっていません

エラー番号	説明
505	フィールドの値が入力値の制限オプションで要求されているようにデータベースファイル内の既存値になっていません
506	フィールドの値が入力値の制限オプションに指定されている値一覧に含まれていません
507	フィールドの値が入力値の制限オプションに指定されている計算式を満たしません
508	検索モードに無効な値が入力されました
509	フィールドに有効な値が必要です
510	関連する値が空であるか、使用できません
511	フィールド内の値が最大文字数を超過しました
512	レコードがすでに別のユーザによって変更されています
513	レコードを作成するには、レコードの少なくとも 1 つのフィールドに値がある必要があります
600	印刷エラーが発生しました
601	ヘッダとフッタの高さを加算するとページの高さを超えます
602	現在の段数設定ではボディ部分がページ内に収まりません
603	印刷接続が遮断されました
700	インポートできないファイルタイプです
706	EPSF ファイルにプレビューイメージがありません
707	グラフィックの変換ファイルが見つかりません
708	ファイルをインポートできないか、ファイルをインポートするにはカラーのコンピュータが必要です
709	QuickTime ムービーのインポートに失敗しました
710	データベースファイルが読み取り専用になっているため QuickTime ファイルの参照を更新できません
711	インポートの変換ファイルが見つかりません
714	入力したパスワードでは設定されている権限が不足しているためこの操作は認められていません
715	指定された Excel ワークシートまたは名前の付いた範囲がありません
716	ODBC インポートでは、DELETE、INSERT、または UPDATE を使用する SQL クエリーは使用できません
717	インポートまたはエクスポートを続行するための十分な XML/XSLT 情報がありません
718	（Xerces からの）XML ファイルの解析エラーです
719	（Xalan からの）XSL を使用した XML 変換エラーです
720	エクスポート時のエラー。対象のドキュメントフォーマットでは繰り返しフィールドはサポートされていません
721	パーサまたはトランスフォーマで原因不明のエラーが発生しました
722	フィールドのないファイルにデータをインポートすることはできません
723	インポート先のテーブルでレコードを追加または変更する権限がありません
724	インポート先のテーブルにレコードを追加する権限がありません
725	インポート先のテーブルでレコードを変更する権限がありません
726	インポートファイルのレコードの方がインポート先のテーブルのレコードよりも多くなっています。一部のレコードはインポートされません
727	インポート先のテーブルのレコードの方がインポートファイルのレコードよりも多くなっています。一部のレコードは更新されません
729	インポート中にエラーが発生しました。レコードをインポートすることができません
730	サポートされていない Excel のバージョンです。ファイルを Excel 7.0（Excel 95）、Excel 97、2000、XP または 2007 のフォーマットに変換して、もう一度実行してください

エラー番号	説明
731	インポート元のファイルにデータが含まれていません
732	このファイルには内部に他のファイルが含まれているため、挿入できません
733	テーブルをテーブル自体にインポートすることはできません
734	このファイルタイプをピクチャとして表示することはできません
735	このファイルタイプをピクチャとして表示することはできません。ファイルとして挿入および表示されます
736	この形式にエクスポートするには、データが大きすぎます。データは切り捨てられます
737	インポート元の Bento テーブルがありません
800	ファイルをディスク上に作成できません
801	システムディスクにテンポラリファイルを作成できません
802	ファイルを開くことができません このエラーの原因は、次の 1 つ以上です ■ 無効なデータベース名 ■ ファイルが FileMaker Server で閉じられている ■ 無効なアクセス権
803	ファイルが単独使用に設定されているか、またはホストが見つかりません
804	ファイルは現在の状態では読み取り専用として開くことができません
805	ファイルが損傷しています。修復コマンドを使用してください
806	このバージョンの FileMaker Pro ではファイルを開くことができません
807	ファイルが FileMaker Pro のファイルではないか、重大な損傷があります
808	アクセス権情報が壊れているため、ファイルを開くことができません
809	ディスク / ボリュームがいっぱいです
810	ディスク / ボリュームがロックされています
811	テンポラリファイルを FileMaker Pro ファイルとして開くことができません
813	ネットワーク上でレコードの同期エラーが発生しました
814	最大数のファイルがすでに開いているため、ファイルを開くことができません
815	ルックアップファイルを開くことができません
816	ファイルを変換できません
817	このソリューションに属していないため、ファイルを開くことができません
819	リモートファイルのローカルコピーを保存できません
820	ファイルを閉じる途中です
821	ホストによって接続解除されました
822	FMI ファイルが見つかりません。見つからないファイルを再インストールしてください
823	ファイルをシングルユーザに設定できません。ゲストが接続しています
824	ファイルが損傷しているか、FileMaker のファイルではありません
825	ファイルには保護ファイルを参照する権限がありません
900	スペルチェックのエンジンにエラーが発生しています
901	スペルチェック用のメイン辞書がインストールされていません
902	ヘルプシステムを起動できませんでした
903	共有ファイルではコマンドを使用できません
904	コマンドは、FileMaker Server がホスト管理しているファイル内でのみ使用できます

エラー番号	説明
905	アクティブなフィールドが選択されていません。アクティブなフィールドが存在する場合のみコマンドを使用することができます
906	現在のファイルは共有されていません。コマンドは、ファイルが共有されている場合のみ使用することができます
920	スペルチェックエンジンを初期化できません
921	編集するユーザ辞書をロードできません
922	ユーザ辞書が見つかりません
923	ユーザ辞書が読み取り専用です
951	予期しないエラーが発生しました
954	サポートされていない XML 文法です
955	データベース名がありません
956	データベースセッションが最大数を超過しました
957	コマンドが競合しています
958	クエリーに引数がありません
959	カスタム Web 公開テクノロジーが無効です
1200	一般的な計算エラーです
1201	関数の引数が足りません
1202	関数の引数が多すぎます
1203	計算式が未完了です
1204	数字、テキスト、フィールド名、または「(」を入れてください
1205	コメントは「*/」で終了できません
1206	テキストは半角のダブルクォーテーションマークで終わらなければなりません
1207	カッコが一致していません
1208	演算子または関数が見つからないか、「(」は指定できません
1209	名前（フィールド名またはレイアウト名）が見つかりません
1210	プラグイン関数はすでに登録されています
1211	この関数では一覧を使用できません
1212	演算子（+、-、* など）を入れてください
1213	この変数はすでに Let 関数で定義されています
1214	AVERAGE、COUNT、EXTEND、GETREPETITION、MAX、MIN、NPV、STDEV、SUM、および GETSUMMARY 関数で、フィールドの値を指定できない部分に式が使われています
1215	この引数は Get 関数の無効な引数です
1216	GetSummary 関数の 1 番目の引数は、集計フィールドのみに限られます
1217	区分けフィールドが無効です
1218	数字を評価できません
1219	フィールド固有の式にフィールドは使用できません
1220	フィールドタイプは標準にするか、計算する必要があります
1221	データタイプは数字、日付、時刻、またはタイムスタンプでなければなりません
1222	計算式を保存できません
1223	指定された関数はまだ実装されていません
1224	指定された関数は存在しません

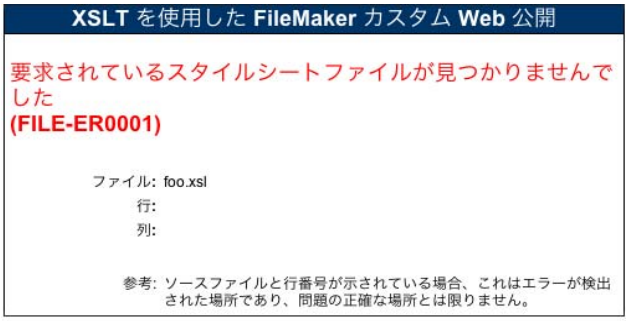
エラー番号	説明
1225	指定された関数は、このコンテキストではサポートされていません
1400	ODBC クライアントドライバの初期化に失敗しました。ODBC クライアントドライバが適切にインストールされていることを確認してください
1401	環境の割り当てに失敗しました（ODBC）
1402	環境の解放に失敗しました（ODBC）
1403	切断に失敗しました（ODBC）
1404	接続の割り当てに失敗しました（ODBC）
1405	接続の解放に失敗しました（ODBC）
1406	SQL API のチェックに失敗しました（ODBC）
1407	ステートメントの割り当てに失敗しました（ODBC）
1408	拡張エラー（ODBC）
1409	拡張エラー（ODBC）
1410	拡張エラー（ODBC）
1411	拡張エラー（ODBC）
1412	拡張エラー（ODBC）
1413	拡張エラー（ODBC）
1450	PHP アクセス権を拡張する操作が必要です
1451	現在のファイルをリモートにする操作が必要です
1501	SMTP の認証に失敗しました
1502	SMTP サーバーによって接続が拒否されました
1503	SSL でエラーが発生しました
1504	SMTP サーバーの接続を暗号化する必要があります
1505	指定された認証方法は SMTP サーバーではサポートされていません
1506	E メールは正常に送信されませんでした
1507	SMTP サーバーにログインできませんでした

Web 公開エンジンのエラーコード番号

Web 公開エンジンが開発モードの場合に、Web 公開エンジン自体でエラーが発生したときは、特定のエラーページが生成されます。このタイプのエラーは、次のように Web 公開エンジンが処理を実行できない場合など、さまざまな原因から発生することがあります。

- 要求されたスタイルシートファイル、または <xsl:include> によってネストされたスタイルシートファイルが見つからない
- ファイルに XML エラーがあるため、要求されたスタイルシートファイルまたはネストされたスタイルシートファイルを解析できない
- ファイルの XSLT または XPath エラーのため、ファイルからスタイルシートを生成できない
- CGI で XML 文法が正しく指定されていないため、リクエストを処理できない
- Web 公開コアと通信して XML を取得できない

Web 公開エンジンが開発モードで動作している場合は、このタイプのエラーのエラーページには、エラーメッセージと、カッコで囲まれたエラー番号が含まれます。例：



Web 公開エンジンが開発モードの場合のエラーページの例

次に、Web 公開エンジンの一部のエラーコードの値を示します。

エラーコード値	説明
QUERY-ER0001	-grammar クエリー引数で XML 文法が指定されていませんでした
QUERY-ER0002	「xxx」は FileMaker XSLT で有効な XML 文法ではありません
FILE-ER0001	要求されているスタイルシートファイルが見つかりませんでした
FILE-ER0002	要求されているファイルが見つかりませんでした
UNKNOWN	予期せぬエラーが発生しました
MCS-000 から MCS-600	予期せぬエラーが発生しました
MCS-601	タイプ「x」のリソースはサポートされていないため、リソース「x」をロードできませんでした
MCS-602	URL「x」を解決できませんでした
MCS-603	「x」に対する HTTP リクエストがタイプ「x」のエラーを返しました
MCS-604	予期せぬエラーのため、リソース「x」をロードできませんでした
MCS-605	content-type が無効なため、リソース「x」をロードできませんでした
MCS-606	ドキュメントの XML エラーのため、リソース「x」をロードできませんでした
MCS-607	認証の問題のため、リソース「x」をロードできませんでした
MCS-700	予期せぬエラーが発生しました
MCS-800	予期せぬエラーが発生しました

Web 公開エンジンが実作業モードの場合は、Web 公開エンジンのエラーに対して、「pe_server_error.html」に記述された次のデフォルトの一般的なテキストメッセージが表示されます。

FileMaker カスタム Web 公開を XSLT とともに使用中に、予期せぬエラーが発生しました。

デフォルトの「pe_server_error.html」ファイルには、このテキストメッセージが 6 つの言語で記述されています。

開発者は、ソリューションの必要に応じて、「pe_server_error.html」エラーページのテキストを編集できます。「pe_server_error.html」ファイルは、Web 公開エンジンをインストールしたホストの「publishing-engine」フォルダ内の「cwpe」フォルダにあります。

開発モードまたは実作業モードでの Web 公開エンジンの設定方法の詳細については、「FileMaker Server ヘルプ」を参照してください。

FileMaker XSLT 拡張関数のエラーコード番号

fmxslt:check_error_status() 拡張関数（68 ページの「拡張関数のエラーステータスのチェック」を参照）は、次の表に示すエラーの 1 つを返します。

エラーコード値	説明
-1	原因不明のエラー
0	エラーなし
	一般的なエラー
10000	無効なヘッダ名
10001	無効な HTTP ステータスコード
	セッションエラー
10100	原因不明のセッションエラー
10101	要求されたセッション名はすでに使用されています
10102	セッションにアクセスできませんでした。存在しない可能性があります
10103	セッションがタイムアウトしました
10104	指定されたセッションオブジェクトは存在しません
	メッセージングエラー
10200	原因不明のメッセージングエラー
10201	メッセージの書式設定エラー
10202	メッセージの SMTP フィールドのエラー
10203	メッセージの「To」フィールドのエラー
10204	メッセージの「From」フィールドのエラー
10205	メッセージの「CC」フィールドのエラー
10206	メッセージの「BCC」フィールドのエラー
10207	メッセージの「Subject」フィールドのエラー
10208	メッセージの「Reply-To」フィールドのエラー
10209	メッセージの本文のエラー
10210	再帰メールエラー - 電子メール用の XSLT スタイルシート内で send_email() を呼び出そうとしました
10211	SMTP 認証エラー - ログインに失敗したか、間違ったタイプの認証が指定されています
10212	無効な関数の使用法 - 電子メール用の XSLT スタイルシート内で set_header()、set_status_code()、または set_cookie() を呼び出そうとしました
10213	SMTP サーバーが無効であるか動作していません
	書式設定エラー
10300	原因不明の書式設定エラー
10301	無効な日付または時刻書式
10302	無効な日付書式
10303	無効な時刻書式
10304	無効な曜日書式

エラーコード値	説明
10305	不適切な形式の日付または時刻文字列
10306	不適切な形式の日付文字列
10307	不適切な形式の時刻文字列
10308	不適切な形式の曜日文字列
10309	サポートされていないテキストエンコード
10310	無効な URL エンコード
10311	正規表現パターンのエラー

索引

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

あ

い

え

お

か

く

け

こ

さ

し

す

せ

ち

つ

て

と

に

ね

は

ひ

ふ

へ

ほ

ま

め

も

ゆ

よ

れ

ろ