福井 和彦 / 清水 孝将

株式会社ミガロ.

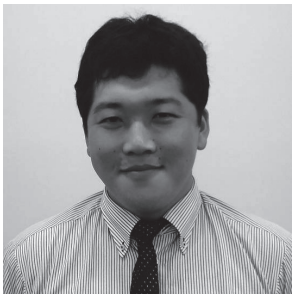
システム事業部システム3課　システム事業部システム2課

Delphi/400:Webからの PDF出力

Web アプリケーションで需要の高い
PDF でアウトプットを実現する方法を紹介する。

略歴　福井 和彦
1972 年 3 月 20 日生まれ
1994 年大阪電気通信大学工学部卒
2001 年 04 月株式会社ミガロ . 入社
2001 年 04 月システム事業部配属

現在の仕事内容
主に Delphi/400 を使用した受託開発で、要件確認から納品・フォローに至るまでのシステム開発全般に携わる。また、Delphi/400 の導入支援やセミナーの講師なども担当。



略歴　清水 孝将
1983 年 10 月 4 日生まれ
2008 年甲南大学文学部卒
2008 年株式会社ミガロ . 入社
2008 年 04 月システム事業部配属

現在の仕事内容
入社 3 年目で Delphi/400 や JC/400 の開発業務を担当。Web に関する知識や技術を身につけ、Web アプリケーションのスペシャリストを目指している。

はじめに

Delphi/400 の「VCL for the Web」を使用することで、クライアント / サーバー型の開発と同様の手法で Web アプリケーションの開発ができるため、特に照会系の画面開発については敷居が低くなったように思う。

本稿では、さらに Delphi/400 を使用した Web アプリケーション開発のバリエーションを広げていただけるよう、PDF 出力機能の実装について紹介していこう。

Webアプリケーションからの帳票出力

Web アプリケーションは Web サーバー上で実行されるため、印刷処理を行おうとしても、クライアントに接続されたプリンタから出力することができない。

また、HTML 文書はクライアントの OS やブラウザ、出力用紙サイズといったさまざまな要因によって、レイアウトが変わることがある。

このような要因から、Web アプリケーションは帳票出力が得意ではなく、実装方法に悩まれている方も少なくないだろう。

そこで今回、Web アプリケーションで PDF 形式の帳票を作成し、その結果をブラウザ上で確認できる仕組みを紹介する。この手法を見ていただくことで、Web アプリケーションへ帳票出力機能を実装する際の参考になれば幸いである。

PDF出力機能の構成

Web アプリケーションから PDF 出力を行うための構成は、図の通りである。【図 1】

●ISAPI アプリケーション
Delphi/400 の「VCL for the Web」で作成する、一般的な Web アプリケーション。
（ISAPI：Internet Server Application Program Interface）

● CGI アプリケーション
Delphi/400 の「WebBroker」で作成する、一般的な Web アプリケーション。
（CGI：Common Gateway Interface）

処理の流れ

図を参照しながら、PDF 出力構成の流れを順番に見ていこう。【図 1】

① ISAPI アプリケーションは、ブラウザから PDF 出力のリクエストを受ける。
・ リクエストを受け付けるための指示画面は、Delphi/400 の VCL for the Web で開発する。

②リクエストを受けた ISAPI アプリケーションは、さらに CGI アプリケーションに対して PDF 出力のリクエストを行う。

③リクエストを受けた CGI アプリケーションは、Delphi/400 の機能を使用

図1

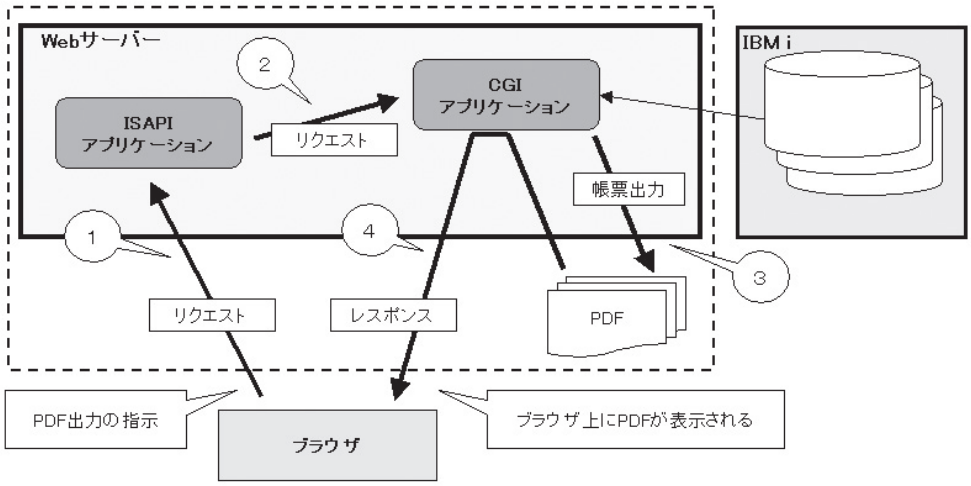


図2-1

図2-2

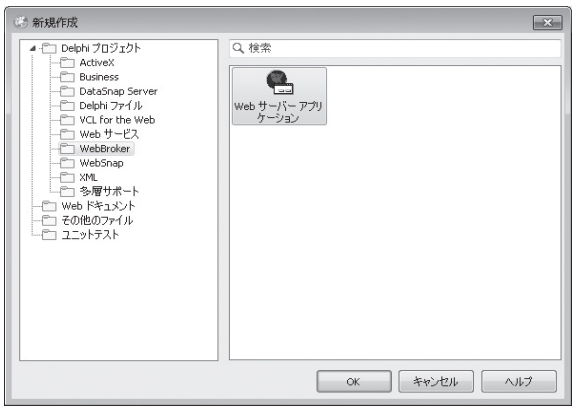


図2-3

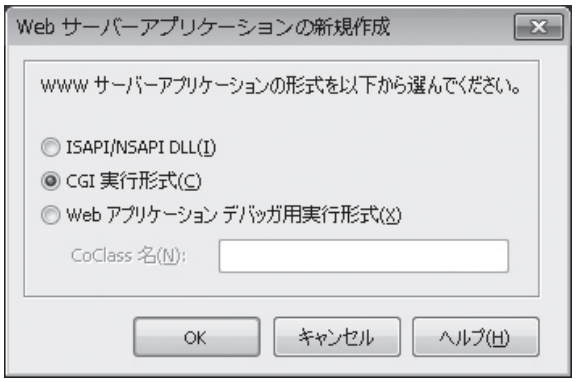
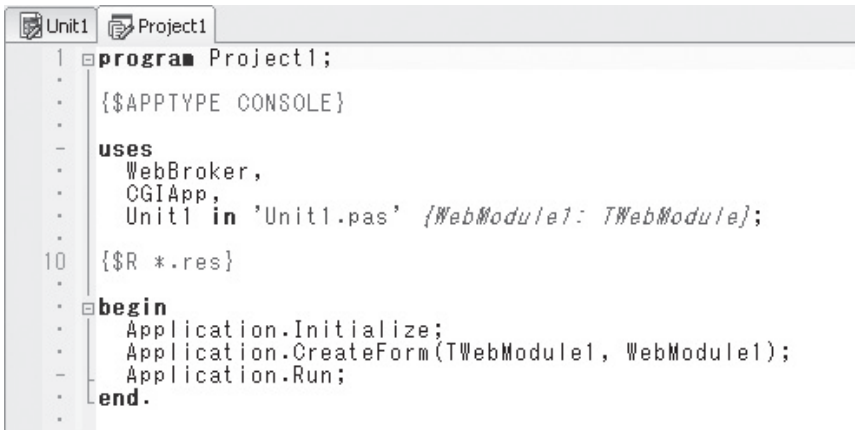


図2-4



して IBM i より必要なデータを取得し、PDF 形式の帳票を出力する。
・CGI アプリケーションは、Delphi/400 の WebBroker で開発する。
・PDF 出力を行うために「PowerPDF」を使用している（※）。

④CGI アプリケーションは、出力した PDF 形式の帳票をブラウザ上で参照できるようブラウザ側へ結果を返す。

※ PowerPDF については、本誌「Migaro.Technical Report 2010」に、帳票出力に関する論文が記載されている。そちらも参考にしていただきたい。

今回のポイント

今回の仕組みのポイントは、次の 2 点である。

● VCL for the Web での開発は画面周りに特化させ、PDF 出力の部分に関しては CGI アプリケーション側で実装し、その CGI アプリケーションを呼び出すようにしたこと。
● PDF 出力の部分を、Delphi/400 を使用して、CGI アプリケーションとして開発したこと。

このような構成にすることで、VCL for the Web の開発では、PDF 出力の仕組みを気にすることなく、画面の開発を行うことができる。

そして、CGI アプリケーションを Delphi/400 で開発することで、CGI アプリケーションとして Web アプリケーションの機能を持つことができ、さらにクライアント / サーバー型の開発手法による機能を実装した CGI アプリケーションを構築することができる。

従って、Delphi/400 で IBM i へ接続し、IBM i より取得したデータをもとに PowerPDF を使用して PDF 出力を行うといった一連の流れには、クライアント / サーバー型の開発手法をそのまま利用でき、これまで培ってきたノウハウを活用することができる。

開発手順

具体的な開発手順について紹介していく。開発手順の流れは、以下の通りである。

(1) CGI 実行形式アプリケーションの作成
(2) CGI アプリケーションの Actions の実装
(3) CGI アプリケーションと PowerPDF との連携
(4) ISAPI アプリケーションからの実行

(1) CGI 実行形式アプリケーションの作成

まずは、実際にどのようにして CGI アプリケーションを作成するのかを紹介したい。

Delphi では、WebBroker というフレームワークを利用して、CGI アプリケーションを容易に作成することが可能である。

手順としては［ファイル］→［その他］→［Delphi プロジェクト /WebBroker］→［Web サーバーアプリケーション］を選択する。「Web アプリケーションの新規作成」の Window が表示されるので、「CGI 実行形式」を選択すると、開発画面に遷移する。

開発画面には、Unit.pas（TWebModule）が自動で生成されている。このユニットに、CGI アプリケーションの処理を記述していくこととなる。【図 2】

(2) CGI アプリケーションの Actions の実装

● CGI アプリケーションの URL

図の URL は、CGI アプリケーションの URL である。

URL には、Pathinfo 部と Query 部というものが存在する。Pathinfo 部は CGI アプリケーションの処理の分岐に使われ、Query 部は CGI アプリケーションへ渡すパラメータの役割を持っている。【図 3】

WebBroker では、TWebModule が持つ Actions を設定することによって、Pathinfo ごとの処理を容易に分岐させることができる。

TWebModule が持つ Actions は、クライアント / サーバー型アプリケーションの Action と非常によく似た仕組みになっている。Actions は、オブジェクトインスペクタから、Actions のプロパティエディターで追加することができる。この Actions の OnAction イベントにコーディングすることで、CGI アプリケーションの処理を実装することができる。【図 4】

ここからは、実際のコーディングの説明を行う。

前述した通り、Web アプリケーションとは、ブラウザ（ユーザー）が Web サーバーに対してリクエスト（要求）を送り、Web サーバーがレスポンス（結果）をブラウザ（ユーザー）に返すということが大きな仕組みとなっている。

CGI アプリケーションがユーザー（ブラウザ）からのリクエストを受け取る時や、それとは逆にユーザー（ブラウザ）にレスポンスを返す場合のどちらも、TWebModule.Request プロパティで行う。アプリケーションに渡された、Pathinfo や Query の値はすべて Request プロパティ内で保持されている。

(3) CGI アプリケーションと PowerPDF との連携

では、帳票出力処理の実装の説明に入りたい。

最初に、PowerPDF で作成したテンプレート Form を、CGI アプリケーションのプロジェクトに追加する。続いて、この追加した Form を、CGI アプリケーション内部で生成し、帳票出力に利用する。

なお、PowerPDF の Form 自体は、クライアント / サーバー形式で作成する場合と同じものを使用することができるため、すでに作成済みのテンプレートがある場合などはそのまま流用することも可能である。

【コーディング処理の流れ】
①テンプレートファイルを生成
②リクエストからの情報 Query を取得
③パラメータをもとに SQL をデータベースに発行
④該当レコードをテンプレートファイルに転送
⑤PDF ファイルに出力（ファイルに保存）
⑥保存されたファイルの読込
⑦レスポンスに対して、読み込んだファイルと表示アプリケーションの指定を返却

この流れで連携処理を実装していく。流れの中で、WebBroker 特有のクラスやメソッドを利用する部分は①と⑦の部

図3

図4-1

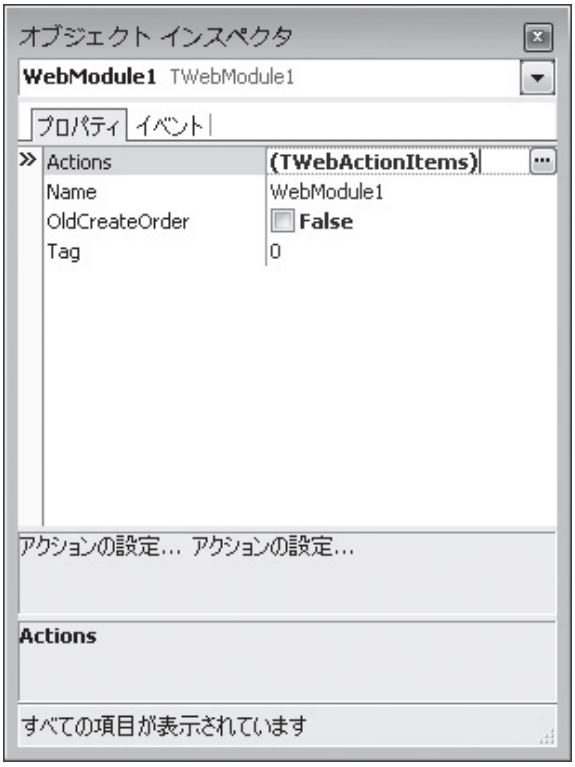


図4-2

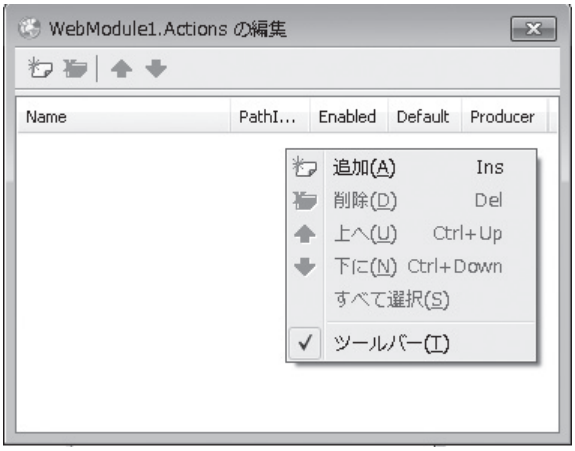
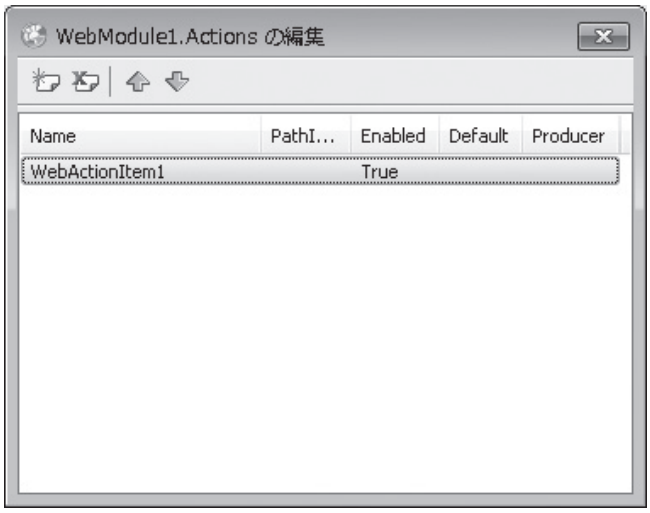


図4-3



分のみで、他の部分はクライアント/サーバー形式のコーディングと全く同じである。

詳しい記述は、ソースを見ていただきたい。【ソース1】

このように作成されたCGIアプリケーションを実行すると、画面のように、PDFが表示される。【図5】

● CGIアプリケーションへのアクセス

作成したCGIアプリケーションにブラウザからアクセスする際は、以下のようなURLでアクセス可能である。

URLの構成については、前述の図を参照してほしい。【ソース2】

http:// サーバー名 / スクリプト名 /Path info?Query=＊＊＊＊＊

(4) ISAPIアプリケーションからの実行

ISAPIアプリケーションから呼び出す場合は、NewWindowメソッドを使用してCGIアプリケーションを起動する。【図7】

Pathinfoの値やQueryの値を呼出時に変えることによって、CGIアプリケーションの処理を分岐させることができる。

ソース例に示すように、CGIアプリケーションのURLで「ESTCODE=0011」の「0011」の部分を見積番号とした場合、CGIアプリケーションを呼び出す際には、「0011」の部分にISAPIアプリケーションで指定された見積番号を指定することで、対象の見積書を出力することができる。

セキュリティ管理

PDFやExcelファイルなどをそのままブラウザ上で表示するためには、Webサーバーの公開された範囲にファイルを配置しなくてはいけない。だが、実際に機密情報や個人情報などが記載されているようなファイルを、Webサーバー上に置くことは不可能である。

しかし、CGIアプリケーションを介した場合は、Webサーバー上で公開されていないファイルに対してアクセスすることが可能となる。CGI内部に、ユーザー認証の処理を組み込めば、限定されたユーザーに対してPDF表示を実現することができる。

最後に

Webアプリケーションを構築する上で悩みの種であった帳票出力を実装する方法について、その解決策の1つとして、今回「ISAPIアプリケーション」+「CGIアプリケーション」の構成でPDF出力結果をブラウザへ表示するという仕組みを紹介してきた。

ここまでご覧いただいて、Delphi/400でCGIアプリケーションを開発し、VCL for the Webで開発したISAPIアプリケーションと連携させることで、PDFの出力が簡単に行えることがお分かりいただけたであろう。

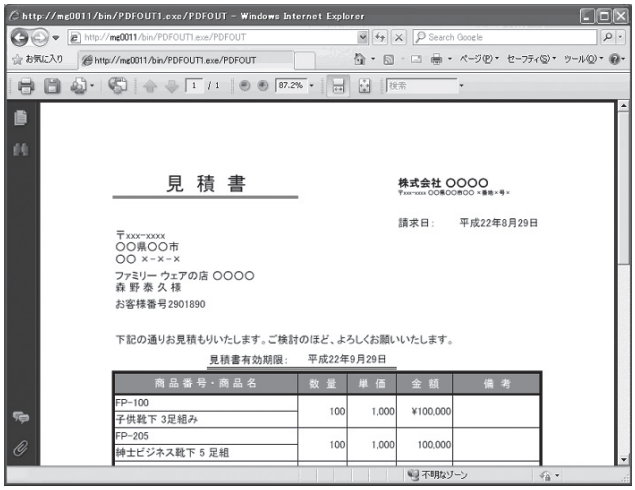
今回紹介した例はPDF出力であったが、CGIアプリケーションの作り方次第では、Excel形式への出力やCSV形式への出力を行うことも可能であるので、PDF出力以外の方法についても、ぜひともチャレンジしていただきたい。今回の内容が、Webアプリケーションからの帳票出力方法の参考になれば幸いである。

M

ソース1

```
procedure TWebModule1.WebModule1WebActionItem1Action(Sender: TObject;
  Request: TWebRequest; Response: TWebResponse; var Handled: Boolean);
var
  sFileName :String;
  sFileStream: TFileStream;

begin
  // 1、 テンプレートファイルを生成
  frmPrintPDF :=TfrmPrintPDF.Create(nil);

  try
    // 2、 リクエストからの情報Queryを取得
    SQLQuery1.ParamByName('ESTCODE').AsString :=
                          Request.QueryFields.Values['ESTCODE'];
    // 3、 パラメータを元にSQLをデータベースに発行
    SQLQuery1.Open;

    //4、 該当レコードをテンプレートファイルに転送
    with frmPrintPDF do
    begin
      //******ここでPDFテンプレートへの転送を記述****START***



      //******ここでPDFテンプレートへの転送を記述****END*****
      PReport1.FileName := sFileName;
      //5、 PDFファイルに出力（ファイルに保存）
      PReport1.BeginDoc;
      try
        PReport1.Print(PRPage1);
      finally
        PReport1.EndDoc;
      end;

      //6、 保存されたファイルを読込み
      sFileStream := TFileStream.Create(sFileName, fmOpenRead);
      try
      //7、レスポンスにたいして、読込んだファイルと表示アプリケーションの指定を返却
        Response.ContentType := 'application/pdf';
        Response.ContentStream := sFileStream;
        Response.SendResponse;
      finally
        sFileStream.Free;
      end;
    end;
  finally
    frmPrintPDF.Release;
  end;
end;
```

図5



ソース2

```
procedure TIWForm1.IWButton1Click(Sender: TObject);
begin
  WebApplication.NewWindow('http://mg0011/bin/PDFOUT1.exe/PDFOUT?ESTCODE=0011');
end;
```