Cardiff TeleForm® v10.1

VBA ガイド

― Ver.1.1 ―

この文書に含まれる情報は、公表された時点での株式会社ハンモックの考え方を表しています。株式会社ハンモックは、この文書で明示もしくは暗黙に示された内容を保証するものではありません。この文書は情報の提供のみを目的としています。記載されている内容は予告なく変更することがあります。

重要事項

- このガイドは、TeleForm の基本と VBA を理解されていることを前提条件として作成しております。
- BasicScript については、このガイドの提供と弊社ホームページの FAQ に掲載されているサンプルのみのご提供とさせて頂いており、個別のお問い合わせは**サポート範囲外**となります。
  開発を行われる方の責任において、BasicScript の組み込みを行って頂きますようお願い致します。

改版履歴

版	日付	内容
初版	2007 年 6 月 25 日	・ 新規作成
Ver1.1	2007 年 9 月 28 日	・ 重要事項の追記

目次

第1章　VBA の概要

この章では以下のことを学びます。

- VBA の利便性
- VBA の機能
- スクリプト タイプ
- エントリー ポイント
- TeleForm オブジェクト モデル

1-1. VBA とは

1-1-1. TeleForm でのスクリプトについて

TeleForm ではスクリプト言語として Microsoft の Visual Basic for Applications (VBA)が提供されています。*

スクリプトにより、帳票をどのようにインデックスを付け、解釈し、評価し、アクセスし、データを出力するかなどの機能を拡張することができます。

例えば、TeleForm VBA を利用することで次のような機能が実現できます：

フォーム評価時

- フォーム データの検査、修正、拡張
- フォーム再検査の必要性の決定
- フィールドやフォームの評価ステータスの変更

データ修正および入力時

- フィールドの不正データ入力時の強制リトライ
- 入力データに基づいた次フィールド動的選択（スキップ-アンド-フィル ロジック）
- フォーム モードに移行する前の不正チェック
- 特定の TeleForm *Verifier* オペレーターのみ閲覧可能な特定のフォーム情報の許可
- 特定の TeleForm *Verifier* オペレーターに帳票を送信
- エクスポート前のデータ整形

エクスポート時

- エクスポート前のデータ整合性の確認
- データ エクスポートの完全な制御
- カスタム操作によるフォーム処理

全般機能

- スクリプトでユーザー インターフェースを提供するカスタム ダイアログ ボックス作成
- TeleForm Reader での自動定期インターバル処理

1-1-2. TeleForm の VBA について

TeleForm の VBA は TeleForm オブジェクトモデルという特別なオブジェクトと同様に、Microsoft VBA Library（TeleForm VBA Script Editor のヘルプメニューから Microsoft VBA HELP をご覧ください）で使用できるオブジェクトやプロパティが使用できます。

TeleForm とのやり取りは TeleForm オブジェクトモデルを使います。

---

* 以前までは Basic Script Editor が提供されてきましたが、v10 より VBA が提供されるようになりました。v10 では設定を変更することで Basic Script Editor も利用できますが、Basic Script Editor は今後サポートされなくなる予定です。

TeleForm オブジェクトモデルは以下の構成です。

- オブジェクトモデルはいくつかのクラスを持ちます。各クラスは、TeleForm の特定の情報を持ちます。
- 各 TeleForm オブジェクトは唯一のプロパティを持ちます。各プロパティは TeleForm の情報のサブセットを参照します。プロパティは TeleForm の情報を完全に参照することができ、スクリプトのブロックを形成します。
- 各プロパティはデータ型で分類されます。データ型は、プロパティの値のタイプを知ることが出来ます。

VBA オブジェクトのように TeleForm オブジェクトは、オブジェクト名の後にドット演算子（．）を入力することで、プロパティやメソッドが自動的にドロップダウンリストに表示されます。

1-2. VBA の機能

1-2-1. プロジェクトモデル

VBA スクリプトはプロジェクトとして TeleForm に保存されます。プロジェクトは、そのプロジェクトのアイテムから形成されます。そしてプロジェクトのアイテムは、コード モジュール、クラス モジュール、とフォーム (ダイアログ テンプレート) から作られています。TeleForm オブジェクトはホストプロジェクトアイテムからのイベントに提供されます。すべてのプロジェクトのプロパティとアイテムは、一つのファイル (複合ドキュメント) に格納されます。

VBA はその Script Editor の一つのインスタンスを提供します。すべてのロードされたプロジェクトは VBA Script Editor に表示されます。システム プロジェクトは、TeleForm アプリケーションを開いたときにロードされ、表示されます。フォームやドキュメントを TeleForm Designer で開いたとき、そのプロジェクトが VBA Script Editor に表示されるでしょう。

VBA Script Editor はデバッガも兼ね揃えています。それゆえに、ロードされた全プロジェクトはデバッガに表示されます。スクリプトをデバックするには、プロジェクトにブレークポイントを設定するだけです。スクリプトを実行すると、ブレークポイントでスクリプトが停止します。ブレークポイントに加えて、VBA は『ステップ イン』、『ステップ アウト』や『ステップ オーバー』のような一般的なデバックや、変数を見る機能があります。

※ 注意 VBA Script Editor でスクリプトを編集するには、必ず TeleForm Designer から行ってください。他の TeleForm Application からでは、スクリプトを保存することはできません。

1-2-2. プロジェクト タイプ

TeleForm で使用できる VBA プロジェクト タイプは以下の 4 種類です。

- フォーム プロジェクト ― 全ての TeleForm フォームにつき一つのフォーム スクリプトが存在します。フォーム スクリプトは、フォーム マージ制御、データ不正確認、Verifier でのデータ コントロール、そしてエクスポート データの修正などに使用されます。フォーム スクリプトの利用頻度は非常に高いため、VBA を書く上でフォーム スクリプトを作成する機会が最も多いでしょう。
- システム プロジェクト ― 1 つの TeleForm システムにつきシステム スクリプトは 1 つだけ存在します。システム プロジェクトは以下の 4 つのプロジェクト アイテムを持ちます。
  - バッチ プロジェクト アイテム
  - グローバル フォーム プロジェクト アイテム
  - システム プロジェクト アイテム

・ ピリオディック プロジェクト アイテム

- エクスポート プロジェクト – エクスポート プロジェクトは、データのエクスポートに関する処理を記述します。複数のエクスポート プロジェクトを作成することが可能です。
- インプット プロジェクト ― インプット プロジェクトは、システム スクリプト外で定期的に処理を実行することができます。複数のインプット プロジェクトを作成することが可能です。
- ライブラリ プロジェクト ― 複数のライブラリ スクリプトを作成することが可能です。ライブラリ スクリプトはそれだけで直接実行することはできません。その代わり、関数を定義しておき、それらを上記のスクリプトから呼び出すことが可能です。例えば、複数のフォーム スクリプトで常に使用する関数がある場合、それをライブラリ スクリプトに格納し、各フォーム スクリプトから呼び出すことができます。

TeleForm プロジェクトは、以下の順番で呼び出されます。

1. ライブラリ
2. システム
3. インポート
4. エクスポート
5. フォーム

1-2-3. イベント

ライブラリ スクリプトを除く各スクリプトは、それぞれ固有のエントリー ポイントを持っています。これらのエントリー ポイントは、主要な TeleForm オペレーションを制御可能にする TeleForm 特有のサブルーチンと考えることができます。スクリプト内の適切なエントリー ポイントにコードを入力することで様々な TeleForm オペレーションをカスタマイズすることができます。

エントリー ポイントとは、TeleForm がスクリプトを実行する場所を指し示します。コードが入力されたエントリー ポイントは、いつ TeleForm がそのコードを実行するのかを表しています。言い換えるならば、各エントリー ポイントはフォーム処理サイクルでの一意のポイントを示しています。

エントリー ポイントと TeleForm 内のデータ フローとの関連性を理解すると、TeleForm が呼び出す適切なエントリー ポイントにコードを書くことができます。

下のダイアグラムは、各イベントのタイミングを示しています。

- 『Form_』で始まるイベントは、システム プロジェクトとフォーム プロジェクトのフォーム プロジェクト アイテムにあります。
- 『Batch_』で始まるイベントは、システム プロジェクトのバッチ プロジェクト アイテムに

あります。

- 『System_』で始まるイベントは、システム プロジェクトのシステム プロジェクト アイテムにあります。
- 『Periodic_』で始まるイベントは、システム プロジェクトのピリオディック プロジェクト アイテムにあります。
- 『Export_』で始まるイベントは、エクスポート プロジェクトにあります。
- 『Input_』で始まるイベントは、インプット プロジェクトにあります。

詳しいエントリー ポイント イベントのダイアグラムは第 5 章 付録をご覧ください。

例えば、TeleForm Reader でフォーム イメージを評価する時には次の一連のイベントが起こります。

1. TeleForm がフォーム評価を開始します。
2. TeleForm が Form_Evaluate エントリー ポイントを呼び出します。
3. このエントリー ポイントにスクリプトが存在している場合、スクリプトが実行されます。もし存在しなければ空のエントリー ポイントが実行されます。
4. TeleForm はフォーム処理を継続します。

エントリー ポイントは事前に定義されたサブルーチンとみなすことができます。エントリー ポイントにスクリプトが存在していようといまいと、TeleForm は対応する処理が実行されるタイミングで必ずエントリー ポイントを呼び出します。

1-2-4. オブジェクトとプロパティ

すべての TeleForm オブジェクトが TeleForm プロジェクト アイテムで使用できるわけではありません。

以下の表は、TeleForm オブジェクトと、使用できるプロジェクト アイテムの関係を示しています。

オブジェクト	説明
tfApplication	すべてのプロジェクト アイテムとイベントで使用できる最上位のオブジェクトで、実行中のアプリケーション情報を含みます。
tfBatch **	システム プロジェクトのバッチ プロジェクト アイテムの最上位オブジェクトです。バッチ情報を提供します。
tfChoice	tfChoices のコレクションを含み、tfChoices オブジェクトの Item プロパティにアクセスします。
tfChoices	tfChoice オブジェクトのコレクションです。tfField の Choices プロパティを通じてアクセスします。選択フィールドへのアクセスを提供します。
tfExport *	エクスポート プロジェクトのエクスポート プロジェクト アイテムの最上位オブジェクトです。現在のエクスポート セッション情報へのアクセスを提供します。
tfField	tfFields のコレクションで、tfFields オブジェクトの Item プロパティへアクセスします。フォームのフィールドのデータや他の属性へのアクセスを提供します。これはもっとも使用されるオブジェクトです。
tfFields	tfField オブジェクトのコレクションです。tfForm オブジェクトの Fields プロパティを通じてアクセスします。全フィールドへのアクセスを提供します。
tfForm	フォーム プロジェクトのフォーム プロジェクト アイテム、システム プロジェクトのグローバル フォーム プロジェクト アイテムの最上位オブジェクト、そしてエクスポート プロジェクトのエクスポート プロジェクト アイテムです。タイトルや ID などの一般的なフォームの情報を提供します。
tfRows	コレクションを含み、tfField オブジェクトの Rows プロパティでアクセスします。詳細グループのフィールドへアクセスします。詳細グループの各行は tfFields のコレクションです。
tfTopChar	tfTopChars コレクションを含み、tfTopChars オブジェクトの Item プロパティでアクセスします。
tfTopChars	tfTopChar オブジェクトのコレクションで、tfTopChoice オブジェクトの TopChars プロパティでアクセスします。

オブジェクト	説明
tfTopChoice	tfTopChoices のコレクションを含み、tfTopChoices オブジェクトの Item プロパティでアクセスします。
tfTopChoices	tfTopChoice オブジェクトのコレクションで、tfField オブジェクトの TopChoices プロパティでアクセスします。フィールド認識を基にした解釈された文字ごとに 3 つの最適推測文字へのアクセスを提供します。
tfUser	tfApplication オブジェクトの CurrentUser プロパティを通じてアクセスします。名前や権限のようなユーザ情報を含みます。

* … エクスポート プロジェクトのみで利用可能

** … バッチ プロジェクトのみで利用可能

第 5 章 付録の TeleForm VBA オブジェクト モデル ダイアグラムもご覧ください。

1-2-5. データ型

各プロパティは次のデータ型のいずれかに分類されます。

型	説明
Boolean	True か False を表示する 16 ビット整数型
Date	日時をあらわす 64 ビット浮動小数点数型
Double	倍精度浮動小数点数型
Integer	16 ビット整数型
Long	32 ビット整数型
Object	プロパティを持ったデータ型
String	文字列型
Variant	バリアント型

1-2-6. 大文字･小文字の区別

全てのオブジェクトやプロパティは大文字･小文字の区別をしません。コードを書く際は、これらを気にする必要はありません。

```
City.Text = CITY.TEXT = city.text = CiTy.tExT
```

しかしながら、大文字･小文字の区別は「アルファベット」と「アルファベットと数値」のフィールドの値を評価する際には非常に重要です。TeleForm のフィールドは大文字･小文字を解釈し格納するよう設定できるため、フィールドの値をテストするようなスクリプトでは注意が必要です。

例えば、「City」フィールドが ”San Francisco” という値を保持しているとします。この時に以下のようなスクリプトを書いたとします。

```
If City.Text = “SAN FRANCISCO”
```

この評価は失敗します。なぜならば ”San Francisco” は “SAN FRANCISCO” とマッチしないからです。

大文字･小文字の区別を排除した文字列比較を行う場合は、UCase や LCase を使用することができます。

```
If UCase (City.Text) = “SAN FRANCISCO”
```

この評価は真を返します。

■Tip ― このような問題を避ける最も適切な方法は、スクリプトで使用するフィールドに認識設定オプションで『大文字へ変換』チェックボックスを選択することです。もしこれを選択したならば、スクリプト内では大文字を使うようにしてください。

1-2-7. TeleForm VBA でのデバック

あなたは、現在実行中の Teleform アプリケーションのデバッグモードでプロジェクト アイテムを開くことによって、あなたの Teleform VBA プロジェクト アイテムで起こる実行時エラーをデバッグすることができます。

1. TeleForm アプリケーションの**ユーティリティ**メニューから**スクリプトのデバック**を選択します。
2. デバックのタイプを選択します。
   - フォーム スクリプトはフォーム スクリプト アイテムのデバックができます。
   - その他 はシステム、ライブラリ、エクスポート プロジェクト アイテムをデバックします。
3. スクリプトを実行するために、TeleForm のアプリケーションを実行します。
4. スクリプトは問題のある行で止まり、最前面に VBA Script Editor が現れます。

1-2-8. インタースクリプト依存

フォームやエクスポート プロジェクト アイテムがライブラリ プロジェクトから変数や関数を使うこと

は一般的でしょう。他のプロジェクトの変数や関数を使うときは、必ずプロジェクトに『参照設定』をする必要があります。

現在の VBA プロジェクトで参照設定のリストを表示するためには、VBA Script Editor の**ツール**から**参照設定**を選択します。そして、追加したい参照設定にチェックをつけます。

※ 注意　ライブラリ　スクリプトのみが参照設定可能です。他のスクリプトでの動作は未定義です。

1-3. VBA ツアー

VBA が TeleForm とどのように連携するのか、少しわかったところで、VBA のコーディングに関してクイック ツアーを行いましょう。このツアーでは VBA をどのようにコーディングし実装するのか、についてみていきます。

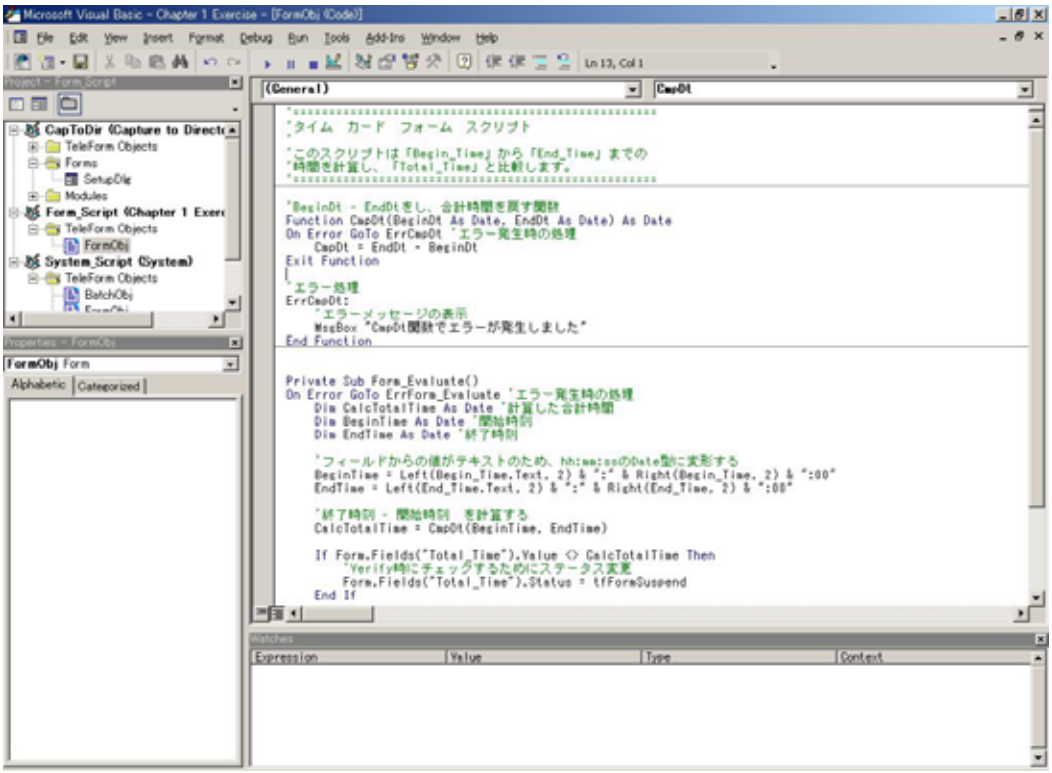
1-3-1. ステップ 1:スクリプト タイプを決定する

スクリプトを書く前に、どのタイプのスクリプトを書くかを決めなければいけません。スクリプトに処理させる TeleForm のプロセスを明確に見極めておく必要があります。また、エントリー ポイントについても考慮する必要があります。このツアーでは、もっともよく利用されるスクリプト タイプであるフォーム スクリプトを選択します。

1-3-2. ステップ 2:TeleForm Designer でスクリプトを開く

スクリプトを開くと「スクリプトの編集」ウィンドウが表示されます。「スクリプトの編集」ウィンドウはコードを書いたり、編集したり、コンパイルをするのに使用するアプリケーションです。このツアーでは「Chapter 1 Exercise」のフォーム スクリプトを開きます。

1. TeleForm Designer で「Chapter 1 Exercise」を開いてください。
   (「Chapter 1 Exercise」は [02_フォーム] の「Chapter 1 Exercise.tft」です。)
2. 「フォーム」メニューから「スクリプト」を選択します。「VBA Script Editor」が表示されます。



1-3-3. ステップ 3：スクリプトを書く

「スクリプトの編集」ウィンドウは、テキスト エディタで見られるような多くの基本機能とスクリプトのステートメントを記述するのに使用される VBA 特有の機能を持っています。

何もない状態からスクリプトを作成する場合は、次の慣例に従ってコーディングすることをお奨めします。

- 他の人にスクリプトの内容を伝えるため、そして自分のためのメモとしてスクリプトの先頭にコメントを書きます。
- 定数や Public 変数を宣言する際は、スクリプトの先頭（エントリー ポイントの前）で行います。
- 論理的に整理できるセクションをグループとして区別するために空白行を使用します。例えばこの開発者は、変数を 1 つのグループに、関数を別のグループにしています。
- 各グループに最低 1 行のコメントを書きます。これにより、他の人がスクリプトを見た時に、コードの内容が明確になります。また自分の理解を助ける手段にもなります。

1-3-4. ステップ 4：スクリプトをコンパイルする

スクリプトを作成し終えると、VBA ステートメントの文法をチェックするため、コンパイルする必要があります。文法に問題なければステップ 5 に進んでください。文法に問題があった場合は、エラーメッセージが表示されます。このエラー メッセージは問題のある行まで移動し、その問題が何なのか知らせてくれます。

重要：
コンパイルが OK だとしても、そのスクリプトの実行が成功するとは限りません。コンパイラーは全ての起こりえるエラーを検出することはできません。

1. 「VBA Script Editor」の「デバック」メニューから「コンパイル」を選択します。選択できない状態である場合、すでにコンパイル済みですので、ステップ 5 へ進んでください。
2. スクリプトにエラーがあると、エラーが発生した部分が選択状態になります。
3. 「ファイル」メニューから「保存」を選択します。

1-3-5. ステップ 5：スクリプトを実行する

スクリプトのコンパイルが OK ならば、それが期待どおりに動作するか実行します。このことはスクリプトを動作させるための環境を用意するということも意味しています。例えばこの例では、サンプルイメージを評価し TeleForm Verifier で修正する作業が必要です。（サンプル イメージは total_time フィールドに不正な値が入っています）

1. TeleForm Reader を起動します。

2. 「ファイル」メニューから「イメージの評価」を選択します。「ファイルを開く」ダイアログ ボックスが表示されます。
3. 「Image01.tif」を選択し「開く」をクリックします。
4. TeleForm Reader はサンプル イメージを評価します。この時、Form_Evaluate エントリー ポイントが呼ばれ、実行されます。
5. TeleForm Verifier を起動します。
6. 「テンプレート」から「Chapter 1 Exercise」を選択します。
7. 「保存されているイメージ」から「Image01.tif」を選択し「修正」ボタンをクリックします。
8. フォーム モードが開始されるので、total_time フィールドまで TAB で移動します。
9. 次のフィールドに移るために TAB を押します。この時、下の図のようなエラー メッセージが表示されます。

10. 「OK」をクリックし total_time フィールドに戻ります。
11. total_time フィールドに正しい時間 (08:30) を入力し、TAB を押します。今度は、Verifier はこの値を受け入れてくれます。

1-4. 演習

この演習では、「1-3. VBA ツアー」を実践します。
以下の手順に従って演習を進めてください。

1. 目の前に次の環境があることを確認してください。
   - TeleForm v10.1 がインストールされている Windows マシン
     （以下のアプリケーションがインストールされていることを確認してください）
     - TeleForm Designer
     - TeleForm Reader
     - TeleForm Verifier

2. 「1-3. VBA ツアー」のステップ 1～5 を行ってください。

第2章　基本構文

この章では以下のことを学びます。

・ 構文の特徴
・ 構文の各要素
・ よく使用する関数とサブルーチン

注意：
この章で紹介される言語仕様は完全ではありません。詳細は VBA のオンライン ヘルプを参照してください。また、VBA は Microsoft Visual Basic 6.0 の言語仕様とほぼ同等です。Microsoft Visual Basic 6.0 のヘルプも参考になりますので必要に応じ参照してください。

2-1. 構文の特徴

2-1-1. 全体の流れ

全体的に構文は以下のような流れとなっています。

1. 変数を宣言します。
2. 関数、サブルーチンを宣言します。関数に与える引数の名前と型、および関数の返す値の型の定義を記述します。
3. 関数およびサブルーチンを記述します。

VBA では大文字小文字は区別しません。フィールドの値は大文字、小文字を区別します。

また、命令と命令の間にある空白は無視されます。空白をあけないでプログラムを記述することも可能ですが、見やすいように適切に空白を設けます。ブロック(Sub－End Sub、Function－End Function、For－Next、Do－Loop、If－Then－End If)の中は前の段より 4 文字または 2 文字空白を設けて字を下げます。(この字下げのことをインデントと言います)

例:

```
Sub Test
    Dim i As Integer

        For i = 0 To 10
            If i = 5 Then
                MsgBox “i is 5“
            End If
        Next
End Sub
```

2-1-2. コメント

コメントは実行されませんので覚え書きとして使用できます。後でプログラムを読むときにわかるように記述をします。コメントには「'」を使用します。

例:

```
’ Comment
Sub Test
```

2-1-3. 値

整数は「.」(ピリオド)を含まない数値です。浮動小数点数は「.」を含む数値です。100 は整数ですが、100.0 は浮動小数点数です。

また、文字列は「"」(ダブルクォート)で囲みます。

2-1-4. 名前

名前とは変数、および関数・サブルーチン名として使用できる文字列です。名前の先頭はアルファベットでなければいけません。それ以降では英数字が使用できます。また「_」(アンダー バー)も使用できます。名前の大文字、小文字は区別されません。つまり変数として Aa を宣言した場合、AA、aA、aa も同じように使用できます。

名前として予約語を使用することはできません。予約語とは、すでに定義済みの名前、VBA キーワードのことです。例えば long という名前は使用できません。

以下が予約語の一覧です。

Access	Alias	And	Any
Append	As	Base	Begin
Binary	Boolean	ByRef	ByVal
Call	CancelButton	Case	CDecl
CheckBox	Chr	ChrB	ChrW
Close	ComboBox	Compare	Const
CStrings	Currency	Date	Declare
Default	DefBool	DefCur	DefDate
DefDbl	DefInt	DefLng	DefObj
DefSng	DefStr	DefVar	Dialog
Dim	Do	Double	DropListBox
Else	ElseIf	End	Eqv
Error	Exit	Explicit	For
Function	Get	Global	GoSub
Goto	GroupBox	HelpButton	If
Imp	Inline	Input	Input
InputB	Integer	Is	Len
Let	Lib	Like	Line
ListBox	Lock	Long	Loop
LSet	Mid	MidB	Mod
Name	New	Next	Not
Nothing	Object	Off	OKButton

On	Open	Option	Optional
OptionButton	OptionGroup	Or	Output
ParamArray	Pascal	Picture	PictureButton
Preserve	Print	Private	Public
PushButton	Put	Random	Read
ReDim	Rem	Resume	Return
RSet	Seek	Select	Set
Shared	Single	Spc	Static
StdCall	Step	Stop	String
Sub	System	Tab	Text
TextBox	Then	Time	To
Type	Unlock	Until	Variant
WEnd	While	Width	Write
Xor			

2-2. 構文の要素

2-2-1. 変数

変数には値(文字)を入れることができます。変数は使用する前に型を宣言してください。宣言をしなくても変数を使用することはできますが、多くの場合解決の困難なバグの原因となります。変数の宣言は以下のように行います。宣言なしで変数が使用された場合は、使用された時点で変数がバリアント型で作成されます。

Dim（変数名）As（変数の型）

例:

```
Dim a As Integer, b As Long
```

変数の型とはその変数に入れることのできる値の集合のことです。以下の代表的な型があります。

Integer	整数型 (-32768 ～ 32767)
Long	長整数型 (-2^32 ～ 2^32-1)
Double	浮動小数点型
String	文字列型

変数が初期化されたき、変数に入っている値は、文字列変数の場合は空文字(“”)、数値型の場合は 0 です。しかし、これを期待してプログラムを作成することは奨励できません。変数は参照する前に適切な値で初期化してください。

関数･サブルーチンの中で宣言された変数はその中でしか有効でありません。関数 foo で変数 i を宣言して、これに 100 を入れたとします。別の関数で i を見たとき、その値は 100 である保証はありません(見ることができません)。このような変数を局所変数(ローカル変数)と言います。関数･サブルーチンの外で宣言された変数はすべての範囲で使用することができます。これを広域変数(グローバル変数)と言います。(このように変数の有効な範囲のことをスコープと言います)

変数への値の代入は = 演算子を使用します。

例:

```
Dim A As Integer, S As String
A = 1           ’Aに1を代入します。
S = “123”       ’Sに”123”を代入します。
```

数値型(整数型、浮動小数点型...)同士の代入は自動変換されますが、数値型と文字列型の代入は変換関数が必要です。

例:

```
Dim A As Double, B As Integer, S As String
A = 2.3
B = A              'B には 2 が入ります。(整数変換されます)
S = CStr(A)        'S には "2.3 "が入ります。
```

2-2-2. 定数

定数の宣言は以下のように行います。

```
Const (定義名) = (定数値)
```

例:

```
Const konnitiha = "Hello World!! "
Const Pi = 3.14159265358979
```

2-2-3. 構造体型

構造体型とは、複数の変数が集まったものです。例えば構造体 People を定義します。

例:

```
Private Type people
    sName As String
    iAge As Integer
    sAddress As String
End Type
```

この構造体は sName、sAddress という文字列変数、iAge という整数変数をもつ構造体です。この構造体の使用例を以下に示します。

例:

```
Dim Tarou As people, Jirou As people

Tarou.sName = "Tarou "
Tarou.iAge = 10
Jirou.sName = "Jirou "
```

2-2-4. 演算子

VBA では以下の演算子が使用できます。

変数への代入	=
四則演算	+ - * /
べき乗演算	^
整数除算	¥ (例: 5 ¥ 2 の結果は 2 です)
剰余算	Mod (例: 5 Mod 2 の結果は 1 です)
文字列の結合	+ & (例: “AB“ + “C“ の結果は “ABC“ です)
比較演算	= <> > < <= >=
論理演算･ビット演算	And Or Not Xor

演算子の優先順位は一般の数字と同じですが、式が複雑な場合には()を用いて明示的に示すようにしましょう。

例:

```
If ((A = 1) And (B = 2)) Or ((A = 2) And (B = 1)) Then
```

2-2-5. 関数とサブルーチン

関数は呼び出されると仕事(処理)を行い、何かの値を返す処理の単位です。サブルーチンは関数と違い値を返しません。

関数は以下のように定義します。(ここでは foo という関数を定義します)

例:

```
Function foo(a As Integer, b As Integer) As Integer ’関数の定義を開始
    If a > b Then       ’a が b より大きいなら
        foo = a         ’この関数は a を返します
    Else                ’そうでないなら…
        foo = b         ’この関数の戻り値を b とします
    End If              ’条件文の終わり
End Function            ’関数の定義の終わり
```

この関数は a、b の 2 つの整数を受け取り大きい方の整数を返します。関数の返す値を設定するには、その関数の名前に値を代入することで行います。

サブルーチンは以下のように定義します。

例:
```
Sub bar(A As String)
    (処理内容)
End Sub
```

関数やサブルーチンの途中で、それらを終了したい場合があります。その時は Exit Function または Exit Sub を使用します。

次の例は、再起呼び出しという手法を用いて階乗(!)を計算します。5!は 5*4*3*2*1 = 120 です。

例:
```
Function kaijou(n As Long) As Long
    If n = 1 Then
        Kaijyou = 1
        Exit Function
    End If
    Kaijyou = n * Kaijyou(n - 1)
End Function
```

これは次のようにも記述できます。

例:
```
Function kaijou(n As Long) As Long
    If n = 1 Then
        Kaijyou = 1
    Else
         Kaijyou = n * Kaijyou(n - 1)
    End If
End Function
```

関数やサブルーチンを呼び出すには、次のようにします。

例:
```
i = Kaijyou(5)
```

```
bar “Hello!! “
```

2-2-6. 制御構造

If 文は条件によってある処理を行わせたい場合に使用します。今 A が 1 で B が 2 だとします。次の条件を実行すると A に 2 が入ります。

例:

```
If A < B Then B = A Else A = 0
```

もし A が-1 だと A には 0 が入ります。

行わせたい処理が複数行ある場合は、

例:

```
If A < B Then
    B = A
    A = 0
Else
    A = B
    B = 0
End If
```

のように記述します。

For 文は繰り返し文です。次の例では 1 + 2 +… + 9 + 10 を計算します。

例:

```
sum = 0
For i = 1 To 10
    sum = sum + i
Next
```

For－Next の間の文は必ず 1 度は実行されます。次の例では sum に 1 が入ります。

例:

```
sum = 0
```

```
For i = 1 To 0
    sum = sum + i
Next
```

For 文の他に繰り返しを行う命令として、Do－Loop 文があります。

例:

```
sum = 0
i = 1
Do
    sum = sum + i
    i = i + 1
Loop While i < 11
```

例:

```
sum = 0
i = 1
Do
    sum = sum + i
    i = i + 1
Loop until i > 10
```

Do－Loop 内の文を実行する前に条件判定を行うか、実行した後に条件判定を行うかの区別は重要です。次の文では内部の命令は実行されるので x に-1 が入ります。

例:

```
i = 0
x = 0
Do
    x = x - 1
    i = i + 1
Loop While i < 0
```

しかし、次の例では x は 0 のままです。

例:

```
i = 0
```

```
x = 0
Do While i < 0
    x = x - 1
    i = i + 1
Loop
```

2-2-7. よく使用する関数とサブルーチンの説明

（[ ]で囲まれた引数は省略可能な引数です）

〈数値操作〉

関数　　Abs(数値)

数値の絶対値を返します。つまり Abs(-1)は 1、Abs(1)も 1 です。

関数　　Int(数値)

数値の小数点を取り除いた、整数部分を返します。Int(1.5)は 1 です。

〈文字列の操作〉

関数　　Cstr(数値)

数値を文字列にして返します。数値には整数、小数ともに使用できます。

例:

```
CStr(100)      '→ "100 "
CStr(-100)     '→ "-100 "
```

この関数は、数値を表示したい場合 MsgBox 文とともによく使用します。

例:

```
MsgBox "Value of X is "  + CStr(X)
```

変数 X の値を表示するメッセージボックスを表示します。

関数　　Val(文字列)

文字列で表現された数値を数値型に変換します。変数 X が整数型で文字列 "123" で表現された数値を X にしたい場合、A = "123" ではエラーとなりますので、A = Val(123) とします。

関数　　Left(文字列、数値)　Right (文字列、数値)

それぞれ、文字列の左から数値で指定された文字数分、右から指定された文字数分を取り出しま

す。

例:
Left(“1234“, 2)　　‘→ “12“
Right(“1234“, 2)　　‘→ “34“

関数　　String(数値、文字列)
文字列を指定された数値回繰り返した文字列を返します。

例:
String(3, “A“)　　‘→ “AAA“

関数　　Mid(文字列、先頭位置、[文字数])
文字列の先頭から指定された文字数を取り出します。文字数が省略されたときは先頭から後ろの文字列を返します。

例:
Mid(“1234“, 2, 2)　　‘→ “23“
Mid(“1234“, 2)　　‘→ “234“

サブルーチン　　Mid(文字列変数、先頭位置、文字数) → 置換文字列
文字列変数の先頭から文字数で指定された文字数を置換文字列で置き換えます。

例:
A = “1234“
Mid(A, 2, 2) = “56“　　‘→ A には “1564“ が入ります。

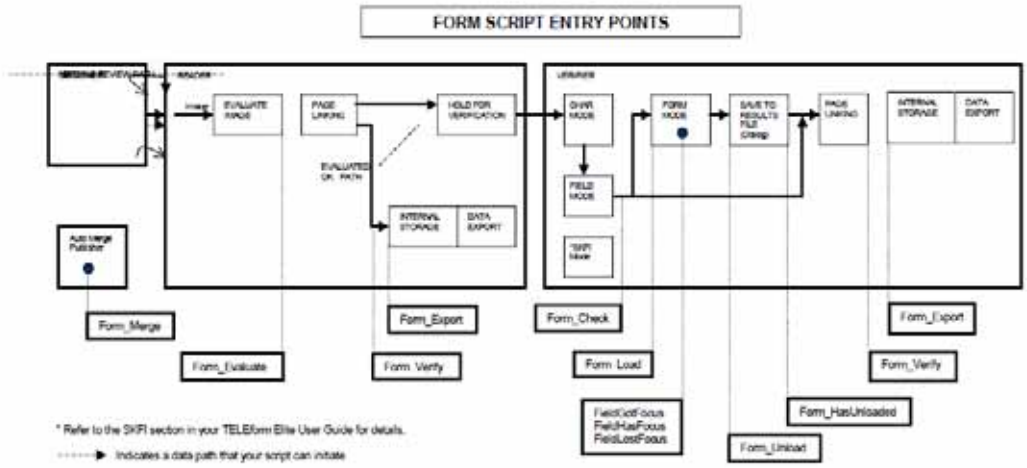
第3章　フォーム スクリプト

この章では以下のことを学びます。

- エントリー ポイント
- Form クラス
- Fields コレクション
- Field クラス

3-1. エントリー ポイント

次の図は、各フォーム スクリプトのエントリー ポイントが TeleForm のデータ フロー上、いつ呼ばれるかを示しています。



このようにエントリー ポイントはフォーム処理の中で順に処理されます。各エントリー ポイントの詳細は次のページから紹介しています。

エントリー ポイント	説明
Sub Form_Merge ： End Sub	このエントリー ポイントは TeleForm Auto Merge Publisher でのフォーム マージ処理の一番初めに呼ばれます。フォーム上のマージされるデータはスクリプト内のフィールドと一致します。このエントリー ポイントでユーザはマージ前にデータを修正できます。 注意: マージはフォーム スクリプトで中断できません。Form_Merge エントリー ポイントはマージ処理で使用されるデータを修正するためだけに使用されます。
Sub Form_Evaluate ： End Sub	このエントリー ポイントは TeleForm Reader に評価された後、すぐに呼ばれます。 マルチページ フォームの場合: ・ マルチページ フォームのリンクされたグループ毎に実行されます。 ・ ページ リンク フィールドがいくつかのページ結合に失敗するとフォームは複数回処理されます。 このエントリー ポイントは、フォーム データの評価や再評価用にフィールドにマークをしたりするのに頻繁に使用されます。 このエントリー ポイントは TeleForm Internet Server で受け取った全ての HTML フォームで呼ばれます。
Sub Form_Check ： End Sub	TeleForm Verifier でイメージが修正されている時、文字モードとフィールド モードが終了した後、フォーム モードに入る前にこのエントリー ポイントが呼ばれます。このエントリー ポイントを使用してフォーム モードへ入る前に、修正データを保存したり評価を実行したりできます。 TeleForm Verifier で同時に複数のイメージを修正している場合は次の事柄が適用されます: ・ フォームの全フィールドが文字モードやフィールド モードで評価される場合、Sub Form_Check はそのフォームが文字モードやフィールド モードで評価され続ける間、バックグラウンド プロセスとして初期化されます。 ・ フォーム モードがどのフォームでも必要な場合、Sub Form_Check はフォーム モード発生前に初期化されます。 ・ フォームが SKFI 領域を含んでいた場合、Form_Check は文字モード、フィールド モード、SKFI モードが終了した後に呼ばれます。 注意: Form_Check での評価が 1 秒または 2 秒以上かかる場合、Verifier オペレーターは、Verifier がフォーム モードへデータを流す準備をしている間、「Waiting for validation」というメッセージを受け取るかもしれません。
Sub Form_Load ： End Sub	このエントリー ポイントは TeleForm Verifier でフォーム モードに入るとすぐに呼ばれます。

エントリー ポイント	説明
Sub FieldGotFocus : End Sub	このエントリー ポイントは TeleForm Verifier のフォーム モードでフィールドがフォーカスを受け取る直前に呼ばれます。
Sub FieldHasFocus : End Sub	このエントリー ポイントは TeleForm Verifier のフォーム モードでフィールドがフォーカスを受け取った時（フィールドが反転した時）に呼ばれます。Sub FieldHasFocus はそのフィールドに関してオペレーターから情報を取得するのに使用されます。
Sub FieldLostFocus : End Sub	このエントリー ポイントは TeleForm Verifier のフォーム モードでフィールドがフォーカスを失った時（フィールドからタブ アウトした時）に呼ばれます。Sub FieldLostFocus は修正されたデータをすぐにチェックし Verifier オペレーターによる不正なデータ投入を防ぐために使用されます。
Sub Form_Unload : End Sub	このエントリー ポイントは TeleForm Verifier でフォームが閉じられる前に呼ばれます。正確には変更を保存するダイアログが表示される直前か、ユーザが手動でフォームを閉じた時に呼ばれます。これはデータのダブルチェックや再度修正をさせるためにフォームのステータスを強制的に変更するのに使用されます。 このエントリー ポイントは SKFI ストリーミング モードの SKFI 領域から出た直後にも呼ばれます。
Sub Form_HasUnloaded : End Sub	このエントリー ポイントは Sub Form_Unload の直後に呼び出されます。変更を保存するダイアログは Sub Form_HasUnloaded が呼び出される前に表示されます。これは Form_Load で開いたファイルを閉じるのに使用されます。
Sub Form_Verify : End Sub	このエントリー ポイントは Verifier オペレーターが全てのフォーム イメージを修正した後で呼び出されます。このエントリー ポイントを使用して完全なクロス バリデーションを行うことができます。評価が正しくなければ再度評価をさせることができます。 このエントリー ポイントは Verifier による評価プロセスを通っていない場合でも呼び出されます。（TeleForm Reader で評価 OK となったフォームのこと） TeleForm Reader や Verifier で Form_Verify に入る前に全てのページ リンク処理は終了しています。 注意： Form_Verify は、メイン（ユーザ） スレッドで実行される他のエントリー ポイントとは異なる（バックグラウンド）スレッドで実行されます。Control Center はメイン スレッドの動作をログに書き込み、バックグラウンド スレッドの動作は書き込みません。そのため Form_Verify での動作はログに書き込まれません。
Sub Form_Export : End Sub	このエントリー ポイントはデータ ファイルへデータがエクスポートされる直前に呼ばれます。Sub Form_Export はエクスポート前にデータを変更することができます。そのため、エクスポート スクリプトで定義されたデータ フォーマットに変更したりすることが可能です。

- フィールド固有のエントリー ポイント

フォーム評価時に、あるフィールドのためだけにスクリプトが動作すると便利なことがあります。これを利用すると、修正されたデータをすぐにチェックしたり、Verifier オペレーターからの不正データ入力を防いだりすることができます。この機能を有効にするために、VBA は各フィールド用に 3 つのエントリー ポイントを用意しています。

TeleForm Verifier のフォーム モード時にフィールドがフォーカスを受け取ったり失ったりすると、次のエントリー ポイントが実行されます。

エントリー ポイント	説明
Sub *FieldName*_GotFocus ： End Sub	このエントリー ポイントは TeleForm Verifier のフォーム モードで該当のフィールドがフォーカスを受け取る直前（フィールドへタブ インした時）に呼ばれます。
Sub *FieldName*_HasFocus ： End Sub	このエントリー ポイントは TeleForm Verifier のフォーム モードでフィールドがフォーカスを受け取った時（フィールドが反転した時）に呼ばれます。
Sub *FieldName*_LostFocus ： End Sub	このエントリー ポイントは TeleForm Verifier のフォーム モードでフィールドがフォーカスを失った時（フィールドからタブ アウトした時）に呼ばれます。

スクリプトが通常のエントリー ポイント（FieldGotFocus や FieldLostFocus など）とフィールド固有のエントリー ポイントを定義していた場合、通常のエントリー ポイントがフィールド固有のエントリー ポイントより先に呼ばれます。

今 total というフィールドがあると仮定した場合、フィールド固有エントリー ポイントは以下のように定義されます。

- total_GotFocus
- total_HasFocus
- total_LostFocus

3-2. フォーム スクリプト クラスとプロパティ

TeleForm には 8 つのオブジェクト クラスが存在します。各クラスは固有のプロパティ セットを持っています。これらのプロパティを使用するとあらゆる TeleForm 情報にアクセスすることが可能です。

3-2-1. Form クラス

Form クラスのプロパティは現在フォーム スクリプトで処理されているフォームの情報を保持しています。Form クラスの情報を参照するには以下のようにします。

```
Form.FormPropertyName
```

FormPropertyName は Form クラスの適切なプロパティを指しています。例えば以下のようになります。

```
id = Form.FormID          'フォーム ID を id に代入
name = Form.Title         'フォームのタイトルを name に代入
```

Form クラスのプロパティは以下のように定義されています。

プロパティ	型	アクセス	説明
Title	String	Read Only	フォームの名前。
FormID	Long	Read Only	フォームにアサインされた ID。
Mode	tfFormMode enum	Read Only	現在の処理モード。
Image	String	Read Only	フォーム イメージのパス。
Status	tfFormStatus enum	Read Only	フォームの状態を格納。フォーム評価時は再評価が必要か、評価 OK のどちらかが指し示されています。Verifier 時は、修正されたデータをどのようにコントロールするか決定することができます。「Form.Status プロパティ値」で更に詳細を説明します。
CurField	String	Read Only	GotFocus、HasFocus、LostFocus エントリー ポイントのみで定義されます。このプロパティは現在のフィールド名を保持します。
CurGroup	String	Read Only	GotFocus、HasFocus、LostFocus エントリー ポイントのみで定義されます。このプロパティは現在のフィールドが属する詳細グループ名を保持します。現在のフィールドが詳細グループに属さない場合はこのプロパティは空です。

- Form.Mode プロパティ値

値	名前	説明
0	**tfFormModeUnknown**	不明な Mode
1	**tfFormEval**	Reader: フォーム評価
2	**tfFormFill**	Reader: HTML またはフォーム処理、またはフォーム モードにおける SKFI データエントリーの処理
3	**tfFormSuspense**	Verifier: フォーム モードで修正、フォーム モードのキャプチャデータ処理や新しいフォームの補充
4	**tfFormExporting**	Reader, Verifier, Designer: データのエクスポート
5	**tfFormFinalValidate**	Reader, Verifier: Form_Verify イベントでの評価または修正終了
6	**tfFormCheck**	評価または修正終了。Verifier: 文字モード、フィールド モード、キャプチャ モードからフォーム モードへ遷移
7	**tfFormMerge**	Auto Merge Publisher: Form_Merge のフォームのマージ。この値はマージ前のフォームの全フィールドへの読み書きアクセスを許可します。Remote_Fax や Remote_Phn のようなフォーム上に印刷されない仮想マージ フィールドはこれに含まれません
8	**tfFormSKFI**	Verifier: キャプチャ モードでデータのインデックスを入力
9	**tfFormDataReview**	Verifier: データ再評価モードでのクオリティ コントロールの実行

- Form.Status プロパティ値

次の表は TeleForm Reader と Verifier がエントリー ポイントでセットできる Status プロパティ値を示しています。他のエントリー ポイントでは Form.Status を設定することはできません。

定数	定義	評価 (Form_Evaluate) (Form_Verify)	修正 (Form_Unload)	修正 (Form_Verify)
0	**tfFormAccept**	評価 OK	修正をファイルに保存	修正をファイルに保存
1	**tfFormSuspend**	要再検査	フォーム モードに戻る	後の評価のために変更保存
2	**tfFormCancel**	未定義	後の評価のために変更を保存せずフォーム モードを終了	未定義
3	**tfFormSaveAndExit**	未定義	現在の状態でフォームを保存しフォーム モードを終了	未定義

3-2-2. tfFields コレクション(配列)

tfFields コレクションはフォーム スクリプトとエクスポート スクリプトで使用できます。tfFields は、処理中のフォームの全フィールドへのアクセスを提供するコレクション オブジェクトです。他のコレクション オブジェクトと同様に、tfFields コレクションにはフォームのフィールド数を示す Count プロパティがあります。これは詳細グループの中のフィールドは含めません。トップ レベル フィールドの名前つきフィールド数を示します。キャプチャ領域、詳細グループ、仮想フィールドを含むアドレス グループのメンバー フィールドは、このコレクションに含まれています。

注意:

詳細グループは各列に tfFields コレクションがあります。

- tfFields コレクション情報の参照

tfFields コレクションはフォームのフィールドのセット、または詳細グループの列のフィールドを表しています。これらのコレクションは各アイテムへアクセスするのに配列構造を利用します。1*は配列の最初の要素を示します。各 tfFields の tfField オブジェクトのプロパティを参照するには以下のようにします。

Form.Fields(i).*PropertyName* または Form.Fields(*FieldName*).*PropertyName*

* 従来の BasicScript™ の 0 と異なるため注意してください。

i	1 から Fields.Count の間の整数
FieldName	フォームのフィールドのフィールド ID
PropertyName	フィールド クラスの適切なプロパティ

例:

```
Dim i As integer
For i = 1 To Form.Fields.Count
    DispMsg “The “ & i & “th field is “ & Form.Fields(i).Name
Next i

If Form.Fields.Exists(“Name “) Then
    DispMsg “The Name field contains the value “ & Form.Fields(“Name “).Text
End If

Dim Field as TfField
For Each Field in Form.Fields
   DispMsg Field.Name & “.Value = “ & Field.Value
Next Field
```

次のプロパティが Fields コレクションで定義されています。

プロパティ	型	アクセス	説明
Count	Long	Read Only	フォームのトップ レベルのフィールド数。エクスポート時はエクスポートされたフィールド数となります。
Exists			フィールドが実在するか確認
*Item( )	tfField Object	Read Only	コレクションに含まれる各フィールド オブジェクト。括弧の中の値は次のようになります: ・ 1 から Fields.Count の間の整数 ・ フィールド名 注意: このプロパティにバリアントは使用できません。 文字列が Fields コレクションの中の適切なフィールドを示していない場合、ランタイム エラーが発生します。**Exists** プロパティでフィールドの存在を確認してください。 整数が 0 から Fields.Count - 1 の間でなかった場合、ランタイム エラーが発生します。

*はこのオブジェクトのデフォルト プロパティです。

3-2-3. tfField オブジェクト

エクスポート リストにあるフィールドやフォームのフィールドを表します。tfFields オブジェクトの Item プロパティを用いてアクセスします。フォーム上のフィールドのデータや属性へのアクセスを提供します。これは最もよく使用されるオブジェクトです。

tfField オブジェクトはフォーム スクリプト、エクスポート スクリプト、ライブラリ スクリプトで使用することができます。エクスポート スクリプトでは Fields コレクションを通じてのみ Field オブジェクトにアクセスすることができます。

- tfField オブジェクト情報の参照

tfField オブジェクト情報を参照するには以下のようにします。

FieldName.FieldPropertyName

FieldName	フォームのフィールドのフィールド ID
FieldPropertyName	フィールド クラスの適切なプロパティ

例:

City:

S	A	N		F	R	A	N	C	I	S	C	O		

State:


```
If City.Text = “SAN FRANCISCO “ Then State.Text = “CA “
```

各 Field オブジェクトは次のプロパティを持っています。

プロパティ	型	アクセス	説明
Name	String	Read Only	フィールドのフィールド名 (ID)。
Type	tfFieldType enum	Read Only	データ フィールド形式を表した事前に定義された値。
Text	String	Read-Write	テキスト フォーム中のフィールドに関係したデータの値。空白部はスクリプトの実行前に削除されます。 注意: 全ての TeleForm オブジェクトのプロパティ同様、Text プロパティは直接代入することでのみ値を変更することができます。関数内で引数として渡された場合には、値を変更することができません。

プロパティ	型	アクセス	説明
*Value	Variant	Read-Write	フィールドに関係したデータの値。TeleForm では、tfField.Type を基にして tfField.Text を正しいデータ型に変換し、tfField.Value で戻します。
Status	tfFieldStatus enum	Read-Write	フィールド ステータスを表す。フィールドが評価されたり修正されたりすると、このプロパティはフィールドが OK か、要再検査か指し示します。
Missing	Boolean	Read Only	フィールドが属しているフォームのページの欠落ステータス。このプロパティは Form_Evaluate でのみ利用できます。 True: フォーム ページは欠落している False: フォーム ページの欠落なし マルチページ フォームの各ページは個別に評価されるため、フィールド評価の前にこのプロパティを確認する必要があるかもしれません。例えば SSN.Missing が True の場合、現在評価中のページ セットに SSN フィールドが含まれていないことを意味しています。この行方のわからないフィールドは、後続のスクリプト呼び出しで処理されることになります。
Mask	String	Read-Write	Text プロパティの各文字の状態を表した 0 から 9 までの文字。0 は認識に成功したことを表しており、その他の値は次のように認識に失敗したことを表しています: 0: 認識成功 1: 他のエラー 2: 予約 3: 未入力 4: 複数マーク 5: 不正文字 6: あいまいなマーク 7: マーク (辞書にない単語) 8: マーク (最高精度の文字) 9: マーク (最高精度が見つからない) 重要: このプロパティは常に利用可能ではありません。(HasMask プロパティを参照) Choice フィールドは選択肢ごとに 1 つの Mask を持っています。

*はこのオブジェクトのデフォルト プロパティ

プロパティ	型	アクセス	説明
HasMask	Boolean	Read Only	フィールドの Mask ステータス: True: Mask プロパティ有効 False: Mask プロパティ利用不可 重要: Field.HasMask が False の時に、Field.Mask にアクセスするとランタイム エラーが発生し、スクリプトの実行が終了します。
Length	Long	Read Only	Text プロパティの最大文字数。
TabIndex	Long	Read-Write	TeleForm Verifier のフォーム モードでの、フィールドのタブ オーダー。 この値は 0 から Fields.Count - 1 の間になります。 新しいタブ インデックスが設定されると、そのインデックス以降のフィールドのオーダーがずれます。 Form_Evaluate でこのプロパティにアクセスするとエラーになります。
Description	String	Read Only	Designer で設定したフィールドの説明
FormPageNumber	Long	Read Only	Designer で作られたフォーム上のフィールドのページ番号。最初のページが 0。
Left	Long	Read Only	フィールドの左端の X 座標。
Right	Long	Read Only	フィールドの右端の X 座標。
Top	Long	Read Only	フィールドの上端の Y 座標。
Bottom	Long	Read Only	フィールドの下端の Y 座標。
Choices	tfChoices Object	Read Only	tfChoice オブジェクトのコレクション。 Choice オブジェクトでない場合はスクリプト エラーが発生します。 Choices はコレクションなので Count プロパティを持っています。

プロパティ	型	アクセス	説明
TopChoices	tfTopChoices Object	Read Only	TopChoice オブジェクトのコレクションです。各オブジェクトは Text プロパティの文字に相当します。 このプロパティは HasTopChoices プロパティが True の時に利用できます。 重要: このプロパティは常に利用可能ではありません。(HasTopChoices プロパティを参照)
HasTopChoices	Boolean	Read Only	TopChoices プロパティのステータス。 True: TopChoices プロパティが定義されている False: TopChoices プロパティが定義されていない 重要: この値が False の時に、TopChoices にアクセスするとランタイム エラーが発生し、スクリプトの実行が終了します。
SetFocus	Method		TeleForm Verifier でフィールド選択に用います。 マウス操作でフィールド選択をしている場合、SetFocus は無視されます。ただし、ユーザがフォーカスを変更しようとしているフィールドに SetFocus がされている場合は別です。これはフィールドに正しい値を入力させることを保証するのに利用できます。 SetFocus は LostFocus や GotFocus イベントで使用できます。 注意: キャプチャ モードでは、現在のキャプチャ領域外のフィールドへの SetFocus は無視されます。
LoseFocus	Method		Verifier でタブ インデックスで指定されている次のフィールドを開きます。 注意: HasFocus イベントでのみ使用できます。

プロパティ	型	アクセス	説明
Rows	tfRows	Read Only	列のコレクション。このコレクションは詳細グループのみに適用されます。この値は 1 から *DetailField*.Rows.Count の間になります。 注意: 列自身もコレクションなので Count プロパティを持ちます。
CurRow	Long	Read Only	現在処理されている列。この値は GotFocus、HasFocus、LostFocus のみで利用可能で、詳細グループのメンバーであるフィールドに適用されます。
ImagePage-Number	Long	Read Only	現在処理されているフォーム イメージのページ ナンバー。イメージの最初のページはページ 0 です。
DoubleKey	Boolean	Read Only	フィールドがダブル キーとマークされている場合、Field.DoubleKey は True となり、それ以外では False となります。 注意: Field.DoubleKey はデータ再評価モードとフォーム モードのみで定義されます。フォーム モード修正中は False となります。
Image-Orientation	tfImageOrientation enum	Read Only	現在処理されているフォーム イメージの方向を格納。このプロパティは次の値がセットされます: ImageRotate90 – ページは 90 度回転です ImageRotate180 – ページは 180 度回転です 全ての回転は、Left、Right、Top、Bottom 座標を適用する前に左回転で適用されます。つまり、座標適用前にどのくらいイメージを回転させるかを決めるプロパティです。

- *FieldName*.Type プロパティ値

Field.Type プロパティは次の表の値のいずれかとなります。

名前	値	説明
tfInvalidType	0	不正
tfNumberType	1	数字
tfStringType	2	文字列
tfTextFileType	3	テキスト ファイルのファイル名
tfImageFileType	4	イメージのファイル名
tfDetailType	5	詳細グループ レコード
tfChoiceType	6	選択フィールド
tfDataType	7	データ フィールド
tfVarCharType	8	VarChar
tfSKFIType	9	キャプチャ領域
tfBinaryType	10	バイナリ

- *FieldName*.Status プロパティ値

例:

```
total.Status = (total.Status Or tfFldInvalid)
Field.Status = Field.Status AND NOT tfFldReview
```

Or を使用すると、すでに total フィールドが持っている tfFldReview のようなステータスを保持することが可能です。また二つ目の AND を例では、Field.Status から tfFldReview ステータスのみを取り除くことができます。

名前	10 進法値	16 進法値	説明
tfFldOK	0	0000 0000	評価 OK
tfFldNotFilled	1	0000 0001	入力なし
tfFldThreshold	2	0000 0002	あいまいな選択やエントリー
tfFldRange	4	0000 0004	フィールドのプロパティ画面で定義される数値範囲外のデータ
tfFldTooMany	8	0000 0008	選択フィールドでの複数選択
tfFldBadInterp	16	0000 0010	低精度の文字認識
tfFldIOError	32	0000 0020	ファイル書き込み失敗 (イメージ領域)
tfFldReview	64	0000 0040	フィールドのプロパティ画面で「常に閲覧」がチェックされている
tfFldInvalid	128	0000 0080	フィールド評価失敗
tfFldLookup	256	0000 0100	データベースルックアップエラー、不正な値
tfFldDictionary	512	0000 0200	辞書にない単語
tfFldIllegalChar	1024	0000 0400	不正文字
tfFldMissingPg	2048	0000 0800	ページ欠落
tfFldBlankZone	4096	0000 1000	イメージ領域に入力なし
tfFldLengthErr	8192	0000 2000	長さが不正 (バー コード)
tfFldKeyNotFound	16384	0000 4000	フィールドのキーが見当たらない
tfFldWordChg	32768	0000 8000	辞書による単語変換
tfFldLookupTemplate	65792	0001 0100	テンプレート ルックアップの失敗
tfFldLookupDate	131328	0002 0100	データ ルックアップの失敗
tfFldInvalidCurrency	196864	0004 0100	通貨ルックアップの失敗
tfFldUnknownLex	262400	0008 0100	Lex ルックアップの失敗
tfFldSpellChkFailed	327936	0005 0100	スクリプト スペル チェッカーの失敗
tfFldInvalidDate	268435456	1000 0000	不正な日付
tfFldBestGuess	536870912	2000 0000	最高精度の文字
tfFldLLocationErr	4000 0000	0008 0100	不正なフィールド オブジェクト位置

3-3. 演習

- 準備

演習を始める前にまずは環境を準備します。以下の手順を行ってください。

1. TeleForm Designer で「Chapter 3 Exercise」を開いてください。
   （「Chapter 3 Exercise」は［02_フォーム］の「Chapter 3 Exercise.tft」です。）
2. 「フォーム」メニューから「スクリプト」を選択してください。

- 演習

「Chapter 3 Exercise」は、「Chapter 1 Exercise」と同一の単純なタイムカードです。ユーザは勤務の開始時間と終了時間を記入し、総勤務時間を合計に記入します。（お昼などの休憩は考慮しません）

作成するスクリプトは、開始時間と終了時間を使って合計が正しいかどうかを確認し、正しくない場合は Verifier オペレーターにメッセージ ボックスを表示し、修正させます。

フォームは、以下の 3 つの区切り付きフィールドで構成されています。

- Begin_Time　（開始時間）
- End_Time　（終了時間）
- Total_Time　（合計）

1. まずはスクリプトの先頭にコメントを書きます。以下のコメントを書いてください。

```
'*****************************************************************
'   タイム カード フォーム スクリプト
'
'   このスクリプトは「開始」から「終了」までの時刻を自動計算し
'   「合計」と照合します。
'*****************************************************************

Sub Form_Evaluate
```

2. 次にスクリプトで使用する定数と関数宣言を書きます。コメントと Sub Form_Evaluate の間に以下のように書いてください。

```
'*************************************************************

Const SumFail    = 0
Const SumSuccess = 1

Sub Form_Evaluate



```

3. 合計が正しいかを確認する関数 SumOK() を作成します。スクリプトの一番下にその関数の実体を定義します。

```
End Sub

Function SumOK() As Integer

        Dim start_min As Integer
        Dim end_min   As Integer
        Dim total_min As Integer

        start_min = Val(Left(Begin_Time.Text, 2)) * 60 + Val(Mid(Begin_Time.Text, 3, 2))
        end_min = Val(Left(End_Time.Text, 2)) * 60 + Val(Mid(End_Time.Text, 3, 2))
        total_min = Val(Left(Total_Time.Text, 2)) * 60 + Val(Mid(Total_Time.Text, 3, 2))

        If total_min = end_min - start_min Then
                SumOK = SumSuccess
        Else
                SumOK = SumFail
        End If

End Function
```

4. フォーム評価の時に SumOK() を実行し、合計が正しくない場合、Total_Time.Status を tfFldInvalid にします。

```
Sub Form_Evaluate

        If SumOK = SumFail Then
                Total_Time.Status = (Total_Time.Status Or tfFldInvalid)
        End If

End Sub
```

5. ここで一度スクリプトをコンパイルし保存します。コンパイルでエラーが発生する場合はその内容を確認しスクリプトを修正してください。

6. スクリプトが保存できたら TeleForm Designer でフォームを保存します。その後、TeleForm Reader、Verifier を起動し、「Image1.tif」と「Image2.tif」を使ってスクリプトを実行してみてください。
（各 tif ファイルは［03_演習用データ］－［Chapter 3 Exercise］にあります）

「Image1.tif」を評価すると、Verifier では、total_time フィールドにどんな値を入力しても不正となってしまいます。これはフォーム評価時に total_time.Status に tfFldInvalid が入ってしまったためです。

- 問題

スクリプトを完全なものとするため、total_time フィールドからフォーカスが外れる時に値を再度確認し、問題なければ Status を FldOK、問題があればメッセージを表示し再度入力を促す、ということをしたいとします。どのエントリー ポイントを使用し、どのようなロジックを実装すべきか考えてみてください。

（回答は［03_演習用データ］－［Chapter 3 Exercise］の「Answer.txt」にあります。）

※ スクリプトの完成版は［03_演習用データ］－［Chapter 3 Exercise］の「Script.txt」です。

第4章　エクスポート スクリプト

この章では以下のことを学びます。

- エントリー ポイント
- Export クラス
- エクスポート スクリプトの実行

4-1. エクスポート スクリプトの開き方

エクスポート スクリプトを作成、編集、コンパイルをするには VBA エディタを使用しなければいけません。このスクリプト エディタは TeleForm Designer で「スクリプトの編集」ウィンドウを開くことで初期化されます。

新規エクスポート スクリプトを作成するには
1. TeleForm Designer を起動します。
2. 「ユーティリティ」メニューから「エクスポート/システムスクリプト」をクリックします。
3. VBA Script Editor が起動したら、「ファイル (File) / 新規 (New...) 」をクリックします。
4. プロジェクト タイプにエクスポート スクリプトを選択し、プロジェクトの名前を入力します。
5. OK をクリックします。

既存のエクスポート スクリプトを開くには
1. TeleForm Designer を起動します。
2. 「ユーティリティ」メニューから「エクスポート/システムスクリプト」をクリックします。
「VBA Script Editor」が開きます。

注意:
エクスポート スクリプトの編集は Designer から起動した「VBA Script Editor」からのみ行ってください。別のアプリケーションでスクリプトを編集した場合、保存が行えません。

4-2. エントリー ポイント

エクスポート スクリプトのエントリー ポイントは、TeleForm がエクスポート スクリプトをどこで実行するかを示しています。

付録『プロジェクト エントリー ポイント イベントのダイアグラム』に、各エクスポート スクリプトのエントリー ポイントが TeleForm のデータ フロー上、いつ呼ばれるかを示しています。

各エントリー ポイントは以下のようになっています。

エントリー ポイント	説明
Sub Export_Setup : End Sub	このエントリー ポイントは TeleForm Designer の「自動エクスポート設定」で「保存」を選択した際に呼ばれます。また、このエントリー ポイントはエクスポート スクリプトを保存する際や「編集機能」において、「構成ダイアログのサポート」が選択されている場合のみ呼ばれます。
Sub Export_Record : End Sub	このエントリー ポイントは、各エクスポート レコードごとに一度呼ばれます。フォーム上の全てのエクスポート フィールドにアクセスすることができます。通常はこのエントリーで各フィールドのデータ ファイル書き込みなどを行います。

4-3. エクスポート スクリプト クラスとプロパティ

4-3-1. Export オブジェクト

唯一エクスポート スクリプトだけが Export オブジェクトを使用することができます。このオブジェクトはエクスポート セッションに関する情報を保持しています。

- Export クラス情報の参照

Export クラス情報を参照するには以下のようにします。

```
Export.ExportPropertyName
```

ExportPropertyName は Export クラスの適切なプロパティを指しています。例えば以下のようになります。

```
Export.Path
```

Export オブジェクトのプロパティは以下のように定義されています。

プロパティ	型	アクセス	説明
Path	String	Read-Write (Export_Setup では Write のみ)	データ ファイルのフル パス。これはスクリプトがフィールド データを書き込むファイルです。
Header	Boolean	Read Only	エクスポートのヘッダのステータス。 True：「ヘッダを含める」を選択 False：「ヘッダを含める」を未選択
Result	tfExportResult enum	Read-Write	レコードのエクスポートの結果を格納。この値は、各 Export_Record の最後で TeleForm に返ります。 エクスポートに失敗した場合は **tfExportError** を、成功した場合は **tfExportSuccess** を格納します。
Master	Export	Read Only	処理中のエクスポート処理が詳細グループのネストされたエクスポート処理である場合、マスターtfExport オブジェクトが返ります。詳細グループの各列は、エクスポート処理時は別々のレコードとして扱われます。 このプロパティの値は： ・ フォーム レベルのエクスポートの場合は Nothing ・ 詳細グループの場合は値
Fields	TfFields Object	Read Only	フォームのエクスポート可能な全フィールドのコレクションを返します。
EnableImageTab	Boolean	Read-Write	Export_Setup でイメージ タブを有効にする。このプロパティはイメージのエクスポートがサポートされているときのみ使用できます。
Form	tfForm Object	Read Only	エクスポートされる tfForm オブジェクト。Title, FormID, Mode, Image, Status, CurField, CurGroup, Fields プロパティを含みます。
Extra	Long	Read-Write	エクスポート フラグ（未実装）

- 詳細グループのエクスポート

エクスポート スクリプトを作成する際、通常詳細グループのための特別な処理は必要ありません。エクスポート スクリプトはマスター フォーム レコードの時と同様、詳細グループの各列に対して呼ばれます。

TeleForm はマスター　レコードよりも詳細グループを先に処理するため、固定されたファイル　ハンドルは使用すべきではありません。この場合、FeeFile() 関数を使用し、利用可能なファイル　ハンドルを取得してください。

もし詳細グループに対して特別な処理をしたい場合、Export.Master がマスター　エクスポート　オブジェクトを指し示していますので、これを利用してください。マスター　フォーム　レコードのエクスポート時は Export.Master は **Nothing** となっています。

4-3-2. Field クラス

エクスポート　スクリプトは Fields コレクションを通じて Field クラスへアクセスします。*FieldName* を使用することはできませんので注意してください。

4-4. エクスポート スクリプトの実行

エクスポート スクリプトを実行するには、まずは TeleForm Designer の「VBA Script Editor」でスクリプトをコンパイルし、保存をする必要があります。コンパイル エラーが発生した場合は、スクリプトの実行を行う前に、そのエラーを解決しなければいけません。

エクスポート スクリプトを実行するには以下の手順に従ってください。

1. 新規エクスポート スクリプトを作成し、TeleForm Designer を終了します。

重要:
「自動エクスポート設定」の形式リストにこの新しいエクスポート形式を登録するには TeleForm Desinger を再起動する必要があります。

2. TeleForm Designer を起動します。
3. フォームを開きます。
4. 「フォーム」メニューから「自動エクスポート設定」をクリックし、ダイアログ ボックスを表示します。
5. 「新規」ボタンをクリックします。
6. 「メイン」タブで「形式」から作成したエクスポート スクリプトを選択し「保存」をクリックします。
7. Export_Setup エントリー ポイントで指定したパスが表示されることを確認します。

注意:
「保存」ボタンをクリックしても何も起こらない場合、「構成ダイアログのサポート」をチェックしているにも関わらず Export_Setup エントリー ポイントにコードがありません。この場合は「構成ダイアログのサポート」のチェックを外してください。

8. 「有効」をチェックし「OK」ボタンをクリックください。
9. 「自動エクスポート設定」ダイアログ ボックスで、エクスポート パスと有効のチェックを確認してください。問題がなければ「OK」をクリックします。
10. フォームを保存します。
11. TeleForm Reader、Verifier を再起動します。
12. TeleForm Reader でフォーム イメージを評価します。
13. TeleForm Verifier でフォーム イメージを修正します。エクスポート処理は自動で実行されます。

※ 「ユーティリティ」メニューの「内部データエクスポート」を使用して手動エクスポートを実行することもできます。

4-5. 演習

- 準備

演習を始める前にまずは環境を準備します。以下の手順を行ってください。

1. TeleForm Designer で「Chapter 4 Exercise」を開いてください。
   (「Chapter 4 Exercise」は[02_フォーム]の「tf48673.tfw」です。)
2. 「ユーティリティ」メニューから「エクスポート/システムスクリプト」を選択して「スクリプトの編集」ウィンドウを開いてください。
3. 「ファイル」メニューから「新規」 → 「エクスポート スクリプト」を選んでください。

- 演習

「Chapter 4 Exercise」はあるお店の注文書です。顧客は欲しい商品の数量を書き込んで送付します。

今回の演習では、入力されたデータをファイルに出力するスクリプトを作成します。

1. Export_Setup エントリー ポイントに出力するデータ ファイルを指定します。以下のコードを入力してください。

```
Sub Export_Setup

        Export.Path = "C:¥Export.log"

End Sub
```

2. 複数のエントリー ポイント サブルーチンで使用する出力ファイルの FileSystemObject を宣言します。なお、「ツール / 参照設定」で MicroSoft Scripting Runtime を指定してください。

```
Dim fso As New FileSystemObject

Sub Export_Setup
```

3. ファイル出力を実装します。

```
Sub Export_Record

    Dim exportFile As TextStream
    Set exportFile = fso.OpenTextFile(Export.Path, ForAppending, True)

    Dim s As String


    s = "エクスポートしました。"
    exportFile.WriteLine s

End Sub
```

4. 「ファイル」メニューから「名前を付けて保存」をクリックします。ダイアログで「エクスポート形式名」を「Chapter 4 Exercise」とし、「構成ダイアログのサポート」にチェックをして「OK」を押します。

「4.4 エクスポート スクリプトの実行」を参照して、エクスポート スクリプトを実行してください。イメージ ファイルは［03_演習用データ］－［Chapter 4 Exercise］の「Image01.tif」を使用してください。

「C:¥Export.log」が作成され、「エクスポートしました。」と書き込まれていることを確認します。

- 問題

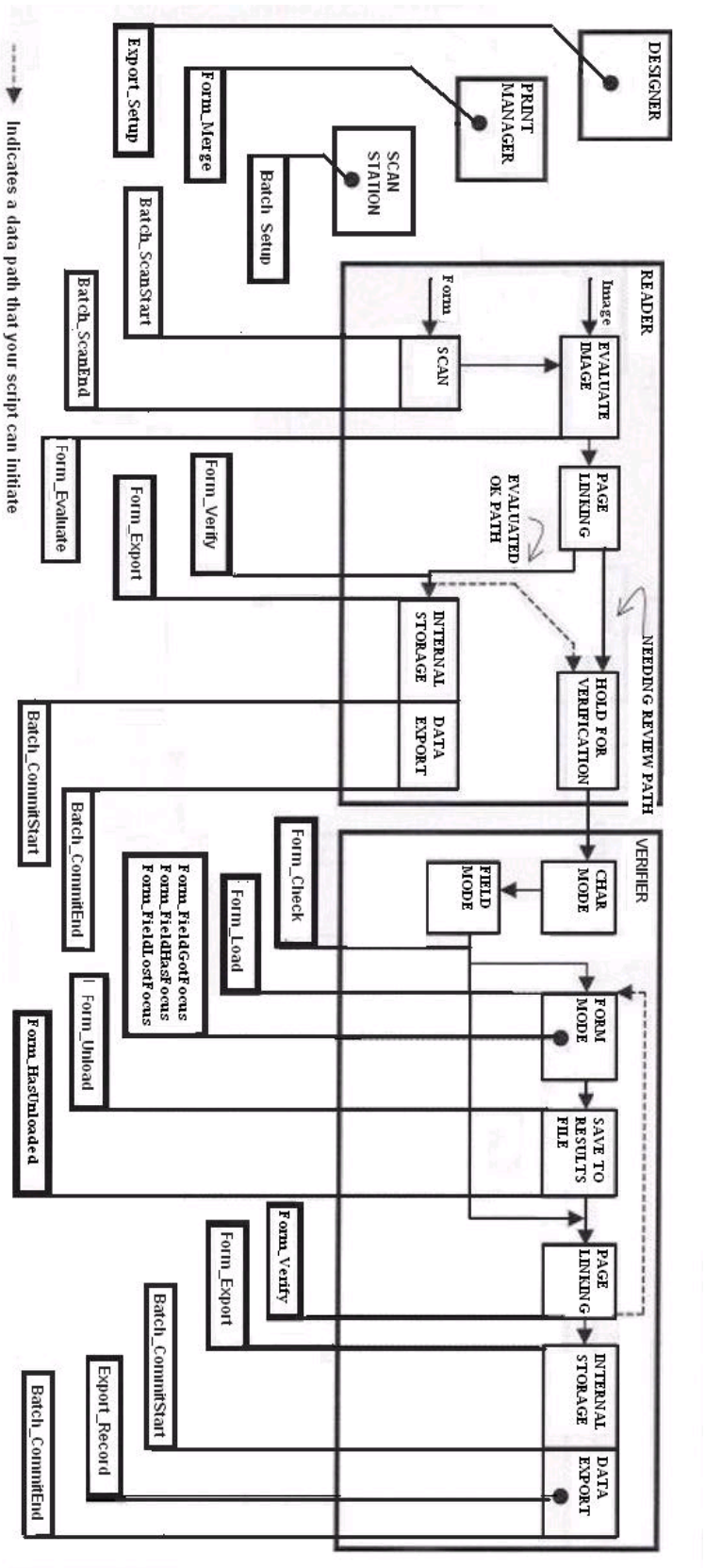
まず、「自動エクスポート設定」でエクスポートするフィールドを su_1、su_2、su_3 に限定します。このとき、これらのフィールドの値をファイルに出力するにはどうしたらよいでしょうか。

（回答は［03_演習用データ］－［Chapter 4 Exercise］の「Answer.txt」にあります。）

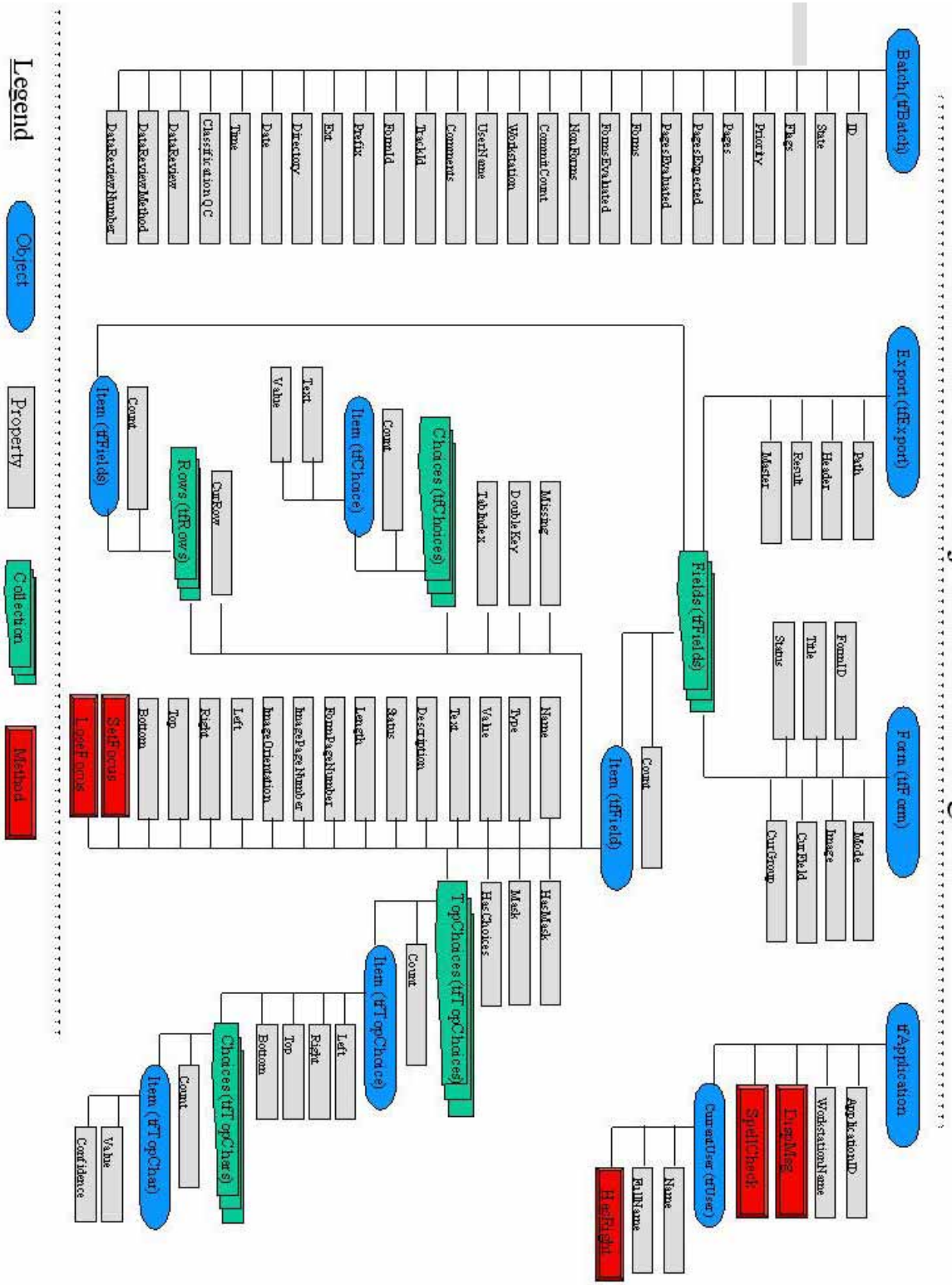
※ スクリプトの完成版は［03_演習用データ］－［Chapter 4 Exercise］の「Script.txt」です。

第5章　付録

・ プロジェクト エントリー ポイント イベントのダイアグラム



・ TeleForm VBA オブジェクト モデル ダイアグラム



プロジェクト エントリー ポイント イベントのダイアグラムの訂正

Reader 中の DATA EXPORT に Export_Record イベントが抜けている

TeleForm VBA オブジェクト モデル ダイアグラムの訂正

tfField オブジェクトの HasChoices プロパティは HasTopChoices の間違い

tfFields オブジェクトに Exists プロパティが抜けている