FileMaker® Server 13

カスタム Web 公開 with PHP

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

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

第 01 版

目次

はじめに

このガイドについて

このガイドでは、PHP、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。FileMaker Pro の詳細については、「FileMaker Pro ヘルプ」を参照してください。

このガイドでは、FileMaker Server 上でのカスタム Web 公開 with PHP に関する次の情報を説明します。

- PHP 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
- PHP 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データを取得するために FileMaker API for PHP を使用する方法

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」
イントラネットまたはインターネット上で Web ブラウザユーザにアクセス可能な FileMaker Pro および FileMaker Pro Advanced データベースのレイアウトの作成	『FileMaker WebDirect™ ガイド』
カスタム Web 公開 with PHP	『FileMaker Server カスタム Web 公開 with PHP』（このマニュアル）
カスタム Web 公開 with XML	『FileMaker Server カスタム Web 公開 with XML』
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 ソフトウェアでサポートされている SQL ステートメントと標準	『FileMaker SQL リファレンスガイド』

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。

FileMaker WebDirect 公開 : FileMaker WebDirect を使うと、データベースのレイアウトをすばやく簡単に 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、他のソフトウェアをインストールしなくても、FileMaker WebDirect ソリューションに接続してレコードを表示、編集、ソート、および検索することができます。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。

FileMaker WebDirect を使用するには、ホストコンピュータで FileMaker Server を実行する必要があります。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 WebDirect 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 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（FileMaker WebDirect を使用すれば、データベースでデータが更新されると同時に、Web ブラウザ内のデータも更新されます）。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 : FileMaker データベースをカスタム Web サイトに統合するには、FileMaker Server で使用できる カスタム Web 公開テクノロジーを使用します。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 FileMaker API for PHP を使用して、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 Web ページを自分でコーディングすることにより、ユーザインターフェースとユーザがデータと交信する方法を完全に管理できます。
- カスタム Web 公開 with XML: 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。FileMaker クエリーコマンドと引数とともに HTTP URL を使用することにより、FileMaker Server でホストされているデータベースに問い合わせて結果データを XML 形式でダウンロードし、結果として生成された XML データを任意の用途に使用できます。

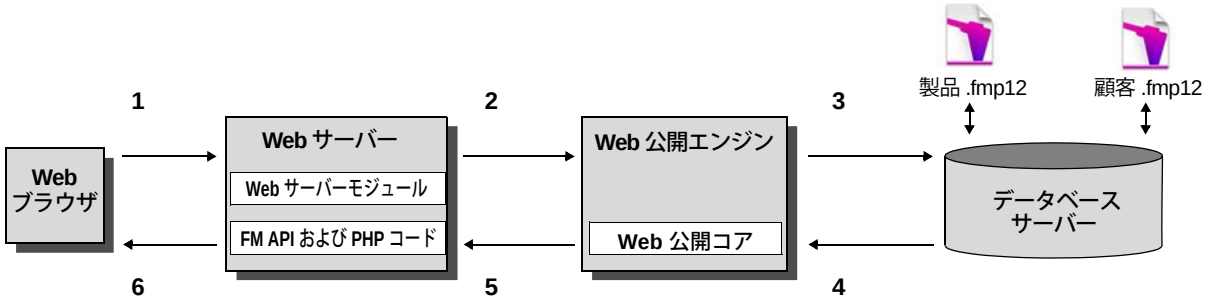
Web 公開エンジンについて

FileMaker WebDirect 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が使用されています。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: 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。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リクエストします。
4. FileMaker Server が、リクエストされた 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。
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 Pro ユーザーズガイド』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ほぼすべてのタイプ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。

カスタム Web 公開 with XML

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できます。

PHP と XML 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 から利用できます。
- 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。

XML 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 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。
- カスタム Web 公開を使用した XML データへのアクセス、および FileMaker Pro データベースからの XML エクスポートには、FMPXMLRESULT を使用できます。

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

第 2 章
カスタム Web 公開 with PHP について

カスタム Web 公開 with PHP では、PHP スクリプト言語を使用して FileMaker データベースからのデータをカスタマイズした Web ページレイアウトと統合できます。カスタム Web 公開 with PHP は、FileMaker API for PHP を提供します。これは FileMaker により作成された PHP クラスで、FileMaker Server がホストするデータベースにアクセスします。この PHP クラスは、FileMaker Server の Web 公開エンジンに接続し、ご使用の Web サーバーの PHP エンジンに対してデータを利用可能にします。

カスタム Web 公開 with PHP の主な機能

- オープンソース PHP スクリプト言語を使用する Web アプリケーションを作成します。FileMaker Server でサポートされている PHP 5 のバージョンを使用するか、PHP 5 の独自のバージョンを使用します。（独自の PHP バージョンの使用を選択した場合は、13 ページの「FileMaker API for PHP の手動によるインストール」を参照してください）。
- FileMaker Server 上でデータベースをホストします。FileMaker Server がデータベースをホストするので、FileMaker Pro はカスタム Web 公開には必要ありません。
- ホストされている FileMaker データベース内のレコードを作成、削除、編集、または複製できる PHP コードを記述します。記述したコードは、ホストされているデータベースに変更を確定する前に、フィールドおよびレコードの妥当性チェックを実行できます。
- レイアウト、ポータル、値一覧、および関連フィールドにアクセスする PHP コードを記述します。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。16 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。
- 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 PHP コードを記述します。FileMaker は 65 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。22 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。
- 複雑な検索条件を実行する PHP コードを記述します。

カスタム Web 公開の必要条件

このセクションでは、PHP 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に必要な事項、カスタム Web 公開ソリューションにアクセスするために Web ユーザに必要なこと、および Web 公開ソリューションをホストすることによるサーバーに与える影響について説明します。

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 PHP を使用してデータベースを公開するには次の条件が必要です。

- 3 つのコンポーネントを含む FileMaker Server の展開
  - Microsoft IIS（Windows）または Apache（OS X）のいずれかの Web サーバー。FileMaker Web サーバーモジュールは、Web サーバー上にインストールされます。
  - FileMaker Web 公開エンジン
  - FileMaker データベースサーバー

- Web サーバー上にインストールされた PHP。FileMaker Server では、サポートされている PHP 5 のバージョンをインストールするか、ユーザ独自のバージョンを使用できます。OS X 上の最低限必要な PHP のバージョンは、PHP 5.3.15 です。Windows 上の最低限必要な PHP のバージョンは、PHP 5.3.27 です。PHP の詳細については、http://php.net を参照してください。Web サーバーにインストールされた PHP のバージョンは、cURL（クライアント URL ライブラリ）機能対応でなければなりません。cURL については、http://php.net/curl を参照してください。

  重要 FileMaker Server でサポートされている PHP 5 をインストールする場合、OS X Server Admin ツールには表示されません（リストされないようになっています）。OS X Server Admin ツールを使用して PHP をオンにする場合は、FileMaker Server でサポートされている PHP 5 バージョンを無効化し、独自の PHP バージョンを有効化します。

- FileMaker Server でホストされている 1 つ以上の 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 PHP 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が必要です。

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、固有の静的（不変）な 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。

FileMaker API for PHP の手動によるインストール

FileMaker Server をインストールする際、FileMaker でサポートされているバージョンの PHP（PHP 5）をインストールするオプションを使用できます。すでに PHP エンジンのインストールおよび設定が終了し、FileMaker API for PHP のみを追加する場合は、FileMaker API for PHP クラスを手動でインストールし、PHP スクリプトで利用できるようにします。

FileMaker がサポートする PHP のバージョンをインストールしていない場合は、次の設定タスクをご使用の PHP エンジンのバージョン上で行ってください。

- php.ini 内の cURL モジュールを有効にする。
- php.ini 内の include_path 変数にある FileMaker API for PHP の場所を指定する。
- 日付と時刻を含むデータベースにアクセスしている場合、PEAR Date パッケージをインストールします。詳細については、http://pear.php.net/package/date/ を参照してください。

メモ FileMaker Server は、OS X 10.8 では PHP バージョン 5.3.15、OS X 10.9 では PHP バージョン 5.4.17、また Windows では PHP バージョン 5.3.27 でテストされています。PHP の機能を最大限に活用するためには適切なバージョンの使用をお勧めします。

FileMaker API for PHP を PHP スクリプトからアクセスできるようにする方法

FileMaker Server をインストールすると、FileMaker API for PHP パッケージが .zip ファイルとして次の場所に含まれます。

- IIS（Windows）：
  [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥Web Publishing￥FM_API_for_PHP_Standalone.zip
  [ドライブ] は、展開した FileMaker Server の Web サーバーコンポーネントが格納されているドライブです。
- Apache（OS X）：
  / ライブラリ /FileMaker Server/Web Publishing/FM_API_for_PHP_Standalone.zip

FM_API_for_PHP_Standalone.zip ファイルには、FileMaker.php という名前のファイルおよび FileMaker という名前のフォルダが含まれます。ファイルを解凍し、FileMaker.php ファイルおよび FileMaker フォルダを次の場所のいずれかにコピーします。

- PHP スクリプトが存在するフォルダ。
  - HTTP または HTTPS 経由の IIS（Windows）：
    [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥HTTPServer￥Conf
    [ドライブ] は、展開した FileMaker Server の Web 公開エンジンコンポーネントが格納されているドライブです。
  - HTTP 経由の Apache（OS X）: / ライブラリ /FileMaker Server/HTTPServer/htdocs
  - HTTPS 経由の Apache（OS X）: / ライブラリ /FileMaker Server/HTTPServer/htdocs/httpsRoot
- PHP インストール内の include_path ディレクトリの 1 つ。OS X のデフォルトの場所は、/usr/lib/php です。

この後の作業を開始するにあたって

カスタム Web 公開ソリューションの開発を開始するための推奨事項は次のとおりです。

- カスタム Web 公開を有効にするには、FileMaker Server Admin Console を使用します。「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。15 ページの「データベースのカスタム Web 公開 with PHP の有効化」を参照してください。
- FileMaker API for PHP 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は、第 5 章「FileMaker API for PHP の使用」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。

データベースのカスタム Web 公開 with PHP の有効化

公開する各データベースでカスタム Web 公開 with PHP を有効にする必要があります。有効にしなければ、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] または [拡張アクセス権の管理] 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。
2. 1 つ以上のアクセス権セットに fmphp 拡張アクセス権を割り当てて、カスタム Web 公開 with PHP を使用可能にします。
3. カスタム Web 公開の拡張アクセス権を含むアクセス権セットを適切なアカウント（たとえば、Admin およびゲストアカウント）に割り当てます。

   重要　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ために、感嘆符（!）やパーセント記号（%）などの特定の英数字以外の文字を含めることができます。コロン（:）は許可されません。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。

4. FileMaker Server Admin Console を使用して、データベースのホストが適切に設定されているか、およびデータベースが FileMaker Server からアクセスできるかを検証します。手順については、「FileMaker Server ヘルプ」を参照してください。

メモ　カスタム Web 公開 with PHP では、持続性のデータベースセッションを使用しないので、FileMaker Pro リレーションシップグラフの中で外部 ODBC データソースへの参照を使用すると、PHP ソリューションが利用できる機能を制限してしまう可能性があります。ご使用のデータベースが外部の SQL データソースからのデータにアクセスする場合、外部テーブルのレコードデータを更新できない可能性があります。

カスタム Web 公開 with PHP 用のレイアウトの作成

カスタム Web 公開 with PHP は、FileMaker Pro 内のデータに直接アクセスすることはできませんが、データベースの中で定義されたレイアウトを使用してアクセスできます。カスタム Web 公開 with PHP 用の固有のレイアウトを作成する必要はありませんが、PHP ソリューション専用のレイアウトを作成すると、次のような理由から役に立ちます。

- PHP ソリューションに含める必要のあるフィールド、ラベル、計算およびポータルに限定したレイアウトを作成することで、パフォーマンスが向上します。
- レコードが持つフィールドが減るので、データ処理が減少し、PHP コードが簡素化されます。
- データからインターフェースの設計作業を分離することで、Web ユーザ用にインターフェースをカスタマイズできます。

公開されたデータベースの保護

カスタム Web 公開 with PHP を使用すると、公開したデータベースへのアクセスを制限できます。次のメソッドを使用することができます。

- カスタム Web 公開 with PHP に使用されるデータベースアカウントにパスワードを要求します。
- カスタム Web 公開 with PHP の拡張アクセス権を、アクセスを許可するアクセス権セットでのみ有効にします。
- 特定のデータベースに対してカスタム Web 公開 with PHP を無効にするには、そのデータベースのすべてのアクセス権セットの fmphp 拡張アクセス権を選択解除します。「FileMaker Pro ヘルプ」を参照してください。
- FileMaker Server Admin Console を使用して、Web 公開エンジン内のすべてのカスタム Web 公開ソリューション向けにカスタム Web 公開を有効化または無効化します。『FileMaker Server 入門ガイド』および「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指定できます。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。

FileMaker Server では、ディスクに書き込むデータとクライアントに転送するデータの暗号化がサポートされています。

- FileMaker Pro Advanced のデータベース暗号化機能を使用してデータベースを暗号化します。暗号化によって、FileMaker データベースファイルと、ディスクに書き込まれる一時ファイルが保護されます。データベースの暗号化については、『FileMaker Pro ユーザーズガイド』、『FileMaker Server 入門ガイド』および「FileMaker Pro ヘルプ」を参照してください。
  - FileMaker Server 上でホストされている暗号化されたデータベースは、Admin Console またはコマンドラインインターフェイス（CLI）を使用して開きます。FileMaker Server 管理者としてデータベース暗号化パスワードを使用してファイルを開いて FileMaker クライアントが暗号化されたデータベースを使用できるようにします。
  - 暗号化された FileMaker データベースが FileMaker Server 管理者によって暗号化パスワードを使用して開かれると FileMaker クライアントは、暗号化パスワードを入力することなく暗号化されたデータベースにアクセスできます。暗号化されたデータベースを開く方法については、「FileMaker Server ヘルプ」を参照してください。
- Web サーバーと Web 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 接続は、HTTPS 接続でアクセスされます。一度設定して動作が開始すれば、それ以上クライアントが操作する必要はありません。SSL の有効化、設定および保守の詳細については、お使いの 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 Pro 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 http://www.filemaker.co.jp/documentation から入手することができます。

保護されたデータベースへのアクセス

カスタム Web 公開 with PHP を使用すると、データベースのパスワード保護、データベースの暗号化、セキュリティ保護された接続によって公開したデータベースへのアクセスを制限できます。PHP ソリューションを使用して Web ユーザがデータベースにアクセスする際には、PHP コードは FileMaker API for PHP を使用してデータベースに証明書を提供する必要があります。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、fmphp 拡張アクセス権が有効化されていない場合、FileMaker API for PHP はエラーを返すので、PHP コードはユーザのログイン情報を提供する必要があります。

FileMaker API for PHP チュートリアルには、setProperty() メソッドを使用して保護されたデータベースにユーザ名およびパスワードを設定する方法を示す例が含まれています。28 ページの「FileMaker API for PHP チュートリアル」を参照してください。

次に、カスタム Web 公開を使用してパスワードで保護された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。

- カスタム Web 公開対応のアカウントにパスワードが割り当てられていない場合、PHP ソリューションはアカウント名のみ提供する必要があります。
- ゲストアカウントが無効な場合は、PHP ソリューションはアカウント名およびパスワードを提供する必要があります。PHP ソリューションは、Web ユーザにアカウント名およびパスワードの入力を求めるか、PHP コードの中にアカウント名およびパスワードを保存することができます。アカウントには、fmphp 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化され、fmphp 拡張アクセス権が有効になっている場合 :
  - PHP ソリューションでは、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に Web ユーザに求める必要はありません。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。
  - 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。
- PHP ソリューションでは、[再ログイン] スクリプトステップを使用して、ユーザが異なるアカウントを使用してログインできます（たとえば、ゲストアカウントからより権限の大きいアカウントへ切り替えるなど）。「FileMaker Pro ヘルプ」を参照してください。ただし PHP 接続では、持続的なデータベースセッションを使用しないので、PHP ソリューションは次に続くそれぞれのリクエストに使用するために、アカウント名およびパスワードを保存する必要があります。

  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を使用して、この機能をデータベースに対して有効化し、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。

Web 上でのオブジェクトフィールドの内容の公開

オブジェクトフィールドの内容は、データベースに埋め込んだり、相対パスを使用した参照でリンクしたり、外部に保存できます。

データベースに埋め込まれたオブジェクトフィールド

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、次の手順を実行してオブジェクトフィールドのオブジェクトを PHP ソリューションで使用します。

- FileMaker API for PHP を使用してデータベースオブジェクト ($fm) を適切な資格情報（アカウント名とパスワード）で定義します。

```
$fm = new FileMaker();
$fm->setProperty('database', $databaseName);
$fm->setProperty('username', $userName);
$fm->setProperty('password', $passWord);
```

- 正しい HTML タグを使用して、オブジェクトフィールドに含まれている Web 互換オブジェクトの種類を示し、HTML タグのソース属性向けのファイルパスを表す URL 文字列を作成します。

```
<IMG src="img.php?-url=<?php echo urlencode($record->getField('Cover Image')); ?>">
```

- 次に、getContainerData() メソッドを使用してオブジェクトフィールドのオブジェクトを取得します。

```
echo $fm->getContainerData($_GET['-url']);
```

FileMaker API for PHP チュートリアルには、オブジェクトフィールドの使用方法を示すその他の例が含まれています。28 ページの「FileMaker API for PHP チュートリアル」を参照してください。

メモ

- Web 公開エンジンは、インタラクティブオブジェクトのオーディオファイル (.mp3)、ビデオファイル (.mov、.mp4、.avi を推奨) および PDF ファイルのプログレッシブダウンロードをサポートしています。たとえば、Web ユーザは、ムービーファイルが完全にダウンロードされる前にムービーの再生を開始できます。プログレッシブダウンロードを可能にするには、ストリーミングをサポートするか、Web での表示に最適化するオプションを使用してファイルを作成する必要がある場合があります。たとえば、PDF ファイルは、[Web 表示用に最適化] オプションを使用して作成します。
- FileMaker Server の保護された接続を有効にする設定が選択されていない場合、FileMaker Server でデータ転送に使用する接続は転送中に暗号化されません。
  - FileMaker クライアントは、わずかな遅延後インタラクティブオブジェクトデータを認識します。
  - FileMaker Server は、FileMaker Pro、FileMaker Go、または Web クライアントがデータをリクエストしたときに、サーバー上のキャッシュフォルダにオブジェクトフィールドデータを暗号化します。データは、FileMaker Server が定期的にキャッシュフォルダを空にするまで、サーバーのキャッシュフォルダで 2 時間暗号化されたままになります。データはクライアント上にローカルにはキャッシュされません。
- FileMaker Server の保護された接続を有効にする設定が選択されている場合、FileMaker Server ではセキュア接続を使用してデータを転送します。FileMaker クライアントは、ユーザが交信する前にオブジェクトデータを完全にダウンロードします。一時キャッシュファイルが作成されず、データは転送中に暗号化されるため、データは、ソリューションがローカルデータベースであるかのように安全です。

保護された接続を有効にする設定が変更された場合、新しい設定を有効にするにはデータベースサーバーを停止して再起動する必要があります。

参照ファイルを含むオブジェクトフィールド

オブジェクトフィールドにファイル参照が保存されている場合は、PHP コードで getContainerData() メソッドを使用してデータベースからオブジェクトフィールドのオブジェクトを取得、または getContainerDataURL() メソッドを使用してオブジェクトフィールドのオブジェクトの完全修飾 URL を取得できます。

次の手順に従って、Web 公開エンジンを使用して参照先ファイルを公開する必要があります。

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 サーバーの次の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。
   - HTTP または HTTPS 経由の IIS（Windows）の場合 :
     [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥HTTPServer￥Conf
     [ドライブ] は、展開した FileMaker Server の Web 公開エンジンコンポーネントが格納されているドライブです。
   - HTTP 経由の Apache（OS X）の場合 : / ライブラリ /FileMaker Server/HTTPServer/htdocs
   - HTTP 経由の Apache（OS X）の場合 : / ライブラリ /FileMaker Server/HTTPServer/htdocs/httpsRoot

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（Multipurpose Internet Mail Extensions）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。インターネットに対して登録されている最新の MIME 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。
- オブジェクトフィールドに保存されたすべての QuickTime ムービーは参照として保存されます。

外部に保存されたデータを含むオブジェクトフィールド

オブジェクトフィールドがオブジェクトを外部に保存している場合（フィールドオプションのダイアログボックスで [オブジェクトデータを外部に保存] を選択した場合）、PHP コードは getContainerDataURL() メソッドを使用してオブジェクトフィールドのオブジェクトの完全修飾 URL を取得する必要があります。

FileMaker API for PHP を使用して、適切な資格情報（アカウント名およびパスワード）を持つデータベースオブジェクトを定義し、getContainerDataURL() メソッドを使用してオブジェクトフィールドのデータを取得します。

HTML img タグを使用した画像の表示の例

```
$fm=new FileMaker($database, $hostspec, $user, $password);
$findCommand = $fm->newFindCommand($layout);
$findCommand->addFindCriterion('type', 'png');
$result = $findCommand->execute();
$records = $result->getRecords();
foreach ($records as $record) {
    echo $record->getField('container').'<br>';
    // For images, use the HTML img tag
    echo '<img src="'.$fm->
      getContainerDataURL($record->getField('container')) .'">';
    break;
    }
```

HTML embed タグを使用した埋め込みデータの表示の例

```
$fm=new FileMaker($database, $hostspec, $user, $password);
$findCommand = $fm->newFindCommand($layout);
$findCommand->addFindCriterion('type', 'pdf');
$result = $findCommand->execute();
$records = $result->getRecords();
foreach ($records as $record) {
    echo $record->getField('container').'<br>';
    // For movies and PDF files, use the HTML embed tag
    //echo '<embed src="'.$fm->
      getContainerDataURL($record->getField('container')) .'">';
    break;
    }
```

オブジェクトフィールドデータの FileMaker Server へのアップロード

データベースのアップロードに FileMaker Pro を使用する際は、外部に保存されたオブジェクトフィールドデータは、プロセスの一環として FileMaker Server にアップロードされます。データベースの FileMaker Server への転送については、「FileMaker Pro ヘルプ」を参照してください。

外部に保存されたオブジェクトを含むオブジェクトフィールドを使用しているデータベースを手動でアップロードする場合、次の操作を行って外部に保存されたオブジェクトを Web 公開エンジンを使用して公開する必要があります。

データベースを手動でアップロードするには :

1. データベースファイルをサーバー上の適切な場所に配置します。FileMaker Server で開く FileMaker Pro データベースファイル、またはそれらのファイルへのショートカット（Windows）またはエイリアス（OS X）を、次のフォルダに配置します。
   - Windows: [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥Data￥Databases￥
     [ドライブ] は、システムが起動されるプライマリドライブです。
   - OS X: / ライブラリ /FileMaker Server/Data/Databases/

   または、オプションで指定した追加データベースフォルダにファイルを配置することもできます。

2. データベースを配置したフォルダ内に、「RC_Data_FMS」という名前のフォルダを作成します（存在していない場合）。
3. 「RC_Data_FMS」フォルダの中に、データベース名と同じ名前のフォルダを作成します。たとえば、データベース名が Customers の場合は、Customers というフォルダを作成します。作成した新しいフォルダに外部に保存されたオブジェクトを配置します。

   メモ データベースが FileMaker Server 上でホストされている場合は、複数のデータベース間で共通のオブジェクトのフォルダを共有する方法はありません。各データベースのオブジェクトは、データベース名と同じ名前で識別されたフォルダにある必要があります。

4. OS X から共有するファイルでは、fmsadmin グループに属するようにファイルを変更します。

   データベースの手動アップロードの詳細については、「FileMaker Server ヘルプ」を参照してください。

メモ

- Web 公開エンジンは、インタラクティブオブジェクトのオーディオファイル (.mp3)、ビデオファイル (.mov、.mp4、.avi を推奨) および PDF ファイルのプログレッシブダウンロードをサポートしています。たとえば、Web ユーザは、ムービーファイルが完全にダウンロードされる前にムービーの再生を開始できます。プログレッシブダウンロードを可能にするには、ストリーミングをサポートするか、Web での表示に最適化するオプションを使用してファイルを作成する必要がある場合があります。たとえば、PDF ファイルは、[Web 表示用に最適化] オプションを使用して作成します。
- FileMaker Server の保護された接続を有効にする設定が選択されていない場合、FileMaker Server でデータ転送に使用する接続は転送中に暗号化されません。
  - FileMaker クライアントは、わずかな遅延後インタラクティブオブジェクトデータを認識します。
  - FileMaker Server は、FileMaker Pro、FileMaker Go、または Web クライアントがデータをリクエストしたときに、サーバー上のキャッシュフォルダにオブジェクトフィールドデータを暗号化します。データは、FileMaker Server が定期的にキャッシュフォルダを空にするまで、サーバーのキャッシュフォルダで 2 時間暗号化されたままになります。データはクライアント上にローカルにはキャッシュされません。
- FileMaker Server の保護された接続を有効にする設定が選択されている場合、FileMaker Server ではセキュア接続を使用してデータを転送します。FileMaker クライアントは、ユーザが交信する前にオブジェクトデータを完全にダウンロードします。一時キャッシュファイルが作成されず、データは転送中に暗号化されるため、データは、ソリューションがローカルデータベースであるかのように安全です。

保護された接続を有効にする設定が変更された場合、新しい設定を有効にするにはデータベースサーバーを停止して再起動する必要があります。

Web ユーザがオブジェクトフィールドのオブジェクトを表示する方法

Web 公開エンジンを使用してデータベースを公開する場合、オブジェクトフィールドのオブジェクトには次の制限が適用されます。

- 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オブジェクトをデータベースにアップロードすることはできません。
- サムネールを有効化したオブジェクトフィールドを使用するデータベースの場合、Web 公開エンジンはサムネールではなく完全なファイルをダウンロードします。

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、一連のタスクを実行できます。FileMaker スクリプトを使用すると、たとえば [パスワード変更] スクリプトステップを使用して 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など、他の方法ではサポートされないタスクを実行することもできます。

FileMaker は 65 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。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 [カスタム Web 公開] を選択します。グレーで表示されている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。

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、個々のユーザに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制限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」が含まれる場合、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。
- ファイルを変換した場合には、Web ユーザが実行する可能性のある各スクリプトを開いて、[スクリプトの編集] ウインドウの [互換性を表示] リストから [Web 公開] を選択し、そのスクリプトがインスタント Web 公開で正しく実行されるようにする必要があります。

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 公開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すべての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。 メモ 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	これら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、スクリプトとユーザの操作（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できます。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。

メモ ファイルを開いたときに実行されるスクリプトを指定するには、OnFirstWindowOpen スクリプトトリガを使用する必要があります。同様に、ファイルを閉じたときに実行されるスクリプトを指定するには、OnLastWindowClose スクリプトトリガを使用する必要があります。

第 4 章
カスタム Web 公開 with PHP の概要

FileMaker API for PHP は、FileMaker データベースから PHP ソリューションへデータを統合するのに役立ちます。この章では、PHP が FileMaker Server のカスタム Web 公開エンジンと連携する方法について説明します。FileMaker API for PHP の詳細については、第 5 章「FileMaker API for PHP の使用」を参照してください。

Web 公開エンジンと PHP ソリューションの連携方法

FileMaker Server は、Web サーバー、Web 公開エンジン、およびデータベースサーバーという 3 つのコンポーネントから構成されます。（これらのコンポーネントは、1 台または 2 台のマシンで展開することができます。詳細については、『FileMaker Server 入門ガイド』を参照してください。）FileMaker Server は、PHP ファイルを PHP エンジンがインストールされている Web サーバー上に配置して、PHP ソリューションをホストします。

- Web ユーザが PHP ソリューションを開くと、Web サーバーはリクエストを PHP コードが処理される PHP エンジンにルーティングします。
- PHP コードが FileMaker API for PHP への呼び出しを含む場合は、それらの呼び出しは解釈され、Web 公開エンジンへのリクエストとして送信されます。
- 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リクエストします。
- データベースサーバーが、リクエストされたデータを Web 公開エンジンに送信します。
- Web 公開エンジンは、API の呼び出しに応えて、Web サーバー上の PHP エンジンへデータを送信します。
- PHP ソリューションはデータを処理して Web ユーザに表示します。

カスタム Web 公開 with PHP の一般手順

次に、カスタム Web 公開 with PHP を使用するための手順の概要を示します。

1. Admin Console で、PHP 公開が有効になっていることを確認します。『FileMaker Server 入門ガイド』を参照してください。
2. Admin Console で、[データベース] ウインドウを選択し、公開する各 FileMaker データベースのカスタム Web 公開 with PHP の fmphp 拡張アクセス権が有効になっていることを確認します。

   必要な場合、FileMaker Pro を使用してデータベースのカスタム Web 公開を有効にします。第 3 章「データベースのカスタム Web 公開の準備」を参照してください。

   メモ PHP ソリューションを開発する際は、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. PHP オーサリングツールを使用して FileMaker API 関数を PHP コードに統合して FileMaker データにアクセスし、PHP ソリューションを作成します。第 5 章「FileMaker API for PHP の使用」を参照してください。

4. Web サーバーの次のフォルダに、ご使用のサイトのディレクトリ構造およびファイルをコピーまたは移動します。
   - HTTP または HTTPS 経由の IIS（Windows）: [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥HTTPServer￥Conf
     [ドライブ] は、展開した FileMaker Server の Web 公開エンジンコンポーネントが格納されているドライブです。
   - HTTP 経由の Apache（OS X）: / ライブラリ /FileMaker Server/HTTPServer/htdocs
   - HTTPS 経由の Apache（OS X）: / ライブラリ /FileMaker Server/HTTPServer/htdocs/httpsRoot
5. 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、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。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。

   18 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」を参照してください。
6. 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。
7. 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、サイトをテストします。
8. サイトを使用可能にし、ユーザに通知します。Web ユーザが入力する URL には以下の形式が使用されます。

   http://< サーバー >/< サイトパス >
   - < サーバー > は、FileMaker Server が存在しているコンピュータです。
   - < サイトパス > とは、上記の手順 4 で使用したディレクトリ構造によって決定される、サイトのホームページへの相対パスです。

   たとえば、ご使用の Web サーバーのアドレスが 192.168.123.101 で、サイトのホームページが c:￥Inetpub￥wwwroot￥customers￥index.php の Web サーバー上にある場合、Web ユーザは次のように URL を入力します。

   http://192.168.123.101/customers/index.php

メモ PHP 5 では、Latin-1 (ISO-8859-1) エンコードを使用します。FileMaker Server は、Unicode (UTF-8) データを返します。FileMaker Server Admin Console を使用して、サイト用にデフォルトの文字コードを指定します。PHP サイトには、UTF-8 または ISO-8859-1 のいずれかを指定できますが、UTF-8 が推奨されます。サイトの PHP ファイルの <HEAD> セクションにある charset 属性に同じ設定を指定します。

PHP ソリューションの展開と使用の詳細については、第 6 章「サイトのステージング、テスト、および監視」を参照してください。

第 5 章
FileMaker API for PHP の使用

FileMaker API for PHP には、FileMaker データベースに対するオブジェクト指向インターフェースを提供する PHP クラス （FileMaker クラス）が実装されています。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。

FileMaker API for PHP は、FileMaker Pro データベース内ですでに使用可能な次の機能を PHP コードで実行できるようにします。

- レコードの作成、削除、編集、または複製
- 検索条件の実行
- フィールドおよびレコードの妥当性チェックの実行
- レイアウトの使用
- FileMaker スクリプトの実行
- ポータルおよび関連レコードの表示
- 値一覧の使用

この章では、FileMaker クラスオブジェクトの使用方法、およびこれらの一般的な機能を PHP ソリューションに追加するメソッドを説明します。この章は、FileMaker API for PHP 全体をカバーするものではありませんが、主要なオブジェクトおよびメソッドを紹介します。

追加情報の入手場所

FileMaker API for PHP の詳細については、次のリソースを参照してください。

すでに PHP エンジンのインストールおよび設定が終了し、FileMaker API for PHP を追加するだけの場合は、13 ページの「FileMaker API for PHP の手動によるインストール」を参照してください。

FileMaker API for PHP リファレンス

FileMaker API for PHP をインストールしている場合は、展開した FileMaker Server の Web サーバーコンポーネントでリファレンス情報を参照できます。

- IIS（Windows）:
  [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥Documentation￥PHP API Documentation￥index.html
  [ドライブ] は、展開した FileMaker Server の Web サーバーコンポーネントが格納されているドライブです。
- Apache（OS X）: / ライブラリ /FileMaker Server/Documentation/PHP API Documentation/index.html

FileMaker API for PHP チュートリアル

FileMaker API for PHP をインストールしている場合は、展開した FileMaker Server の Web サーバーコンポーネントでチュートリアルを参照できます。

- IIS（Windows）: [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥Examples￥PHP￥Tutorial
  [ドライブ] は、展開した FileMaker Server の Web サーバーコンポーネントが格納されているドライブです。
- Apache（OS X）: / ライブラリ /FileMaker Server/Examples/PHP/Tutorial

これらの PHP チュートリアルファイルを利用可能にするには、Web サーバーのルートフォルダにファイルをコピーします。

FileMaker API for PHP の例

FileMaker API for PHP をインストールしている場合は、展開した FileMaker Server の Web サーバーコンポーネントで追加の例を参照できます。

- IIS（Windows）: [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥Examples￥PHP￥API Examples
  [ドライブ] は、展開した FileMaker Server の Web サーバーコンポーネントが格納されているドライブです。
- Apache（OS X）: / ライブラリ /FileMaker Server/Examples/PHP/API Examples

これらの API の例のファイルを利用可能するには、Web サーバーのルートフォルダにファイルをコピーします。

FileMaker クラスの使い方

PHP ソリューションで FileMaker クラスを使用するには、PHP コードに次の文を追加します。
require_once ('FileMaker.php');

FileMaker クラスオブジェクト

FileMaker クラスは、FileMaker Pro のデータベースからデータを取得するのに使用できるクラスオブジェクトを定義します。

クラスオブジェクト	オブジェクトを使用して行う処理
FileMaker データベース	データベースのプロパティの定義 FileMaker Pro データベースファイルへの接続 FileMaker API for PHP の情報の取得
コマンド	レコード追加、レコード削除、レコード複製、レコード編集、検索条件実行、およびスクリプト実行コマンドの作成
レイアウト	データベースレイアウトの使用
レコード	レコードデータの使用
フィールド	フィールドデータの使用
関連セット	ポータルレコードの使用
結果	検索条件から返されたレコードの処理
エラー	エラーが発生したかどうかの確認 エラーの処理

FileMaker のコマンドオブジェクト

FileMaker クラスは、特定のコマンドのインスタンスを作成し、コマンドのパラメータを指定するのに使用する基本コマンドオブジェクトを定義します。コマンドを実行するには、execute() メソッドを呼び出す必要があります。

FileMaker クラスは、次の特定のコマンドを定義します。

- Add コマンド
- Compound Find コマンド
- Delete コマンド
- Duplicate コマンド
- Edit コマンド
- Find コマンド、Find All コマンド、Find Any コマンド
- Find Request コマンド（Compound Find コマンドに追加される）
- Perform Script コマンド

これらのコマンドについては、次のセクションで詳しく説明されています。

- 30 ページの「レコードの使用」
- 31 ページの「FileMaker スクリプトの実行」
- 37 ページの「検索条件の実行」

FileMaker データベースへの接続

FileMaker クラスは、サーバーまたはデータベースに接続するためにインスタンスを作成するデータベースオブジェクトを定義します。クラスコンストラクタを使用するか、setProperty() メソッドを呼び出してオブジェクトのプロパティを定義します。

例 : サーバーに接続し、データベースの一覧を表示

```
$fm = new FileMaker();
$databases = $fm->listDatabases();
```

例 : サーバー上の特定のデータベースへ接続

ユーザ名とパスワードのプロパティによって、この接続用のアクセス権セットが決まります。

```
$fm = new FileMaker();
$fm->setProperty('database', 'questionnaire');
$fm->setProperty('hostspec', 'http://192.168.100.110');
$fm->setProperty('username', 'web');
$fm->setProperty('password', 'web');
```

メモ hostspec プロパティは、デフォルトで http://localhost という値になります。PHP エンジンが展開した FileMaker Server の Web サーバーコンポーネントと同じマシン上で動作している場合は、hostspec プロパティを指定する必要はありません。PHP エンジンが異なるマシン上にある場合、hostspec プロパティを使用して、展開した FileMaker Server の Web サーバーコンポーネントの場所を指定します。

レコードの使用

FileMaker クラスは、レコードを使用するためにインスタンスを作成するレコードオブジェクトを定義します。レコードオブジェクトのインスタンスは、FileMaker Pro データベースの 1 つのレコードを表します。レコードオブジェクトを、Add、Delete、Duplicate、および Edit コマンドと使用して、レコード内のデータを変更します。検索コマンド（Find、Find All、Find Any、および Compound Find）は、レコードオブジェクの配列を返します。

レコードの作成

レコードを作成するには、次の 2 つの方法があります。

- createRecord() メソッドを使用します（レイアウト名を指定、およびフィールド値の配列をオプションで指定）。新規レコードオブジェクトでは個別に値を設定することもできます。

  createRecord() メソッドは、新規レコードをデータベースに保存しません。レコードをデータベースに保存するには、commit() メソッドを呼び出します。

  例 :

  ```
  $rec = $fm->createRecord('Form View', $values);
  $result = $rec->commit();
  ```

- Add コマンドを使用します。newAddCommand() メソッドを使用し、レイアウト名およびレコードデータを持つ配列を指定して FileMaker_Command_Add オブジェクトを作成します。レコードをデータベースに保存するには、execute() メソッドを呼び出します。

  例 :

  ```
  $newAdd =& $fm->newAddCommand('Respondent', $respondent_data);
  $result = $newAdd->execute();
  ```

レコードの複製

Duplicate コマンドを使用して既存のレコードを複製します。newDuplicateCommand() メソッドを使用し、レイアウト名および複製するレコードのレコード ID を指定して、FileMaker_Command_Duplicate オブジェクトを作成します。その後、execute() メソッドを呼び出して、レコードを複製します。

例

```
$newDuplicate = $fm->newDuplicateCommand('Respondent', $rec_ID);
$result = $newDuplicate->execute();
```

レコードの編集

レコードを編集するには、次の 2 つの方法があります。

- Edit コマンドを使用します。newEditCommand() メソッドを使用し、レイアウト名、編集するレコードのレコード ID、および更新する値の配列を指定して、FileMaker_Command_Edit オブジェクトを作成します。その後、execute() メソッドを呼び出して、レコードを編集します。

  例 :

  ```
  $newEdit =& $fm->newEditCommand('Respondent', $rec_ID, $respondent_data);
  $result = $newEdit->execute();
  ```

- レコードオブジェクトを使用します。データベースからレコードを取得し、フィールド値を変更し、commit() メソッドを呼び出してレコードを編集します。

  例 :

  ```
  $rec = $fm->getRecordById('Form View', $rec_ID);
  $rec->setField('Name', $nameEntered);
  $result = $rec->commit();
  ```

レコードの削除

レコードを削除するには、次の 2 つの方法があります。

- データベースからレコードを取得し、delete() メソッドを呼び出します。

  例 :

  ```
  $rec = $fm->getRecordById('Form View', $rec_ID);
  $rec->delete();
  ```

- Delete コマンドを使用して既存のレコードを削除します。newDeleteCommand() メソッドを使用し、レイアウト名および削除するレコードのレコード ID を指定して、FileMaker_Command_Delete オブジェクトを作成します。その後、execute() メソッドを呼び出して、レコードを削除します。

  例 :

  ```
  $newDelete =& $fm->newDeleteCommand('Respondent', $rec_ID);
  $result = $newDelete->execute();
  ```

FileMaker スクリプトの実行

FileMaker のスクリプトは、スクリプトステップの名前付きのセットです。FileMaker クラスは、FileMaker Pro のデータベースで定義された FileMaker スクリプトを使用可能にするためのいくつかのメソッドを定義します。Web 互換のスクリプトステップ（Web ソリューションの中で実行できるスクリプトステップ）については、22 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。

利用可能なスクリプト一覧の取得

listScripts() メソッドを使用して、現在接続されているデータベースから利用可能なスクリプトの一覧を取得します。listScripts() メソッドは、データベース接続が定義された際に指定されたユーザ名およびパスワードで実行できるスクリプトの配列を返します。（29 ページの「FileMaker データベースへの接続」を参照してください。）

例

```
$scripts = $fm->listScripts();
```

FileMaker スクリプトの実行

newPerformScriptCommand() メソッドを使用し、レイアウト、スクリプト名、および必要なスクリプトパラメータを指定して、FileMaker_Command_PerformScript オブジェクトを作成します。その後、execute() を呼び出してスクリプトを実行します。

例

```
$newPerformScript =& $fm->newPerformScriptCommand('Order Summary', 'ComputeTotal');
$result = $newPerformScript->execute();
```

コマンド実行前のスクリプトの実行

setPreCommandScript() メソッドを使用して、コマンドが実行される前に実行するスクリプトを指定します。次の例では、検索コマンドを使用していますが、任意のコマンドと共に setPreCommandScript() メソッドを使用できます。

例

```
$findCommand =& $fm->newFindCommand('Students');
$findCommand->addFindCriterion('GPA', $searchValue);
$findCommand->setPreCommandScript('UpdateGPA');
$result = $findCommand->execute();
```

結果セットをソートする前のスクリプトの実行

setPreSortScript() メソッドを使用して、検索の結果セットが生成された後、結果セットをソートする前に実行するスクリプトを指定します。詳細については、38 ページの「Find コマンドの使用」を参照してください。

例

```
$findCommand =& $fm->newFindCommand('Students');
$findCommand->setPreSortScript('RemoveExpelled');
```

結果セットが生成された後のスクリプトの実行

setScript() メソッドを使用して、検索の結果セットが生成された後に実行するスクリプトを指定します。詳細については、38 ページの「Find コマンドの使用」を参照してください。

例

```
$findCommand =& $fm->newFindCommand('Students');
$findCommand->setScript('myScript','param1|param2|param3');
```

スクリプトの実行順序

setPreCommandScript()、setPreSortScript()、および setScript() メソッドを setResultLayout() および addSortRule() メソッドと共に単一のコマンドに指定できます。次に、FileMaker Server と Web 公開エンジンがこれらのメソッドを処理する順序を示します。

1. setPreCommandScript() メソッドで指定されているスクリプトを実行します（指定されている場合）。
2. 検索またはレコードの削除コマンドのような、コマンド自体を処理します。
3. setPreSortScript() メソッドで指定されているスクリプトを実行します（指定されている場合）。
4. addSortRule() メソッドが指定されている場合は、検索の結果セットをソートします。
5. setResultLayout() メソッド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）。

6. setScript() メソッドで指定されているスクリプトを実行します（指定されている場合）。
7. 最終的な検索の結果セットが返されます。

上のいずれかの手順でエラーコードが生成された場合、コマンドの実行は停止し、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コマンドがあるとします。addSortRule() 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。

FileMaker レイアウトの使用

レイアウトとは、ユーザがレコードをブラウズ、プレビュー、または印刷する時に、フィールド、オブジェクト、グラフィック、レイアウトパートなどがどのように配置されるかを決める情報です。FileMaker クラスは、FileMaker Pro のデータベースで定義されたレイアウトを使用可能にするためのいくつかのメソッドを定義します。レイアウトについての情報は、複数の FileMaker クラスオブジェクトから取得できます。

クラスオブジェクト	次のメソッドを使用
データベース	▪ listLayouts() は、利用可能なレイアウトの名前の一覧を取得します。 ▪ getLayout() は、レイアウト名を指定してレイアウトオブジェクトを取得します。
レイアウト	▪ getName() は、特定のレイアウトオブジェクトのレイアウト名を取得します。 ▪ listFields() は、レイアウト内で使用されるすべてのフィールド名の配列を取得します。 ▪ getFields() は、すべてのフィールドを配列のキーとして持ち、関連する FileMaker_Field オブジェクトを配列の値として持つ関連配列を取得します。 ▪ listValueLists() は、値一覧の配列を取得します。 ▪ listRelatedSets() は、関連セットの名前の配列を取得します。 ▪ getDatabase() は、データベース名を返します。
レコード	▪ getLayout() は、特定のレコードに関連するレイアウトオブジェクトを返します。
フィールド	▪ getLayout() は、特定のフィールドを含むレイアウトオブジェクトを返します。
コマンド	▪ setResultLayout() は、現在のレイアウトとは異なるレイアウトでコマンドの結果を返します。

ポータルの使用

ポータルとは、1 つ以上の関連レコードのデータ行を表示するテーブルです。FileMaker クラスは、FileMaker Pro のデータベースで定義されたポータルを使用可能にするための関連セットオブジェクト、およびいくつかのメソッドを定義します。

関連セットオブジェクトとは、関連ポータルのレコードオブジェクトの配列で、各レコードオブジェクトがポータル内の 1 データ行を表します。

特定のレイアウト上に定義されたポータルの一覧

特定のレイアウトオブジェクトには、listRelatedSets() メソッドを使用して、このレイアウト内で定義されたすべてのポータルのテーブル名の一覧を取得します。

例

```
$tableNames = $currentLayout->listRelatedSets();
```

特定の結果オブジェクト用のポータル名の取得

特定の FileMaker_Result オブジェクトには、getRelatedSets() メソッドを使用して、このレコード内のすべてのポータルの名前を取得します。

例

```
$relatedSetsNames = $result->getRelatedSets();
```

特定レイアウト用のポータルの情報の取得

特定のレイアウトオブジェクトには、getRelatedSets() メソッドを使用して、レイアウト内のポータルについて記述する FileMaker_RelatedSet オブジェクトの配列を取得します。返された配列は、テーブル名を配列のキーとして持ち、関連する FileMaker_RelatedSet オブジェクトを配列の値として持つ関連配列です。

例

```
$relatedSetsArray = $currentLayout->getRelatedSets();
```

特定ポータルの情報の取得

特定のレイアウトオブジェクトには、getRelatedSet() メソッドを使用して、特定のポータルについて記述する FileMaker_RelatedSet オブジェクトを取得します。

例

```
$relatedSet = $currentLayout->getRelatedSet('customers');
```

ポータルのテーブル名の取得

関連セットオブジェクトには、getName() メソッドを使用してポータル用のテーブル名を取得します。

例

```
$tableName = $relatedSet->getName();
```

特定レコード用のポータルレコードの取得

特定のレコードオブジェクトには、getRelatedSet() メソッドを使用して、そのレコードに関する特定のポータル用の関連レコードの配列を取得します。

例

```
$relatedRecordsArray = $currentRecord->getRelatedSet('customers');
```

ポータル内で新規レコードを作成

newRelatedRecord() メソッドを使用して、特定の関連セット内で新規レコードを作成し、commit() メソッドを呼び出して、変更をデータベースに確定します。

例

```
//create a new portal row in the 'customer' portal
$new_row = $currentRecord->newRelatedRecord('customer');

//set the field values in the new portal row
$new_row->setField('customer::name', $newName);
$new_row->setField('customer::company', $newCompany);

$result = $new_row->commit();
```

ポータルからレコードを削除

delete() メソッドを使用して、ポータル内のレコードを削除します。

例

```
$relatedSet = $currentRecord->getRelatedSet('customers');
/* Runs through each of the portal rows */
foreach ($relatedSet as $nextRow) {

    $nameField = $nextRow->getField('customer::name')
    if ($nameField == $badName ) {
        $result =   $newRow->delete();
    }
}
```

値一覧の使用

値一覧とは、事前定義された選択値のセットです。FileMaker クラスは、FileMaker Pro のデータベースで定義された値一覧を使用可能にするためのいくつかのメソッドを定義します。

特定レイアウト用のすべての値一覧名の取得

特定のレイアウトオブジェクトには、listValueLists() メソッドを使用して、値一覧名を含む配列を取得します。

例

```
$valueListNames = $currentLayout->listValueLists();
```

特定レイアウト用のすべての値一覧の配列の取得

特定のレイアウトオブジェクトには、getValueListsTwoFields() メソッドを使用して、すべての値一覧からの値を含む配列を取得します。返される配列は関連配列です。配列のキーは値一覧名で、配列の値は表示名およびそれぞれの値一覧からの選択値の一覧である関連配列です。

例

```
$valueListsArray = $currentLayout->getValueListsTwoFields();
```

メモ getValueLists() メソッドは、現在 FileMaker API for PHP で引き続き使用できますが、推奨されません。代わりに getValueListsTwoFields() メソッドの使用を推奨しています。

名前付きの値一覧の値の取得

特定のレイアウトオブジェクトには、getValueListTwoFields() メソッドを使用して、名前付きの値一覧向けに定義された選択値の配列を取得します。返された配列は関連配列で、キーである値一覧の 2 番目のフィールドと、配列の値である最初のフィールドの関連格納値からの表示値を含みます。

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 getValueListTwoFields() メソッドは、最初のフィールドの値のみ、 2 番目のフィールドの値のみ、または値一覧の両方のフィールドの値の何れかを格納または表示された値として返します。

- [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、getValueListTwoFields() メソッドは、格納ならびに表示された値として値一覧の最初のフィールドから値を返します。
- [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、getValueListTwoFields() メソッドは、格納された値として最初のフィールドから値を、表示された値として 2 番目のフィールドから値を返します。
- [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、getValueListTwoFields() メソッドは、格納された値として最初のフィールドから値を、表示された値として最初と 2 番目の両方のフィールドから値を返します。

getValueListTwoFields() メソッドでイテレータを使用して表示ならびに格納された値を検索します。

例

```
$layout = $fm->getLayout('customers');
$valuearray = $layout->getValueListTwoFields("region", 4);
foreach ($valuearray as $displayValue => $value) {
        ....
}
```

メモ

- getValueList() メソッドは、現在 FileMaker API for PHP で引き続き使用できますが推奨されません。代わりに getValueListTwoFields() メソッドの使用を推奨しています。
- getValueListTwoFields() メソッドを使用する場合、必ず foreach loop を使用して関連配列を扱います。for loop では予想外の結果が返りますので使用しないでください。

検索条件の実行

FileMaker クラスは、 4 種類の検索コマンドオブジェクトを定義します。

- Find All コマンド。37 ページの「Find All コマンドの使用」を参照してください。
- Find Any コマンド。37 ページの「Find Any コマンドの使用」を参照してください。
- Find コマンド。38 ページの「Find コマンドの使用」を参照してください。
- Compound Find コマンド。38 ページの「Compound Find コマンドの使用」を参照してください。

また、FileMaker クラスは、 4 種類すべての検索コマンドに使用できるメソッドをいくつか定義します。

- addSortRule() メソッドを使用して、結果セットのソート方法を定義するルールを追加します。clearSortRules() メソッドを使用して、定義されたソートルールすべてをクリアします。
- setLogicalOperator() メソッドを使用して、論理積による検索から論理和による検索に切り替えます。
- setRange() メソッドを使用して、結果セットの一部のみをリクエストします。getRange() メソッドを使用して、現在の範囲定義を取得します。

  setRange() メソッドを使用すると、検索条件によって返されるレコードの数が減るので、ソリューションのパフォーマンスが向上します。たとえば、検索条件が 100 レコードを返す場合、100 レコードを一度に処理する代わりに、結果セットを 20 レコードずつの 5 グループに分けることができます。
- 検索コマンドと共に FileMaker スクリプトを実行できます。
  - 検索コマンドを実行する前にスクリプトを実行するには、setPreCommandScript() メソッドを使用します。
  - 結果セットをソートする前にスクリプトを実行するには、setPreSortScript() メソッドを使用します。
  - 結果セットの生成後のソート前にスクリプトを実行するには、setScript() メソッドを使用します。

Find All コマンドの使用

Find All コマンドを使用して、特定のレイアウトからすべてのレコードを取得します。newFindAllCommand() メソッドを使用し、特定のレイアウトを指定して、FileMaker_Command_FindAll オブジェクトを作成します。その後、execute() メソッドを呼び出し、検索条件を実行します。

例

```
$findCommand =& $fm->newFindAllCommand('Form View');
$result = $findCommand->execute;
```

メモ Find All コマンドを使用する場合、 1 ページにつき返すデフォルトの最大レコード数を指定することでコンピュータメモリの過負荷問題を回避します。

Find Any コマンドの使用

Find Any コマンドを使用して、特定のレイアウトからレコードをランダムに 1 つ取得します。newFindAnyCommand() メソッドを使用し、特定のレイアウトを指定して、FileMaker_Command_FindAny オブジェクトを作成します。その後、execute() メソッドを呼び出し、検索条件を実行します。

例

```
$findCommand =& $fm->newFindAnyCommand('Form View');
$result = $findCommand->execute;
```

Find コマンドの使用

newFindCommand() メソッドを使用し、特定のレイアウトを指定して、FileMaker_Command_Find オブジェクトを作成します。その後、execute() メソッドを呼び出し、検索条件を実行します。

addFindCriterion() メソッドを使用して、検索条件に基準を追加します。clearFindCriteria() メソッドを使用して、定義済みのすべての検索条件をクリアします。

例 - フィールド名でレコードを検索

```
$findCommand =& $fm->newFindCommand('Form View');
$findCommand->addFindCriterion('Questionnaire ID', $active_questionnaire_id);
$result = $findCommand->execute();
```

例 - ソート順序を追加

```
$findCommand =& $fm->newFindCommand('Customer List');
$findCommand->addSortRule('Title', 1, FILEMAKER_SORT_ASCEND);
$result = $findCommand->execute();
```

Compound Find コマンドの使用

Compound Find コマンドを使用すると、複数の検索条件オブジェクトを 1 つのコマンドにまとめることができます。

Compound Find コマンドの作成 :

- newCompoundFindCommand() メソッドを呼び出して、FileMaker_Command_CompoundFind オブジェクトを作成します。
- newFindRequest() メソッドを使用して、 1 つ以上の FileMaker_Command_FindRequest オブジェクトを作成します。
- setOmit() メソッドを使用して、最終的な結果セットから除外される特定の検索条件の結果セット内のレコードを示します。
- add() メソッドを使用して、Compound Find コマンドオブジェクトへ検索条件オブジェクトを追加します。
- execute() メソッドを呼び出して、Compound Find コマンドを実行します。

例 - Compound Find コマンド

```
// Create the Compound Find command object
$compoundFind = $fm->newCompoundFindCommand('Form View');

// Create first find request
$findreq1 = $fm->newFindRequest('Form View');

// Create second find request
$findreq2 = $fm->newFindRequest('Form View');

// Create third find request
$findreq3 = $fm->newFindRequest('Form View');

// Specify search criterion for first find request
$findreq1->addFindCriterion('Quantity in Stock', '<100');

// Specify search criterion for second find request
$findreq2->addFindCriterion('Quantity in Stock', '0');
$findreq2->setOmit(true);

// Specify search criterion for third find request
$findreq3->addFindCriterion('Cover Photo Credit', 'The London Morning News');
$findreq3->setOmit(true);

// Add find requests to compound find command
$compoundFind->add(1,$findreq1);
$compoundFind->add(2,$findreq2);
$compoundFind->add(3,$findreq3);

// Set sort order
$compoundFind->addSortRule('Title', 1, FILEMAKER_SORT_DESCEND);

// Execute compound find command
$result = $compoundFind->execute();

// Get records from found set
$records = $result->getRecords();

// Print number of records found
echo 'Found '. count($records) ."results.<br><br>";
```

結果セット内のレコードの処理

- getRecords() メソッドを呼び出して、結果セット内の各レコードを含む配列を取得します。配列の各メンバーは、FileMaker_Record オブジェクトか、レコードのインスタンスを作成するための API 内のクラス名セットのインスタンスです。結果セットにレコードが含まれない場合、配列は空の可能性があります。
- getFields() メソッドを呼び出して、結果セット内のすべてのフィールドの名前の一覧を取得します。メソッドはフィールド名のみ返します。フィールドに関する追加情報が必要な場合は、関連するレイアウトオブジェクトを使用します。
- getFoundSetCount() メソッドを呼び出して、対象レコード全体のレコード数を取得します。
- getFetchCount() メソッドを呼び出して、フィルタ済みの対象セットのレコード数を取得します。検索コマンド上で範囲のパラメータを指定しない場合、この値は getFoundSetCount() メソッドの結果と同じになります。これは常に count($response->getRecords()) の値と等しくなります。
- 特定のレコードには、フィールドの内容を文字列として返す getField() メソッドを使用します。
- 特定のレコードには、フィールドの内容を Unix のタイムスタンプ（日付の PHP 内部表現）として返す getFieldAsTimestamp() メソッドを使用します。
  - フィールドが日付フィールドの場合、タイムスタンプは、午前零時のフィールド日付を表します。
  - フィールドが時刻フィールドの場合、タイムスタンプは 1970 年 1 月 1 日のその時刻を表します。
  - フィールドがタイムスタンプフィールドの場合、FileMaker タイムスタンプ値が Unix のタイムスタンプに直接マップされます。
  - 指定されたフィールドが日付または時刻フィールドではない場合、または生成されたタイムスタンプが範囲外である場合は、getFieldAsTimestamp() メソッドは FileMaker_Error オブジェクトを返します。
- 特定のレコードには、バイナリデータとしてオブジェクトフィールドのオブジェクトを返す getContainerData() メソッドを使用します。

  ```
  <IMG src=”img.php?-url=<?php echo urlencode($record->getField('Cover Image')); ?>”>

  echo $fm->getContainerData($_GET['-url']);
  ```

- 特定のレコードには、オブジェクトフィールドのオブジェクトに完全修飾 URL を返す getContainerDataURL() メソッドを使用します。

  ```
  // For images, use the HTML img tag
        echo '<img src="'.$fm->
  getContainerDataURL($record->getField('container')) .'">';
  // For movies and PDF files, use the HTML embed tag
        //echo '<embed src="'.$fm->
  getContainerDataURL($record->getField('container')) .'">';
  ```

検索条件によって返されたポータルの行のフィルタリング

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の数を制限するには、検索条件と共に setRelatedSetsFilters() メソッドを使用します。setRelatedSetsFilters() メソッドには次の 2 つの引数を指定できます。

- 関連セットのフィルタ値 : layout または none。
  - 値 none を指定する場合、Web 公開エンジンによって、ポータル内のすべての行が返され、ポータルレコードは事前にソートされません。
  - 値 layout を指定する場合、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。

- 返されるポータルレコードの最大数 : 整数値または all。
  - この値は、[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] の設定が有効化されている場合のみ使用されます。整数値を指定する場合、最初の行より後の行の数が返されます。all を指定する場合、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。
  - [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、[ポータル設定] ダイアログボックスの [行数] 設定によって、返される関連レコードの最大数が決定されます。

コマンド、レコード、およびフィールドの妥当性の事前チェック

FileMaker クラスを使用すると、データをデータベースに確定する前に、Web サーバー上の PHP ソリューションのフィールド データの妥当性を事前にチェックすることができます。

妥当性の事前チェックを使用するかどうかを決定する際には、Web ユーザが入力しているデータの値の数を考慮します。ユーザが更新するフィールド数が少ない場合は、妥当性の事前チェックを行わないことでパフォーマンスを向上させることができます。ただし、ユーザがたくさんのフィールドにデータを入力する場合は、妥当性の事前チェックを行うことで、レコードが妥当性チェックエラーのためにデータベースによって拒否されることによる不便さから解放されます。

FileMaker クラスを使用する場合、PHP エンジンは次のフィールドの制約の妥当性を事前にチェックします。

- 空でない

  有効なデータは、空でない文字列です。データは少なくとも 1 文字含んでいる必要があります。

- 数字のみ

  有効なデータには、数字のみが含まれます。

- 最大文字数

  有効なデータには、多くとも指定された最大文字数しか含まれません。

- ４桁の西暦の日付

  有効なデータは、M/D/YYYY の形式で 4 桁の西暦の日付を表す文字列です。ここで、M は 1 から 12 までの数で、D は 1 から 31 までの数で、YYYY は 0001 から 4000 までの間の 4 桁の数です。たとえば、1/30/3030 は、有効な４桁の西暦の日付の値です。ただし、4/31/2013 は、４月には 31 日目がないので、無効な 4 桁の西暦の日付の値です。日付の妥当性チェックでは、フォワードスラッシュ（/）、バックスラッシュ（\）、およびハイフン（-）を区切り文字としてサポートします。ただし、文字列には区切り文字を混ぜ合わせて使用することはできません。たとえば、1\30-2013 などは無効です。

- 時刻

  有効なデータは、次のフォーマットの 1 つを使用して 12 時間分の時間の値を表示する文字列です。

  - H
  - H:M
  - H:M:S
  - H:M:S AM/PM
  - H:M AM/PM

  ここで、H は、 1 から 12 までの数で、M および S は 1 から 60 までの数です。

PHP エンジンの妥当性の事前チェックでは、フィールドの種類に基づいてフィールドデータを暗黙にチェックする機能をサポートしています。

- 日付

  日付フィールドとして定義されているフィールドは、年の値には 0 ～ 4 桁の値を含めることができるという点（年の値は空でもよい）を除いて、「4 桁の西暦」の妥当性チェックに従ってチェックされます。たとえば、1/30 は、年が指定されていませんが、有効な日付です。

- 時刻

  時刻フィールドとして定義されているフィールドは、時（H）を表す部分は 24 時間の値をサポートするために 1 から 24 までの数字にすることができるという点を除いて、「時刻」の妥当性チェックに従ってチェックされます。

- タイムスタンプ

  タイムスタンプフィールドとして定義されているフィールドは、時刻の部分は「時刻」の妥当性チェックに従ってチェックされ、日付の部分は「日付」の妥当性チェックに従ってチェックされます。

FileMaker クラスでは、FileMaker Pro で利用可能なフィールドの妥当性チェックオプションすべての妥当性を事前にチェックすることはできません。次の妥当性チェックオプションは、データが確定される際にデータベースに存在するすべてのデータの状態に依存しているので、妥当性は事前にチェックできません。

- ユニークな値
- 既存値
- 下限値
- 値一覧名
- 計算式での妥当性チェック

コマンド内のレコードの妥当性の事前チェック

コマンドオブジェクトには、validate() メソッドを使用して、PHP エンジンが強制できる妥当性チェックのルールに対して 1 つのフィールドまたはコマンド全体をチェックします。オプションのフィールド名の引数を渡した場合、そのフィールドのみの妥当性が事前にチェックされます。

妥当性の事前チェックをパスした場合、validate() メソッドは TRUE を返します。妥当性の事前チェックをパスできなかった場合、validate() メソッドは、何が妥当性チェックをパスできなかったかについての詳細を含む FileMaker_Error _Validation オブジェクトを返します。

レコードの妥当性の事前チェック

レコードオブジェクトには、validate() メソッドを使用して、PHP エンジンが強制できる妥当性の事前チェックのルールに対して 1 つのフィールドまたはレコード内のすべてのフィールドをチェックします。オプションのフィールド名の引数を渡した場合、そのフィールドのみの妥当性が事前にチェックされます。

妥当性の事前チェックをパスした場合、validate() メソッドは TRUE を返します。妥当性の事前チェックをパスできなかった場合、validate() メソッドは、何が妥当性チェックをパスできなかったかについての詳細を含む FileMaker_Error _Validation オブジェクトを返します。

フィールドの妥当性の事前チェック

フィールドオブジェクトには、validate() メソッドを使用して、与えられた値がフィールドに対して妥当かどうかを決定します。

妥当性の事前チェックをパスした場合、validate() メソッドは TRUE を返します。妥当性の事前チェックをパスできなかった場合、validate() メソッドは、何が妥当性チェックをパスできなかったかについての詳細を含む FileMaker_Error _Validation オブジェクトを返します。

妥当性チェックエラーの処理

妥当性の事前チェックが失敗すると、返される FileMaker_Error_Validation オブジェクトには妥当性チェックの失敗ごとに 3 つの要素の配列が含まれます。

1. 妥当性の事前確認に失敗したフィールドオブジェクト
2. 失敗した妥当性チェックルールを示す、妥当性チェックの定数値

   1 - FILEMAKER_RULE_NOTEMPTY

   2 - FILEMAKER_RULE_NUMERICONLY

   3 - FILEMAKER_RULE_MAXCHARACTERS

   4 - FILEMAKER_RULE_FOURDIGITYEAR

   5 - FILEMAKER_RULE_TIMEOFDAY

   6 - FILEMAKER_RULE_TIMESTAMP_FIELD

   7 - FILEMAKER_RULE_DATE_FIELD

   8 - FILEMAKER_RULE_TIME_FIELD
3. 妥当性の事前チェックに失敗したフィールドに入力された実際の値

次のオブジェクトを FileMaker_Error_Validation オブジェクトと共に使用することもできます。

- isValidationError() メソッドを使用して、エラーが妥当性チェックエラーかどうかをテストします。
- numErrors() メソッドを使用して、失敗した妥当性チェックルールの数を取得します。

例

```
//Create an Add request
$addrequest =& $fm->newAddCommand('test', array('join' => 'added', 'maxchars' => 'abcx', 'field' => 'something' , 'numericonly' => 'abc'));

//Validate all fields
$result = $addrequest->validate();

//If the validate() method returned any errors, print the name of the field, the error number, and the value that failed.
if (FileMaker::isError($result)) {
    echo 'Validation failed:'."￥n";
    $validationErrors= $result->getErrors();
    foreach ($validationErrors as $error) {
        $field =  $error[0];
        echo 'Field Name:' .$field->getName()."￥n";
        echo 'Error Code:' .$error[1] ."￥n";
        echo 'Value:' .$error[2] ."￥n";
    }
}
```

出力

```
Validation failed:
Field Name:numericonly
Error Code:2
Value:abc
Field Name:maxchars
Error Code:3
Value:abcx
```

エラー処理

FileMaker クラスは、PHP ソリューション内で発生するエラーを処理するのに役立つ FileMaker_Error オブジェクトを定義します。

コマンドを実行するとエラーが発生する可能性があります。コマンドを実行するとエラーが発生する可能性があります。コマンドが実行した際に返されるエラーは、毎回チェックするようにしてください。

次のメソッドを使用して、FileMaker_Error オブジェクトの中で示されるエラーの詳細を調べます。

- isError() メソッドを呼び出して、変数が FileMaker Error オブジェクトかどうかテストします。
- numErrors() メソッドを呼び出して、発生したエラーの数を取得します。
- getErrors() メソッドを呼び出して、発生したエラーを説明する配列の中から配列を 1 つ取得します。
- getMessage() メソッドを呼び出して、エラーメッセージを表示します。

例

```
$result = $findCommand->execute();
if (FileMaker::isError($result)) {
        echo "<p>Error:" ." . $result->getMessage() .""</p>";
        exit;
}
```

FileMaker Error オブジェクトと共に返されるエラーコードについては、付録 A「カスタム Web 公開 with PHP のエラーコード」を参照してください。

第 6 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」にあるすべての手順を行います。
2. カスタム Web 公開 with PHP が FileMaker Server Admin Console で有効化され、正しく構成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. 展開した FileMaker Server の Web サーバーコンポーネントにサイトのファイルをコピーまたは移動します。

   Web サーバー マシン上の次のディレクトリにサイトファイルをコピーまたは移動します。

   - HTTP または HTTPS 経由の IIS（Windows）: [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥HTTPServer￥Conf
     [ドライブ] は、展開した FileMaker Server の Web 公開エンジンコンポーネントが格納されているドライブです。
   - HTTP 経由の Apache（OS X）: / ライブラリ /FileMaker Server/HTTPServer/htdocs
   - HTTPS 経由の Apache（OS X）: / ライブラリ /FileMaker Server/HTTPServer/htdocs/httpsRoot

5. まだ操作を行っていない場合は、参照先オブジェクトを Web サーバーマシン上の適切な場所にコピーまたは移動します。
   - 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。
   - 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、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。
   - FileMaker Pro を使用してデータベースをアップロードする場合、外部に保存されたオブジェクトフィールドデータは、プロセスの一環として FileMaker Server にアップロードされます。FileMaker Server へのデータの転送については、「FileMaker Pro ヘルプ」を参照してください。
   - オブジェクトを外部に保存したオブジェクトフィールドを使用するデータベースを手動でアップロードするには、19 ページの「外部に保存されたデータを含むオブジェクトフィールド」の説明のとおりに「RC_Data_FMS」フォルダのサブフォルダに参照されているオブジェクトをコピーするか移動する必要があります。

   18 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」を参照してください。

6. サイトのテストを開始します。

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、22 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。
- FileMaker API for PHP を使用してソリューションを作成する場合は、クッキーのサポートを有効化してソリューションを構築することをお勧めします。FileMaker API for PHP の応答時間は、クッキーを有効にすると向上します。カスタム Web 公開の機能を使用するためにクッキーは必要ありませんが、クッキーを有効にすると Web 公開エンジンがセッション情報をキャッシュできます。

メモ 1 台のマシンに Web サーバー、Web 公開エンジン、データベースサーバーをインストールしている場合、ネットワークに接続せずにサイトを表示およびテストできます。コンピュータ上の適切なディレクトリにファイルを移動して次の URL をブラウザに入力します。

http://127.0.0.1/< サイトパス >

< サイトパス > は使用しているサイトのホームページへの相対パスです。

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 公開エンジンログ
- Web サーバーモジュールのエラーログ
- Tomcat ログ

Web サーバーのアクセスログとエラーログの使用

- 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。
- Apache（OS X のみ）: 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。

メモ 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 公開エンジンのログの使用

デフォルトでは、Web 公開エンジンは、アプリケーションエラー、使用状況エラー、システムエラーを含むあらゆる Web 公開エンジンエラーが含まれる wpe.log というログファイルを生成します。Web 公開出力を生成するエンドユーザ XML リクエストや、カスタム Web 公開設定の変更など、カスタム Web 公開に関連する情報を含めるよう Web 公開エンジンを設定することもできます。

FileMaker API for PHP では、HTTP POST を使用して Web 公開エンジンにアクセスしているため、wpe.log ログファイルには PHP リクエストの詳細は記録されません。wpe.log ログファイルを使用して記録された XML リクエストを参照し、いつユーザが PHP リクエストを行ったかを確認することができます。

wpe.log ファイルは、展開した FileMaker Server の Web 公開エンジンコンポーネントに格納されます。

- IIS（Windows）: [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥HTTPServer￥Logs￥wpe.log
  [ドライブ] は、システムが起動されるプライマリドライブです。
- Apache（OS X）: / ライブラリ /FileMaker Server/HTTPServer/Logs/wpe.log

Web 公開エンジンログの設定

wpe.log ファイルは、[Web 公開用のログを有効にする] オプションが Admin Console で有効になっている場合に生成されます。

有効になっているログオプション	wpe.log に記録される情報
エラーレベルメッセージ	アプリケーションエラー、使用状況エラーおよびシステムエラーを含む、あらゆる Web 公開エンジンエラー。
情報およびエラーレベルメッセージ	上記のエラーと Web 公開エンジンへのアクセスに関する情報。カスタム Web 公開出力を生成するすべてのエンドユーザ XML リクエストの記録を含みます。

[エラーレベルメッセージ] の設定はデフォルトで有効化されます。Admin Console を使用したこれらのオプション設定の詳細については、「FileMaker Server ヘルプ」を参照してください。

重要　長期間にわたると、wpe.log ファイルのサイズが大きくなることがあります。Admin Console を使用して wpe.log ファイルの最大サイズを設定してください。wpe.log ファイルが最大サイズに達すると、Web 公開エンジンは wpe.log ファイルを 1 つのバックアップファイル wpe.log.1 にコピーし、新しい wpe.log ファイルを作成します。バックアップを複数保存するには、wpe.log.1 ファイルのアーカイブを定期的に保存してください。

Web 公開エンジンログの形式

wpe.log ファイルは各エントリに以下の形式を使用します。

[TIMESTAMP_GMT] [WPC_HOSTNAME] [CLIENT_IP:PORT] [ACCOUNT_NAME] [MODULE_TYPE] [SEVERITY] [FM_ERRORCODE] [RETURN_BYTES] [MESSAGE]

各要素の意味は次のとおりです。

- [TIMESTAMP_GMT] は、グリニッジ標準時（GMT）のエントリの日付と時刻です。
- [WPC_HOSTNAME] は、Web 公開エンジンがインストールされているコンピュータのコンピュータ名です。
- [CLIENT_IP:PORT] は、XML リクエストが生成されたクライアントの IP アドレスとポートです。
- [ACCOUNT_NAME] は、ホストされている FileMaker データベースにログインするために使用されるアカウント名です。
- [MODULE_TYPE] は、カスタム Web 公開 with XML のリクエストに使用する XML、またはカスタム Web 公開 with PHP のリクエストに使用する PHP です。
- [SEVERITY] は、情報メッセージを示す INFO、またはエラーメッセージを示す ERROR です。

- [FM_ERROR_CODE] は、エラーメッセージを返すエラー番号です。エラー番号は、FileMaker データベースのエラーコードの場合があります（51 ページの「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）。

  また、エラー番号は先頭に文字列「HTTP:」の付いた HTTP エラー番号の場合もあります。
- [RETURN_BYTES] は、リクエストによって返されたバイト数です。
- [MESSAGE] は、ログエントリに関する追加情報を提供します。.

Web 公開エンジンログメッセージの例

次の例は、wpe.log ファイルに含まれるメッセージの種類を示しています。

- Web 公開エンジンの起動および停止

  2013-06-02 15:15:31 -0700 - - - - INFO - - FileMaker Server Web Publishing Engine started.

  2013-06-02 15:46:52 -0700 - - - - INFO - - FileMaker Server Web Publishing Engine stopped.
- XML クエリリクエストの成功または失敗

  2013-06-02 15:21:08 -0700 WPC_SERVER 192.168.100.101:0 jdoe XML INFO 0 3964 "/fmi/xml/fmresultset.xml?-db=Contacts&-lay=Contact_Details&-findall"

  2013-06-02 15:26:31 -0700 WPC_SERVER 192.168.100.101:0 jdoe XML ERROR 5 596 "/fmi/xml/fmresultset.xml?-db=Contacts&-layout=Contact_Details&-findall"
- スクリプトエラー

  2013-06-02 17:33:12 -0700 WPC_SERVER 192.168.100.101:0 jdoe - ERROR 4 - Web Scripting Error:4, File:"10b_MeetingsUpload", Script:"OnOpen", Script Step:"Show Custom Dialog"
- カスタム Web 公開設定の変更

  2013-06-09 10:59:49 -0700 WPC_SERVER 192.168.100.101:0 jdoe - INFO - - XML Web Publishing Engine is enabled.
- システムエラー

  2013-06-02 15:30:42 -0700 WPC_SERVER 192.168.100.101:0 jdoe XML ERROR - - Communication failed

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。このログファイルは「web_server_module_log.txt」という名前で、Web サーバーホスト上の「FileMaker Server」フォルダ内の「Logs」フォルダに格納されます。

- IIS（Windows）: [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥Logs￥web_server_module_log.txt [ドライブ] は、システムが起動されるプライマリドライブです。
- Apache（OS X）: / ライブラリ /FileMaker Server/Logs/web_server_module_log.txt

Tomcat ログの使用

内部 Web サーバーエラーが原因で FileMaker Server に問題が発生した場合は、Tomcat ログを参照することをお勧めします。Tomcat ログは、FileMaker Server 展開の Web サーバーコンポーネントに格納されます。

- IIS（Windows）: [ドライブ]:￥Program Files￥FileMaker￥FileMaker Server￥Admin￥admin-master-tomcat￥logs￥ [ドライブ] は、システムが起動されるプライマリドライブです。
- Apache（OS X）: / ライブラリ /FileMaker Server/Admin/admin-master-tomcat/logs/

サイトのトラブルシューティング

サイトの表示または使用に問題がある場合は、次の点を確認してください。

- カスタム Web 公開 with PHP 用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。15 ページの「データベースのカスタム Web 公開 with PHP の有効化」を参照してください。
- データベースが FileMaker Server によってホストされて開かれている。「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。
- Web サーバーおよび Web 公開エンジンが実行されている。
- Web 公開エンジンで PHP 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。
  - [PHP 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 PHP 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。

付録 A
カスタム Web 公開 with PHP 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 2 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データリクエストのエラー。データが公開されているデータベースからリクエストされると、常に、Web 公開エンジンによってエラーコードが生成されます。FileMaker API for PHP は、このエラーコードを FileMaker_Error オブジェクトとして返します。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。
- PHP エラー。これらのエラーは、cURL モジュールを含む、PHP コンポーネントによって生成され返されます。58 ページの「PHP コンポーネントのエラーコード番号」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

返されたエラーコード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
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

エラー番号	説明
21	リクエストがサポートされていません（ハードリンクに対応していないファイルシステムにハードリンクを作成しようとした場合など）
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
118	テーマが見つかりません
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（Starter Solutions 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
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

エラー番号	説明
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
218	パスワードとパスワードの確認が一致しません
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
307	ホストとの通信エラーのためトランザクションをロックできませんでした
308	別のユーザがテーマを使用しておりロックされています
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	ソート順が無効です
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の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
418	フィールドへの INSERT 操作中に内部エラーが発生しました
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
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フィールドサイズを超えています

エラー番号	説明
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	検証は指定されていませんが、フィールドにデータが一致しません
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モニタが必要です
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/XSL 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 2007/2008 以降のサポートされているバージョンに変換して、もう一度実行してください）
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。入らない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

エラー番号	説明
738	インポートしようとしているテーマはすでに存在します
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
812	ホストの容量を超えています
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
826	指定したファイルパスは有効なパスではありません
850	パスがオペレーティングシステムで有効ではありません
851	ディスクから外部ファイルを削除できません
852	外部格納にファイルを書き込めません
853	1 つ以上のオブジェクトの転送に失敗しました
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
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

エラー番号	説明
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
960	引数が無効です
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、テキスト定数、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	テキスト定数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、「(」は使用できません
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	この引数は取得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
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

エラー番号	説明
1300	指定された名前は使用できません
1301	インポートまたは貼り付けられた関数の引数の 1 つが、ファイルにすでにある関数と同じ名前です
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）
1414	SQL ステートメントが長すぎます
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 メールメッセージ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
1550	プラグインをロードできないか、プラグインが有効なプラグインではありません
1551	プラグインをインストールできません。既存のプラグインを削除できないか、フォルダまたはディスクに書き込めません
1626	プロトコルがサポートされていません
1627	認証に失敗しました
1628	SSL でエラーが発生しました
1629	接続がタイムアウトになりました。タイムアウトの値は 60 秒です
1630	URL 書式が正しくありません
1631	接続に失敗しました

PHP コンポーネントのエラーコード番号

FileMaker API for PHP では、いくつかの PHP コンポーネントを使用しています。これらの PHP コンポーネントは、上記に一覧表示されていない追加のエラーコードを返す可能性があります。

たとえば、Web Publishing Core または FileMaker Server サービスが実行されていない場合、cURL モジュールエラーである CURLE_GOT_NOTHING (52) を受け取る可能性があります。

PHP 関連のエラーコードについては、PHP の Web サイト http://php.net を参照してください。

索引

N

O

P

Q

S

T

U

V

W

X

せ

た

て

と

は

ひ

ふ

ほ

ま

め

ゆ

よ

り

れ

ろ