by Jay Oswal

No more theories. No more practice samples. No more incomplete code. Here is a business-quality RPG Web program for file maintenance that actually works in the live production environment.

Most RPG programmers would love to deploy their interactive RPG applications via an Internet browser without sacrificing their hard-earned RPG skills, wasting years to learn some strange language, or navigating torturous learning curves. Figure 1a shows an actual file maintenance RPG program in production. This program can serve as a template for other programs and inspire RPG programmers who want to replace the 5250 screen with a Web browser but don’t know where to start.

The program, VM (Vehicle Maintenance), was written entirely in ILE RPG and performs common file maintenance functions (e.g., display, modify, insert, delete) against the Vehicle Master file (VEHMST). It uses two HTML documents — VMLST and VMDTL — as the user interface, as well as free APIs from IBM’s CGIDEV2 Toolkit. The interface uses the same theme as a traditional subfile and detail maintenance screens, and includes all error and completion messages. All of the objects used by this application reside on your iSeries, and users simply use an Internet browser to access them from their PCs on a shared TCP/IP network.

Although a good knowledge of HTML and JavaScript will definitely help you develop more advanced Web applications, you can get started now with just basic knowledge. You can learn from the thousands of HTML and JavaScript examples on the Internet; however, the best way to learn application development is to install a useful business-quality application that actually works, look at the result it produces, investigate the code that made it all possible, and then clone it to suit your particular needs.

This article explains how an ILE RPG program on the iSeries (VM in this case) can interface with the users via an Internet browser. This application lists the records from Vehicle Master file, then lets users not only display, add, change, and delete the records, but page up, page down, refresh, and position them as well.

It is crucial that you load and run the application before trying to understand how it works, and that you read the entire article before actually performing the instructions. Total estimated time to implement the application is two hours. I created VM on an iSeries running OS/400 V5R3M0.

Über den Autor Jay Oswal is a system administrator and senior staff business systems analyst at TOTAL E&P USA, Inc. E-mail him at joswal@sbcglobal.net.

Programm Requirements

Before you download the programs in this article to your production environment, be sure you met the following requirements:

- an active license for RPG ILE compiler on your iSeries (5722-WDS)
- a minimum OS/400 release of V5R1M0
- IBM Licensed program 5722-DGI (comes with OS/400)
- good experience in ILE RPG programming
- CGIDEV2 Toolkit — free from IBM
- basic knowledge of HTML and JavaScript
- basic knowledge of FTP commands

The CGIDEV2 Toolkit creates an HTTP server names CGIDEV2APA on your iSeries. This article uses HOUS0201 as the TCP/IP name for the iSeries, port number 8030 for the HTTP server CGIDEV2APA, and CGIPGMR as both the source library and the production library. If you wish to use different values, substitute these values with your own throughout in this article as well as in the source members. This article also uses /CGIPGMR as the directory to hold graphical images of the vehicles. If you decide to use some other directory, then you must correctly specify the image path in the VHIMG field of the VEHMST file. It’s also possible to store images in more than one directory, as long as you specify their corresponding

For a complete list of application requirements, see „Program Requirements“ on page ProVIP 18.

paths correctly in the VHIMG field.

Getting Started

2006 Oktober2 Utility der NEWSolutions	
Load´n´go	• Load n Go Programm: A Business-Quality RPG Web Application by Jay Oswal

When you’re ready to begin, you’ll need to perform the following four steps.

Step 1. Install the CGIDEV2 Toolkit. Go to the IBM Web site and download the CGIDEV2 Toolkit (www-03.ibm.com/servers/eserver/services/assets/ebus_rpgcgitoolkit.html). Follow the instructions to load the CGIDEV2 Toolkit as well as to configure the HTTP server CGIDEV2APA to run your Web applications. **Step 2. Deploy the application.** This article provides four source members, a sample data file, and vehicle images:

- VEHMST(qddssrc.vehmst) — DDS source for a Vehicle Master physical file
- VEHMST(vehmst.vehmst) — Vehicle Master sample data file
- Files *.jpg and *.gif — Vehicle images referenced by VEHMST
- VMLST(htmlsrc.vmlst) — HTML source for the Vehicle Maintenance subfile
- VMDTL(htmlsrc.vmdtl) — HTML source for Vehicle Maintenance record details
- VM(qrpglesrc.vm) — ILE RPG source for the Vehicle Maintenance program

Download all of these files to a folder (e.g., c:\CGIDEV2) on your PC. Upload the following four files to library CGIPGMR using the FTP commands in Figure 2 (the record length of the HTMLSRC source file is 240 bytes):

- Upload qddssrc.vehmst to CGIPGMR/QDDSSRC .VEHMST.
- Upload htmlsrc.vmlst to CGIPGMR/HTMLSRC .VMLST.
- Upload htmlsrc.vmdtl to CGIPGMR/HTMLSRC .VMLST.
- Upload qrpglesrc.vm to CGIPGMR/QRPGLESRC .VM.

Next, run command ADDLIBLE CGIPGMR. Then, using WDSc or PDM, set the member types for the uploaded source members as follows:

    - VEHMST PF

- VM RPGLE

- VMDTL HTML

- VMLST HTML

Künstler Burgy Zapp

Compile the source member VEHMST to create an empty physical file VEHMST in library CGIPGMR (this is required for the next three steps). Create the RPG program VM using command

```
CRTBNDRPG PGM(CGIPGMR/VM)
SRCFILE(CGIPGMR/QRPGLESRC) SRCMBR(VM)
DFTACTGRP(*NO) ACTGRP(VM)       AUT(*USE)
```

Upload vehmst.vehmst to library CGIPGMR. It will replace the empty member of VEHMST with the sample data (Figure 3). Next, upload all the images (files with .gif and .jpg suffices) to IFS directory /cgipgmr (Figure 4). HTML members are not compiled.

Step 3. Run the application. Open your Internet browser, type http://youras400:8030/cgipgmrp/vm.pgm into the address window, and press Enter. Doesn't it look like a typical iSeries file maintenance program? Feel free to navigate all the windows by using the buttons; it's the sample data anyway. Here is an idea: Back up the VEHMST to VEHMST.BKP, run command CLRPFM CGIPGMR/VEHMST to empty the file, and test the program to see the difference in the buttons. Then, copy the backup file back to the original file. Make sure that you try every button and input field to understand the functionality of the program. Let's look at the application details next.

Step 4. Understand the application. By now you must realize that, as an RPG programmer, you have already written many programs such as this one using the 5250 display as the user interface. You need to learn only a handful of new items to start developing your next RPG program using the Internet browser as the user interface. Figure 5 shows a comparison between 5250 and Web programming in ILE RPG. It is important that you remember this comparison while dissecting the program to see how it works.

In summary, this is what happens:

- The user types http://youras400:8030/cgipgmrp/vm .pgm in the address box of an Internet browser window to call the program VM that resides at a location referenced by cgipmrp on „youras400“ system, using the transport services of an HTTP server running at port 8030.
- Upon receiving the user request via the HTTP server, OS/400 starts program VM in a job.
- Program VM retrieves input data from the HTML fields and populates the internal program variables (remember that HTML can handle only character variables).
- Based on the requested function, the program VM
    1. opens the copy of a suitable HTML member (e.g., VMLST to display a list or VMDTL to handle the record details)
    2. prepares the required response using RPG logic
    3. populates the HTML output fields (similar to DDS fields)
    4. writes the required HTML sections (similar to DDS records in a display file)
    5. hands over the finished HTML document to the HTTP server
    6. ends, along with the job it was running in
- When the user responds by clicking a button, the cycle repeats.

Application Details

Now let’s look at how the RPG program VM uses HTML documents VMLST and VMDTL as the new Web interface. The best way to learn the application details is to keep the following items open simultaneously and follow the explanation.

- Open the application by typing http://youras400:8030/cgipgmrp /vm.pgm into the browser’s address field.
- Open the HTML source members VMLST and VMDTL in Notepad.
- Open the RPG source member VM in Notepad.

Opening the source members in Notepad helps you scroll the code easily and enables you to make your own notes within the source members without affecting the originals. The Initial List — HTML Source Member VMLST Entering http://youras400:8030 /cgipgmrp/vm.pgm in the browser’s address field causes the HTTP server CGIDEV2APA to start the RPG program VM on the iSeries. The program uses several copy members to gain access to the required HTML APIs. For the sake of simplicity, I use two different data structures to hold the input and output fields named as such. The program VM overrides the VEHMST file to the one in library CGIPGMR because the HTTP server does not recognize the user’s library list. The program uses the ZHBGETVAR API to copy the contents of input variables from the HTML form into the input variables of the program. Unfortunately, the first time around there is no HTML form in the browser. Therefore, all the input variables are blank, and the subsequent Select statement executes subroutines BGNLST and REFRESH. The BGNLST subroutine uses the GETHTML API to access the HTML source member VMLST, determines the number of rows to display, updates two HTML fields using the UPDHTML VAR API, and copies the HTML section /$top into the program buffer using the WRTSECTION API. The REFRESH subroutine builds a list of VEHMST records starting at the first record in the file and updates the required output fields. It uses the subroutine WRTTABROW to build the list by repeatedly writing one HTML table row at a time into the buffer. The program compiles a Web page by sequentially writing the required sections, similar to writing DDS records to build a 5250 screen. It also writes the end part of the HTML table using the subroutine WRTTABEND. Finally, using the subroutine UPDPGMCTL, the program updates the remaining HTML output variables, writes the /$hidden section of VMLST to preserve the hidden variables in the Web page, writes the *fini section, closes the data file, and ends — losing all the information in its buffer. Let’s take a closer look at what’s happening here. The HTML section /$hidden is a secret between your RPG program and the HTML page in the browser. This is where you store the information necessary to run the next instance of your program whenever the user decides to make the next request. When a user clicks a SUBMIT button to submit the HTML form described in the browser page, VM starts again. The previously stored information is read back into your program via the GET HTMLVAR API for successive processing — for example, to page up or down, or to decide where to position, display, or modify the list. The section *fini is mandatory; it tells the WRTSECTION API to actually send the compiled Web page to the HTTP server. This in turn transmits the page to the user's browser, which converts it into a graphical Web page.

Detail Functions — HTML Source Member VMDTL