IMS Web Programmer's Reference

webp-0002-01

March 10, 1998

Candace Garcia
IBM Corporation

IMS Web Programmer's Reference

webp-0002-01

Contents

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

AIX	IBM
IMS	Net.Data
OS/2	OS/390
RACF	

Windows and Windows NT are registered trademarks of Microsoft Corporation.

Other company, product, and service names may be trademarks or service marks of others.

Chapter 1. Enabling IMS transactions on the Web

The following sections outline the development process that you follow to enable an IMS transaction to the Web, using IMS Web either with or without Net.Data.

Using IMS Web without Net.Data

1. On the host, locate the MFS source for the transaction.
2. Follow the steps in the IMS Web Studio TaskGuide to download the MFS source to your workstation, and to generate C++, HTML files and .mak files for your transaction. The target platform for generation should be that of your Web server. These files are generated in the cgi subdirectory of the *proj* directory.
3. Use the nmake (NT or OS/2) or make (AIX, or SUN, or OS/390) utility and generated .mak file, CGI*proj*.mak, to build the executable form of the CGI-BIN program from the generated C++ source files.
4. Move the CGI-BIN executable and the generated input HTML form to the appropriate directories on your Web server.
5. From a browser, invoke the input HTML form, input the IMS transaction data, and then select the SUBMIT push button on the HTML form in order to run the transaction and receive its output.

Using IMS Web and Net.Data

1. On the host, locate the MFS source for the transaction.
2. Follow the steps in the IMS Web Studio TaskGuide to download the MFS source to your workstation, and to generate C++ source files, a Net.Data Web macro, a module definition file, and a .mak file for your transaction. The target platform for generation should be that of your Web server. These files are generated in the dtw subdirectory of the *proj* directory.
3. Use the `nmake` (NT or OS/2) or `make` (AIX, SUN, or OS/390) utility and generated .mak file, DTW*proj*.mak, to build the transaction DLL from the generated C++ source files. The platform of the transaction DLL must be the same as that of your Web server (the target platform).
4. Move the transaction DLL (DTW*proj*.dll) and Net.Data Web macro (DTW*proj*.mac) to your Web server workstation.
   - Place the Net.Data macro in a directory from which Net.Data retrieves web macros. (See the MACRO_PATH statement in “Configuring Net.Data” in the *Net.Data Administration and Programming Guide*.)
   - Place the transaction DLL in a directory from which the Web server operating system retrieves DLLs or shared libraries.
5. Use the link in the sample file DTW*proj*.htm to modify an HTML file in your Web server HTML tree. You can then use the link to invoke Net.Data and display the input HTML form for the transaction on a Web browser. Fill in the transaction input and select the SUBMIT push button on the form in order to run the transaction and receive its output at the Web browser.

Chapter 2. Executing IMS commands from a Web browser

You can use IMS Web to generate input HTML and the associated executable file to support the execution of an IMS command from a Web browser. The following simple MFS source defines an input message field, COMMAND, for entering the IMS command.

```
CMDIN    MSG   TYPE=INPUT,SOR=(DUMMY)
         MFLD  COMMAND,LTH=119
         MSGEND
```

After the input HTML or Net.Data macro and associated executable file generated from the above source have been moved to the Web server, you can use the COMMAND entry field to enter an IMS command.

The output from the IMS command is returned to the Web browser using one of the default transaction output classes that are included with every project.

The RACF userid and group name used by an IMS project must have RACF authorization to execute a given IMS command in order for that command to be executable through IMS Web.

Chapter 3. Customizing generated code

Because IMS Web does not use information in the MFS FMT statement that is associated with a transaction, the literal information that identifies fields on the original 3270 screen is not available. The appearance of the generated HTML differs from the 3270 display.

This page describes how you can customize your code to make the generated HTML more closely resemble the 3270 display. Customization can be used in general to tailor the generated input and output HTML to the requirements of your environment.

Customization varies, depending on which set of generated files you are going to use: those generated for a basic IMS Web environment (in the *project_name*\cgi directory) or those generated for a Net.Data environment (in the *project_name*\dtw directory).

Customizing the input HTML form

This section describes:

Customizing output HTML

This section describes:

Customizing the input HTML form

This section describes customizing the input HTML form.

How input HTML is generated

The input HTML is generated based on the message input descriptor of the MFS source. For nonconversational IMS Web applications, the input HTML is a separately generated file. For nonconversational Net.Data applications, it is a %HTML block in the generated Net.Data macro.

HTML consists of an HTML form with the fields of the input message (MFLDs) arranged as text entry fields of an HTML table. MAXLENGTH is the defined length of the MFLD (that is, the LTH= value). The rows of the table are arranged in the same order as the MFLD statements within the message input descriptor, with the following exceptions:

- If a field has associated attribute bytes, they are not included in the text entry field. (MAXLENGTH is reduced by the number of attribute bytes.)
- Field literals are not displayed as part of the input HTML.

  Example: For the following MFS source input message field, the corresponding HTML would not contain a text entry field for M0001.

```
...
   M0001    MFLD 'Test data'
...
```

- Filler fields are not displayed as part of the input HTML.

  Example: The following MFS source would not result in a text entry field on the input HTML form.

```
...
             MFLD LTH=6
...
```

For basic IMS Web applications, the POST method is used to transmit the user data, and the destination of the ACTION is the generated CGI-BIN executable on the Web server. IMS Web uses the path that you specify in IMS Web Studio to build the URL of the ACTION.

For Net.Data applications, the POST method references the name of the output %HTML block in the Net.Data macro.

To submit the contents of the form to IMS, you select the SUBMIT push button. To reset fields to null, you select the RESET push button.

How to modify the input HTML

If you are modifying the files generated for a basic IMS Web environment, you must edit the following input HTML file in the *project_name*\cgi directory:

- HTM*project_name*.HTM file

If you are modifying the files generated for a Net.Data environment, you must edit the input HTML block of the following Net.Data macro in the *project_name*\dtw directory:

- DTW*project_name*.MAC

The name of the input HTML block is identified by the Net.Data statement %HTML *project_name*_InHTML).

You can tailor the input HTML for your environment, keeping in mind what is described in the following sections.

Removing a text input field from the input HTML

Because the generated code still contains the code that processes the corresponding attribute, removing a text input field from the input HTML is the same as not providing any data for the field. For example, if the corresponding input MFLD specifies a default literal, the literal value is used in the input message field. If the corresponding input MFLD does not specify a default literal, the field is padded as specified in the MFS source.

Removing an input attribute

Removing an input attribute from the generated classes and CGI-BIN program or transaction DLL code is not recommended, as its removal could result in an incorrect input message going to the IMS application.

After you have modified the input HTML file or Net.Data macro, place it in the appropriate directory and link it from one of your Web pages.

Example of modified input HTML

For an example of a modified input HTML form, see “Modifying input HTML” on page 46 .

Customizing output HTML

This section describes customizing output HTML.

How the CGI-BIN program or transaction DLL is generated

Include files

The program includes interface files for all the classes that it uses:

- HTM*project-name*.hpp, the class that parses the input HTML string. This class is only included in CGI-BIN programs.
- HWS*input_lpage*.hpp, the class that corresponds to the single input LPAGE.
- HWS*output_lpage_1*.hpp, the class that corresponds to the first output LPAGE.
- ... .
- HWS*output_lpage_n*.hpp, the class that corresponds to the nth output LPAGE.

In addition, the program contains an include for HWSUTIL.hpp, the interface to an IMS Web utility class.

Program logic

- For basic IMS Web applications:
  - Receive the input HTML string (via CIN).
  - Pass the input HTML string to the constructor of the parse class to instantiate an object of that class. The object will have attributes corresponding to the input message fields. The constructor parses the HTML string and uses the values obtained as its attribute values.
  - Set the attributes of the input LPAGE object (transaction input object) from:
    - The values specified to IMS Web Studio (RACF user ID, and so forth).
    - The attribute values of the parse object. (These values were input via the Web browser in the input HTML form.)
  - Invoke the execute method of the input LPAGE object to:
    - Build the input message.
    - Pass the input message to the IMS application program for processing.
    - Return the output message to the program as transaction output objects.
  - Invoke the genHTML method for each transaction output object returned from execute to build output HTML for the output logical pages.

  - Send the output HTML buffer to the Web browser (via COUT).
- For Net.Data applications:

  Net.Data's IMS Web language environment passes a list of the values entered in the input %HTML block of the Net.Data to the transaction DLL.
  - Set the attributes of the input LPAGE object (transaction input object) from:
    - The values specified to IMS Web Studio (RACF user ID, and so forth)
    - The values of the list passed by the IMS Web language environment
  - Invoke the execute method of the input LPAGE object to:
    - Build the input message
    - Pass the input message to the IMS application program for processing
    - Return the output message to the program as transaction output objects

  The transaction DLL returns the transaction output to the IMS Web language environment as a list of values. The language environment builds a Net.Data table from the list, which is presented in the output %HTML block of the Net.Data macro.

How the output HTML is generated

For basic IMS Web applications the output HTML is built dynamically when the CGI-BIN program invokes the genHTML method of a transaction output object. The HTML is built in a buffer that is output to standard output using the COUT statement.

For Net.Data applications, the output HTML is statically generated in the %REPORT block of the Net.Data macro. The %REPORT block contains HTML for displaying a Net.Data table. The table is built by the IMS Web language environment from the output of the IMS transaction.

In both cases, each output logical page is represented as an HTML table. <TR><TD><TD> entries are included for each output message field.

The output HTML is generated based on the message output descriptor of the MFS source. Each <TR><TD><TD> set in the table corresponds to a field of the output message (MFLD). The first cell of the row is set to the name of the output message field and the second cell of the row is set to the value of the output message field. The <TR><TD><TD> sets are arranged in the HTML table in the same order as the MFLD statements within the message output descriptor, with the following exceptions:

- If a field has associated attribute bytes, they are not included in the definiton list. Only the data portion of the output message field is represented by a <TR><TD><TD> set.
- A system control area field (SCA) is not included in the table.
- Filler fields are not displayed as part of the output HTML.

  Example: The following MFS source would not result in a field on the output HTML form:

```
...
                    MFLD LTH=6
...
```

All defined output fields are included in the table, even fields of a null or short segment.

How to modify the output HTML

Modifying files generated for a basic IMS Web environment

If you are modifying the files generated for a basic IMS Web environment you must modify the generated code that builds the buffer of output HTML. You can modify the generated code in either of the following ways:

- Modify the genHTML method.
- Replace the invocation of genHTML in the CGI-BIN program with your own code that uses the genList method to access the fields of the output method.

  genHTML and genList are methods of the LPAGE of an output message. If an output message consists of multiple LPAGEs, genHTML and genList methods exist for each output LPAGE, each tailored to build HTML appropriate to that LPAGE. Modify the methods for each output LPAGE that the IMS application can return.

Removing an output field from the HTML of genHTML

Because the generated code still contains the code that processes the corresponding attribute, do not remove any of the code that iterates through the list of output attributes.

After you have modified the code that builds the buffer of output HTML, rebuild the executable CGI-BIN program and place the executable in the appropriate directory on the Web server.

Modifying files generated for a Net.Data environment

If you are modifying the files generated for a Net.Data environment you must edit the output HTML block of the following Net.Data macro in the *project_name*\dtw directory:

- DTW*project_name*.MAC

  The name of the output HTML block is identified by the Net.Data statement %HTML (*project_name*_OutTML).

Example of modified output HTML

For an example of a modified output HTML form, see“Modifying output HTML” on page 48 .

Chapter 4. Using generated code in non-Web applications

A key feature of IMS Web is the generation of C++ classes for use in the execution of IMS transactions from the workstation. You can use these classes, along with the IMS Web Runtime component, either in a Web environment (in a CGI-BIN program) or in a non-Web environment. For example, in a non-Web environment you can use the classes in a GUI application, replacing the CGI-BIN program with your own GUI application and implementing the displayLpg method for output objects. To make this substitution, perform the following actions:

1. Discard the HTM*project_name*.htm, HTM*project_name*.hpp, and HTM*project_name*.cpp files, as these files are related to gathering input from the Web environment.
2. Study the IMS Web Studio-generated CGI *project_name*.cpp file to understand how the execute method of the class generated for the input LPAGE is invoked and how the output from the execute method is handled.
3. Modify the IMS Web Studio-generated CGI *project_name*.cpp file to gather input from a source other than the Web environment. This might involve providing a graphical window that accepts input from the user.
4. Modify the IMS Web Studio-generated CGI *project_name*.cpp file to display the transaction output in a form other than output HTML. This might involve providing a graphical window, and would involve the use of the genList method or your own implementation of the displayLpg method, rather than the genHTML method, to access and display the output message fields.
5. Update the CGI *project_name*.mak file to build your new non-Web application, removing references to the discarded files.

The following sections describe the IMS Web classes and how they can be used to build a non-Web application.

IMS Web transaction class hierarchy

Each IMS transaction is represented by a set of classes that are derived from the following two classes that IMS Web supplies:

HWSTranIn
: This class and its subclasses represent the input to your IMS transaction.

HWSTranOut
: This class and its subclasses represent the output from your IMS transaction.

HWSTranIn and HWSTranOut encapsulate data and functionality that are common to *all* IMS transactions. Their subclasses, which are generated by IMS Web, encapsulate data and functionality that is specific to *your* transaction.

Non-Web application program overview

To help you understand the information in this section, see “Chapter 5. IMS Web classes” on page 17. ″IMS Web classes″ presents the class hierarchy and describes, in detail, the classes that IMS Web supplies and generates.

The following logic makes your application responsible for obtaining the input that is necessary to run the transaction. For example, the application might present a GUI

to the user, with entry fields for the transaction data. Some of the input required to run the transaction relates to the specific transaction; other information is required by IMS Web.

The code segments in this section are based on the MFS source that the IMS Web sample uses (see “Basic IMS Web example” on page 43).

IMS Web requires the following input information:

User The RACF user ID to be passed to IMS OTMA.

Group The RACF group name to be passed to IMS OTMA.

Host The TCP/IP host name of the machine on which the IMS TCP/IP OTMA Connection is installed.

Port The port number for the IMS TCP/IP OTMA Connection.

IMS name
The XCF member name for IMS OTMA.

RUname
RUname is a unique name that identifies the request the application makes to execute the IMS transaction. HWS uses this name as the CLIENTID and passes it to IMS where it is used as the name of the LTERM that is associated with the transaction. Because a physical terminal is not involved, the application must generate this name. For the generated CGI-BIN program, IMS Web creates an RUname by combining a timestamp and a checksum of the Web server TCP/IP host name. The generated RUname will contain only the numerals 0 - 9 and the upper case letters of the English alphabet. You can use the getRUname method that IMS Web provides, in which case you must first instantiate class HWSUtil, where getRUname is a member function. Alternatively, you can generate a unique name using your own algorithm. If you generate your own name without using getRUname, then you should ensure that your algorithm uses only the numerals 0 - 9 and upper case letters in the RUname. Otherwise, when you enter the CLIENTID parameter of an IMS TCP/IP OTMA Connection (ITOC) command on the MVS console, MVS will convert the lower case letters in the CLIENTID to upper case, which will prevent you from being able to enter the correct CLIENTID for that client if its CLIENTID contains any lower case letters.

The logic of your workstation application approximately follows the logic of the CGI-BIN program and should include the following steps:

1. Obtain transaction input.
2. Create an instance of your transaction input class (a transaction input object), as in the following example.

```
LPageName1In  tranObj;       /* Create transaction input   */
                             /* object.
```

3. Using the transaction input data obtained in Step 1, invoke the _set methods inherited from HWSTranIn to begin populating the transaction input object, as in the following example.

```
/* Populate transaction input */
/* object.                    */
tranObj.setUser(...);
tranObj.setGroup(...);
```

```
tranObj.setHost(...);
tranObj.setPort(...);
tranObj.setIMS(...);
tranObj.setRUname(...);
```

4. Invoke the _set methods of the input logical page class to finish populating the transaction input object. Because IMS Web generates this class specifically for your transaction, there are _set methods for the input message fields in the input logical page. The way in which you obtain the data for these methods depends on your application. The following is an example.

```
tranObj.setLNAME(...);       /* Populate transaction input */
tranObj.setFNAME(...);       /* object with data for IMS   */
tranObj.setEMPNO(...);       /* application input message. */
tranObj.setSSN(...);
tranObj.setRATE(...);
...
```

5. Invoke the execute method on the transaction input object to execute the IMS transaction. The execute method builds the input message for the IMS application program by formatting the data of the transaction input object, submits the input message to the IMS application program using the IMS TCP/IP OTMA Connection, and then formats the output messages for return to the caller.

```
...
HWSTranOut *pLpg = NULL;     /* Transaction output object.  */
...
pLpg = tranObj.execute();    /* Execute the IMS transaction.*/
```

   The execute method returns a chain of one or more transaction output objects. Each transaction output object represents a logical page of an output message (aLpg). Multiple output messages can be returned.

   Your non-Web application processes these transaction output objects (returned by an IMS application program) as in the following example.

```
if (!pLpg)
{
   /* Handle error where no object is returned...         */
}
else
{
   while (pLpg)
   {
      ...                       /* Process transaction     */
                                /* output object.          */

      qLpg = pLpg;              /* Get next transaction    */
      pLpg = qLpg->getNext();   /* output object.          */
      free(qLpg);
   }
}
```

The processing of a transaction output object depends on your application. However, this processing usually involves retrieving the attribute values of the transaction output object, and then presenting them to the end user. You can retrieve these values in several ways. This following section describes two of those ways.

Retrieving transaction output object attribute values

This section details two of the ways in which you can retrieve the attribute values of the transaction output object.

1. The base class HWSTranOut includes the virtual function displayLpg. You can override this function in each of your transaction output classes to display the output logical page. The following code segments illustrate how the non-Web application uses the displayLpg method. The input to displayLpg is a void pointer that your application can use when it is appropriate.

```
if (!pLpg)
{
   /* Handle error.                                        */
}
else
{
   while (pLpg)
   {
      pLpg->displayLpg(NULL);  /* Process transaction     */
                               /* output object.          */
      qLpg = pLpg;             /* Get next transaction    */
      pLpg = qLpg->getNext();  /* output object.          */
      free(qLpg);
   }
}
```

   In addition to specifying the above code in your non-Web application, you override the displayLpg method inherited from HWSTranOut by adding the following line to the .hpp file for the output logical page.

```
virtual int displayLpg( void *inData );
```

   You also implement the overriding function by adding the following lines to the .cpp file for the output logical page. The following is a very simple example of displayLpg method implementation. This implementation uses the _get methods of the generated class to retrieve the values of the transaction output object.

```
// Display transaction output object...
int HWSLPG02Out::displayLpg(void *inData )
{
   printf("\nEmployee Data:");
   printf("\n*************\n");
   printf("\nMessage field LNAME:  %s",getLNAME());
   printf("\nMessage field FNAME:  %s",getFNAME());
   printf("\nMessage field EMPNO:  %s",getEMPNO());
   printf("\nMessage field SSN:    %s",getSSN());
   printf("\nMessage field RATE:   %s",getRATE());
   printf("\nMessage field MSGFLD: %s",getMSGFLD());
   printf("\nMessage field DATE:   %s",getDATE());
   return 0;
}
```

2. A more generic method of retrieving attribute values involves the use of the genList method of the transaction output object. genList returns the first object in a chain of objects of type HWSLNode. Each HWSLNode object contains an attribute name, a value, and a pointer to the next object in the chain.

   The following function illustrates how your non-Web application can use the genList method and HWSLNode objects to display default MODs that IMS returns. The section “Error handling” on page 15 contains code that illustrates the use of this function.

```
int displayDefault( HWSTranOut *aLpg )
{
   HWSLNode *aNode    = NULL;
   char     *dataBuf  = NULL;
   int      szDataBuf = 0;

   printf("\n\nDefault Transaction Output %s\n", aLpg->getClassName() );

   aNode = aLpg->genList();
```

```
      while( aNode )
      {
         printf("\nField Name = %s", aNode->name);

         if( dataBuf )
         {
            if( aNode->length + 1 > szDataBuf )
            {
               free( dataBuf );
               dataBuf = NULL;
               szDataBuf = aNode->length + 1;
               dataBuf = (char *)malloc( szDataBuf );
            }
            memcpy( dataBuf, aNode->data, aNode->length );
            *(dataBuf + (aNode->length)) = 0;
         }
         else
         {
            szDataBuf = aNode->length + 1;
            dataBuf = (char *)malloc( szDataBuf );
            memcpy( dataBuf, aNode->data, aNode->length );
            *(dataBuf + (aNode->length)) = 0;
         }

         printf("\nField Value = %s", dataBuf);
         aNode = aNode->next;
      }
      free( dataBuf );
      dataBuf = NULL;

      return 0;
   }
```

Error handling

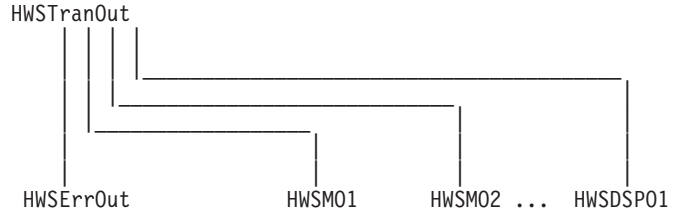
Either IMS Web or IMS can detect an error in the execution of a transaction. If IMS Web encounters an error while processing the execute method, an object of class HWSErrOut is returned, instead of a transaction output object. If IMS detects an error, it returns an error message that specifies a default MOD. In this case, an object of one of the default classes (HWSMO1, ...HWSMO5) is returned. The attribute values of any of these output objects can be retrieved and displayed in the same way that the attribute values of a transaction output object are retrieved and displayed.

Your application program must first determine the type of object that the execute method returns. To make this determination, use the getClassType method of the HWSTranOut class. The code segment in the following example illustrates the use of this method by a non-Web application program.

```
pLpg = tranObj.execute();

if (!pLpg)
{
   /* Handle error...                                    */
}
else
{
   switch( pLpg->getClassType() )
   {
      case HWS_USER:
         while (pLpg)
         {
```

```
       pLpg->displayLpg( NULL );
qLpg = pLpg;
pLpg = qLpg->getNext();
free(qLpg);
    } /* end while */
    break;

case HWS_ERROUT:
   printf("\n*************************************\n");
   printf("ERROR: Unable to execute transaction.\n");
   printf("Error Code: %s\n",((HWSErrOut *)pLpg)->geterrCode());
   printf("Error String: %s\n",((HWSErrOut *)pLpg)->geterrString());
   printf("Error Buffer: %s\n",((HWSErrOut *)pLpg)->geterrBuffer());
   break;

case HWS_MO1:
   displayDefault( pLpg );
   break;

case HWS_MO2:
   displayDefault( pLpg );
   break;

case HWS_MO3:
   displayDefault( pLpg );
   break;

case HWS_MO4:
   displayDefault( pLpg );
   break;

case HWS_MO5:
   displayDefault( pLpg );
   break;

case HWS_DSPO1:
   displayDefault( pLpg );
   break;

default:
   printf("ERROR: Unknown transaction output object type.\n");
   break;
}
```

The following figure illustrates the class hierarchy of the IMS Web error output class and IMS default output classes. For a detailed description of these classes, see “Chapter 5. IMS Web classes” on page 17.



Tips for non-Web applications

For tips that can help you in the development of a non-Web application, as well as other IMS Web tips, see “Tips” on page 40.

Chapter 5. IMS Web classes

Each IMS transaction is represented by a set of classes that are derived from the following two classes that IMS Web supplies:

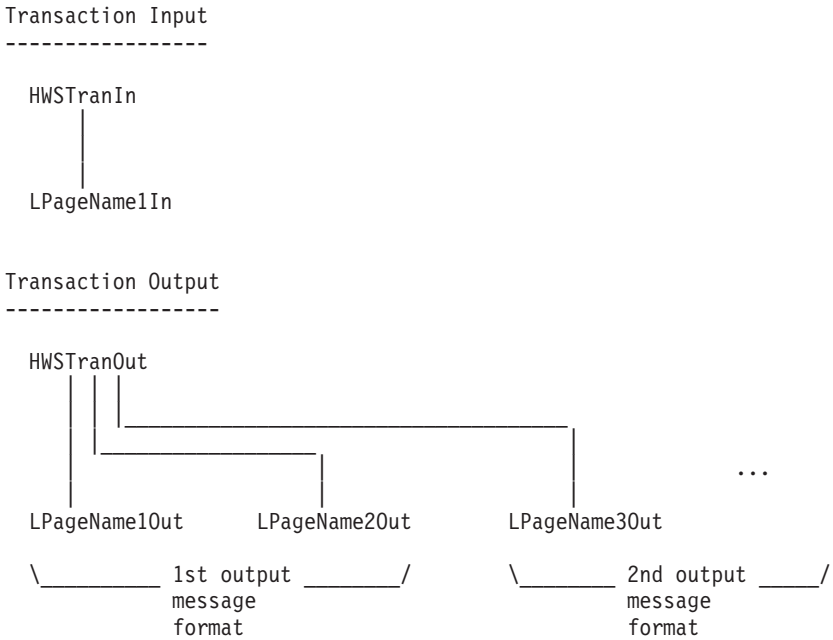
HWSTranIn
: This class and its subclasses represent the input to your IMS transaction.

HWSTranOut
: This class and its subclasses represent the output from your IMS transaction.

HWSTranIn and HWSTranOut encapsulate data and functionality that are common to *all* IMS transactions. Their subclasses, which are generated by IMS Web, encapsulate data and functionality that is specific to *your* transaction.

The first release of IMS Web processes nonconversational transactions that accept a single input message and return one or more output messages. The transaction input must be represented by a single input message definition and a single input logical page definition, but the transaction output can consist of multiple output message definitions, each containing multiple output logical page definitions.

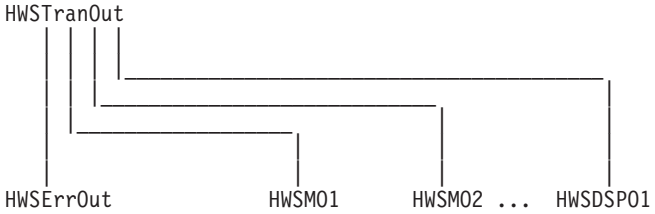
The following illustration shows the IMS Web class hierarchy for a typical transaction. IMS Web generates the subclass LPageName1In from the single input logical page definition of the single input message definition. IMS Web generates the subclasses LPageName1Out, LPageName2Out, LPageName3Out,... from the logical page definitions of the output message definitions. In this illustration, the transaction can output two possible message formats, each containing multiple logical page definitions.



Either IMS Web or IMS can detect an error in the execution of a transaction. If IMS Web encounters an error while it is processing the execute method, an object of class HWSErrOut is returned instead of a transaction output object. If IMS detects an error, it returns an error message and specifies a default MOD. In this case, an

object of one of the default classes (HWSMO1, ...HWSMO5) is returned. In addition, the default MOD HWSDSPO1 can be returned if the IMS Web application submits a command to IMS.

The following illustration shows the class hierarchy of the error and default classes.



IMS Web-supplied classes

HWSTranIn

Class name

HWSTranIn

Class HWSTranIn is an IMS Web-supplied class that encapsulates data and functionality that are common to the input of all IMS transactions. It is the parent (base) class for the transaction-specific input classes that IMS Web generates.

File stem

You can find the interface of class HWSTranIn in file HWSTIN.hpp in the include directory under the directory where the IMS Web Development component is installed. The defaults are as follows:

- c:\imsweb\include on OS/2
- c:\Program Files\IBM\IMS Web Studio\include on Windows
- /usr/lpp/imsweb/include on AIX, Sun, and OS/390

Parent class

None

Public attributes

Parentheses after an attribute name contain the attribute type.

- formatObj

  This attribute is for use by IMS Web-generated code only. It is not intended for customer use as a programming interface.
- language

  This attribute is for use by IMS Web-generated code only. It is not intended for customer use as a programming interface.
- cgiName

  This attribute is for use by IMS Web-generated code only. It is not intended for customer use as a programming interface.
- cgiName

  This attribute is for use by IMS Web-generated code only. It is not intended for customer use as a programming interface.
- userName (char *)

  The RACF userid to be passed to IMS TM through OTMA.
- groupName (char *)

The RACF group name to be passed to IMS TM through OTMA.

- hostName (char *)

  The TCP/IP hostname of the machine on which the IMS TCP/IP OTMA Connection is installed.

- hostPort (char *)

  The port number that is associated with the IMS TCP/IP OTMA Connection.

- imsName (char *)

  The datastore name for IMS OTMA.

- RUname (char *)

  RUname is a unique name that identifies the request that the application makes to execute the IMS transaction. IMS uses this name as the name of the LTERM that is associated with the transaction. Because a physical terminal is not involved, the application must generate this name. For the generated CGI-BIN program, IMS Web creates an RUname by combining a timestamp and a checksum of the Web server TCP/IP host name. You can use the getRUname method that IMS Web provides, or generate a unique number using your own algorithm. If you use the IMS Web method, you must first instantiate class HWSUtil, of which getRUname is a member function.

Public methods

_set methods are used to populate an object with data; _get methods are used to retrieve the data of an objec

- void setUser(char *userName)
- char *getUser()
- void setGroup(char *groupName)
- char *getGroup()
- void setHost(char *hostName)
- char *getHost()
- void setPort(char *hostPort)
- char *getPort()
- void setIMS(char *imsName)
- char *getIMS()
- void setRUname(char *RUname)
- char *getRUname()
- getCGIName

  This method is for use by IMS Web-generated code only. It is not intended for customer use as a programming interface.

- virtual HWSTranOut *execute()

  This is a virtual method. It is implemented in the generated subclass of HWSTranIn (see “LPageNameiIn” on page 23).

- export

  This method is for use by IMS Web-generated code only. It is not intended for customer use as a programming interface.

HWSTranOut

Class name

HWSTranOut

Class HWSTranOut is an IMS Web-supplied class that encapsulates data and functionality that are common to the output of all IMS transactions. It is the parent (base) class for the transaction-specific output classes generated by IMS Web.

File stem

You can find the interface of class HWSTranOut in file HWSTOUT.hpp in the include directory under the directory where the IMS Web Development component is installed. The defaults are as follows:

- c:\imsweb\include on OS/2
- c:\Program Files\IBM\IMS Web Studio\include on Windows
- /usr/lpp/imsweb/include on AIX, Sun, and OS/390

Parent class

None

Public attributes

None

Public methods

- getNext()

  This method is used to get the next HWSTranOut object that was returned from IMS. If there are no more HWSTranOut objects, this method returns NULL.

- setNext

  This method is for use by IMS Web-generated code only. It is not intended for customer use as a programming interface.

- char *getClassName()

  This method is used to determine the name of a transaction output object that is derived from HWSTranOut.

- TOUTType getClassType()

  This method is used to determine the type of a transaction output object that is derived from HWSTranOut and returned by the execute method. TTOUType is an enum type with the following values:

  HWS_USER
  The transaction output object returned is defined by the user. Its definition is represented by an MFS output logical page.

  HWS_ERROUT
  The transaction output object is an IMS Web error object. It is an instance of the class HWSErrOut.

  HWS_MO1
  The transaction output object is an IMS error object. It is an instance of the class HWSMO1. Its definition is represented by the logical page of the MFS output message DFSMO1.

  HWS_MO2
  The transaction output object is an IMS error object. It is an instance of the class HWSMO2. Its definition is represented by the logical page of the MFS output message DFSMO2.

  HWS_MO3
  The transaction output object is an IMS error object. It is an instance of the class HWSMO3. Its definition is represented by the logical page of the MFS output message DFSMO3.

HWS_MO4
The transaction output object is an IMS error object. It is an instance of the class HWSMO4. Its definition is represented by the logical page of the MFS output message DFSMO4.

HWS_MO5
The transaction output object is an IMS error object. It is an instance of the class HWSMO5. Its definition is represented by the logical page of the MFS output message DFSMO5.

HWS_DSPO1
The transaction output object is returned when IMS Web is used to submit an IMS command. It is an instance of the class HWSDSPO1. Its definition is represented by the logical page of the MFS output message DFSDSPO1.

- virtual char *genHTML()

  This is a virtual method. It is implemented in the generated subclass of HWSTranOut. Because this method generates output HTML for a Web browser to use, it is not appropriate for a non-Web application.

- virtual HWSLNode *genList()

  This is a virtual method. It is implemented in the generated subclass of HWSTranOut (see “LPageNamejOut” on page 23).

- virtual HWSLNode *genDisplayList()

  This is a virtual method. It is implemented in the generated subclass of HWSTranOut (see “LPageNamejOut” on page 23).

- virtual int displayLpg(void *inData)

  This is a virtual method. It is usually overridden in the generated subclass of HWSTranOut to display a specific output LPAGE (see “LPageNamejOut” on page 23).

HWSErrOut

Class name
HWSErrOut

Class HWSErrOut is an IMS Web-supplied class used to represent errors encountered during IMS Web transaction processing.

File stem
You can find the interface of class HWSErrOut in file HWSDMOD.hpp in the include directory under the directory where the IMS Web Development component is installed. The defaults are as follows:

- c:\imsweb\include on OS/2
- c:\Program Files\IBM\IMS Web Studio\include on Windows
- /usr/lpp/imsweb/include on AIX, Sun, and OS/390

Parent class
HWSTranOut

Public attributes
None

Public methods

- virtual char *genHTML()

This method returns a buffer containing HTML that displays the data of an HWSErrOut object on a Web browser. Because this method generates output HTML for a Web browser to use, it is not appropriate for a non-Web application.

- virtual HWSLNode *genList()

  This method returns an object, of type HWSLNode, that represents the first attribute value of an HWSErrOut object.

- char *geterrCode()

  This method returns a null terminated string containing the IMS Web code that is associated with the error.

- char *geterrString()

  This method returns a null terminated string describing the IMS Web error.

- char *geterrBuffer()

  This method returns a null terminated string of IMS Web error strings leading up to the error. It is primarily used for diagnosis by support personnel.

HWSLNode

Class name

HWSLNode

HWSLNode is used to represent IMS Web objects that have a name and a value. For example, the attribute values of a transaction output object are represented by a chain of HWSLNode objects.

File stem

You can find the interface of class HWSLNode in file HWSLNODE.hpp in the include directory under the directory where the IMS Web Development component is installed. The defaults are as follows:

- c:\imsweb\include on OS/2
- c:\Program Files\IBM\IMS Web Studio\include on Windows
- /usr/lpp/imsweb/include on AIX, Sun, and OS/390

Parent class

None

Public attributes

The parentheses after an attribute name contain the attribute type.

- name (char *)

  A null terminated string containing the name of the object; for example, the attribute name of a transaction output object.

- length (int)

  The length of the data.

- data (void *)

  A buffer containing **length** bytes of data; for example, the attribute value of a transaction output object. `data` is **not** a null terminated string.

- next (HWSLNode *)

  A pointer to the next HWSLNode object in the chain, or NULL.

- prev (HWSLNode *)

  A pointer to the previous HWSLNode object in the chain, or NULL.

Public methods
None

IMS Web-generated classes

LPageNameiIn

Class name
*LPageNamei*In

IMS Web generates a transaction input class for each IMS transaction. Its name is the name of the single input logical page that is selected in IMS Web Studio (*LPageNamei*), with the suffix In appended. Class *LPageNamei*In encapsulates data and functionality that are specific to the input of a particular IMS transaction. It inherits common data and functionality from its parent class, HWSTranIn.

Parent class
HWSTranIn

Public attributes
None

Public methods

- virtual HWSTranOut *execute()

  This method formats an input message from the data of the transaction input object, sends it to IMS for execution by the application program, formats the output message, and returns it as a transaction output object.
- void set*attribute1*(char *)

  This method is used to set the attribute value of a transaction input object.
- char *get*attribute1*()

  This method is used to obtain the attribute value of a transaction input object.
- void set*attribute2*(char *)

  This method is used to set the attribute value of a transaction input object.
- char *get*attribute2*()

  This method is used to obtain the attribute value of a transaction input object.
- ...

LPageNamejOut

Class name
*LPageNamej*Out

IMS Web generates one or more transaction output classes for each IMS transaction. There is a transaction output class for each logical page of each output message format that is associated with the IMS transaction. The name of each class, *LPageNamej*Out, is the name of the corresponding output logical page (*LPageNamej*), with the suffix Out appended. Class *LPageNamej*Out encapsulates data and functionality that are specific to an output logical page of a particular IMS transaction. It inherits common data and functionality from its parent class, HWSTranOut.

Parent class
HWSTranOut

Public attributes
None

Public methods

- virtual char *genHTML()

  This method returns a buffer containing HTML that displays the data of the transaction output object on a Web browser. Because this method generates output HTML, it is not appropriate for a non-Web application.
- virtual HWSLNode *genList()

  This method returns an object, of type HWSLNode, that represents the first attribute value of a transaction output object.
- virtual HWSLNode *genDisplayList()

  This method returns an object, of type HWSLNode, that represents the first displayable attribute value of a transaction output object. The list of displayable attribute values is used to build a Net.Data table for display by a generated Net.Data macro.
- char *get*attribute1*()

  This method is used to obtain the attribute value of a transaction output object.
- char *get*attribute2*()

  This method is used to obtain the attribute value of a transaction output object.
- ...
- int displayLpg(void *)

  You can add this method. It overrides the parent method and displays the transaction output object in a manner appropriate for your environment.

HTMproject_name

Class name
HTMIn

The generated CGI-BIN program uses the HTMIn class to parse the input string from the Web browser. The constructor for this class takes as input the encoded data from the input HTML form. The constructor parses out the data values from the string and uses them to populate the attributes of the HTMIn object.

Parent class
None

Public attributes
There are attributes for each entry field in the generated input HTML form. These attributes represent the input to the IMS transaction.

Public methods
None

Chapter 6. Developer's reference information

This section includes:

Detailed product description

This section describes the following:

MFS function

This section describes an overview of the IMS Message Format Service (MFS) and the MFS utitlity control statements.

Overview

The MFS makes it possible for an IMS application program to communicate with different types of terminals without changing the way in which it reads and builds messages. MFS formats input data from a device or remote program for presentation to the IMS application program, and then formats the output data from the application program for presentation to the output device or remote program.

An IMS transaction running in the IMS Web environment does not use MFS. The IMS Web Runtime component code provides a function similar to MFS; it prepares IMS application program input and output messages based on information found in MFS definitions. IMS Web Release 1 uses only the information in MFS Message Input Descriptors (MIDs) and MFS Message Output Descriptors (MODs) to format input and output messages. It does not use the information found in the associated MFS Device Input Formats (DIFs) and MFS Device Output Formats (DOFs). For IMS Web Release 1, multiple output messages from a transaction must all use the same MOD.

In place of the MFS device, IMS Web uses a combination of a CGI-BIN program and Web browser HTML. You input IMS transaction data via an IMS Web-supplied HTML form. The form data is submitted to an IMS Web-supplied CGI-BIN program. The CGI-BIN program then formats the input message, based on an MFS MID definition. IMS Web submits the input message to the IMS application, retrieves the output message, formats it based on an MFS MOD definition, and then returns the data via the CGI-BIN program to output HTML.

The IMS Web Studio tool uses the MFS MID and MOD utility control statements to generate the IMS Web-supplied HTML and CGI-BIN programs for a transaction. Release 1 of IMS Web supports most MFS utility control statements and options

that relate to the formatting of input and output messages. All mandatory options must be specified, even if IMS Web does not use them. The descriptions that follow make note of any exceptions.

Restriction: Release 1 of IMS Web supports only nonconversational transactions. That is, IMS Web processes a transaction that accepts a single input message and returns one or more output messages.

The following sections describe how IMS Web uses each MFS utility control statement to format input and output messages.

MFS utility control statements

Requirement: The MFS definitions that you input to IMS Web should first be processed by the MFS Language Utility to ensure that they are valid MFS definitions.

MSG statement set:

- MSG

  label Identifies this message descriptor to IMS Web.

  TYPE=
  Identifies this message definition to IMS Web as an input or output message definition.

  SOR= IMS Web does not use the `SOR=` keyword.

  OPT= Specifies the message formatting option that IMS Web uses to edit messages. IMS Web Release 1 supports options 1 and 2, but does not support option 3.

  NXT= IMS Web does not use the `NXT=` keyword. The single MID that IMS Web uses is selected using the IMS Web Studio tool. Any MOD that the IMS application program requests be used for formatting must be provided to the IMS Web Studio tool.

  PAGE=
  Because IMS Web formats an application program output message for HTML that is built by a CGI-BIN program, instead of for a specific device that is controlled by an operator, IMS Web implements operator logical paging differently. By default, the generated code uses the genHTML method to present, to the client browser, all fields of all output logical pages returned by the transaction.

  FILL= Specifies a fill character for output attributes. IMS Web does not use DPAGE statements. Instead, the `FILL=` specification on the MSG statement determines the fill character. The default value is `C' '`. IMS Web processes output attributes as follows (this list does not include the system control area or attribute bytes):
  - Bytes that contain X'X'00'' are replaced by X'X'40''.
  - X'X'3F'' in an output message field terminates significant data, or indicates absence of data.
  - Output data is converted to the IMS Web Runtime code page and encoding scheme.
  - Output message fields that contain less significant data than the defined field length are justified and filled with the fill character. If `FILL=PT` or `FILL=NULL`, the amount of significant data is used as the size of the output attribute.

Note: For release 2.1.0 and above, IMS Web uses `FILL=NULL` when formatting output data that is left justified.

- LPAGE

  IMS Web uses the LPAGE statement to identify a group of segments and fields that comprise a logical page.

  An input or output message definition can contain multiple logical pages. When an input message definition contains multiple logical pages, you must select one of the logical pages in IMS Web Studio to be used in the generation of the input transaction class. IMS Web's processing of multiple output logical pages is described in the `PAGE=` keyword of the MSG statement.

  label Identifies the logical page to IMS Web, and is used by IMS Web in the name of the corresponding generated class as well as for the name of the interface and implementation files. The *label* is optional in MFS source. If label is not specified, you can specify a label with IMS Web Studio, or IMS Web generates a label of the form HWSLPG*dd*, where *dd* = 00 to 99.

  SOR= IMS Web does not use the `SOR=` keyword.

  COND=
  Describes a conditional test performed by IMS Web. If the test is successful, the segment and field definitions following this logical page are used to format output.

  NXT= IMS Web does not use the `NXT=` keyword. The IMS application program provides the name of the MOD that IMS Web uses to format an output message.

  PROMPT=
  IMS Web does not use the `PROMPT=` keyword.

- PASSWORD

  IMS Web does not use the PASSWORD statement.

- SEG

  IMS Web uses the SEG statement to delineate message segments.

  label IMS Web does not use the label.

  EXIT= IMS Web does not provide support for segment edit routines. You can modify the CGI-BIN program to edit the segment at the attribute level, to perform a similar function.

  GRAPHIC=
  IMS Web does not use the `GRAPHIC=` keyword. If you need uppercase conversion of segment data, you can modify the CGI-BIN program to convert the segment at the attribute level.

- DO

  IMS Web uses the DO statement, and the associated ENDDO statement, to iteratively generate MFLD statements. The use of generated names is described below.

- MFLD

  IMS Web uses the MFLD statement to identify a message field as part of a message input or output segment.

  The pp value of LTH=(pp,nn) is ignored. The first byte of data received from the CGI-BIN program is considered to be the first byte of data for the message field.

  label Identifies the input or output message field. An attribute of the associated

logical page's class is generated for each message field. You can specify the attribute name when the class is generated, using IMS Web Studio. Otherwise, the attribute name is determined as follows:

– If *dfldname* is specified, it is used.
– If *dfldname* is not specified, the MFLD label is used.
– If the MFLD label is not specified, IMS Web generates an attribute name of the form HWSMF*ddd*, where *ddd* = 000 to 999.

For INPUT messages:

dfldname
IMS Web formats the data of the corresponding attribute as a field of the input message. Unless you override it with IMS Web Studio, *dfldname* is the name of the attribute and the name of the corresponding input field on the HTML form.

'literal' only
IMS Web inserts the literal in the input message. This type of input message field does not have a corresponding input field on the HTML form.

(dfldname,'literal')
IMS Web inserts the literal in the input message field if it does not receive data in the corresponding attribute. Unless you override it with IMS Web Studio, *dfldname* is the name of the attribute and the name of the corresponding input field on the HTML form.

For OUTPUT messages:

dfldname
IMS Web formats the data of the output message field to the attribute of the associated logical page class, which in turn displays on the output HTML form. Unless you override it with IMS Web Studio, *dfldname* is the name of the attribute and the name of the corresponding output field on the HTML form.

(dfldname,'literal')
IMS Web formats the literal to the attribute of the associated logical page class, which in turn displays on the output HTML form. Unless you override it with IMS Web Studio, *dfldname* is the name of the attribute and the name of the corresponding output field on the HTML form. Space for the literal is not allocated in the output message segment that the application program supplies.

(dfldname,system-literal)
Space for the literal is not allocated in the output message segment supplied by the IMS application program. However, IMS Web creates a class attribute into which it places the system-literal. Unless you override it with IMS Web Studio, *dfldname* is used as the name of the corresponding class attribute. IMS Web processes MFS system literals as follows:

– LTSEQ

  This system literal does not apply in the IMS Web environment. If it occurs in the MFS source, IMS Web places the 5-character string 00000 in the corresponding attribute.

– LTNAME

  IMS Web places the 8-character string corresponding to the IMS Web request unit name in the attribute for this system-literal. Request unit name is an IMS Web name that is analogous to logical terminal name in the legacy IMS environment. It uniquely identifies the processing of an IMS transaction by a CGI-BIN program. Request unit name is obtained as follows:

  - A checksum is formed from the TCP/IP hostname of the server machine.
  - The process id of the CGI-BIN program is added to the checksum.
  - The hex form of this number is appended to the string HWS.

– TIME

  IMS Web places the 8-character time string HH:MM:SS in the attribute for this system-literal.

– DATE1

  IMS Web places the 6-character date string YY.DDD in the attribute for this system-literal.

– DATE2

  IMS Web places the 8-character date string MM/DD/YY in the attribute for this system-literal.

– DATE3

  IMS Web places the 8-character date string DD/MM/YY in the attribute for this system-literal.

– DATE4

  IMS Web places the 8-character date string YY/MM/DD in the attribute for this system-literal.

– LPAGENO

  This system-literal does not apply in the IMS Web environment. If it occurs in the MFS source, IMS Web places the 4-character string 0000 in the corresponding attribute.

– LTMSG

  This system-literal does not apply in the IMS Web environment. If it occurs in the MFS source, IMS Web places a string of 14 blanks in the corresponding attribute.

(,SCA)

IMS Web formats the data of the output message field to the attribute of the associated logical page class. The system control area does not display on the output HTML form. You can modify the CGI-BIN program to examine the system control area attribute. Unless you override it with IMS Web Studio, the label of the MFLD statement is the name of the attribute. If there is no label, IMS Web uses HWSMFddd, where ddd = 000 to 999.

IMS Web does not convert the data of the system control area to the IMS Web Runtime platform, and it does not replace any X'00' bytes.

nothing

If none of the parameters dfldname, (dfldname,'literal'),

(dfldname,system-literal), or (,SCA) is specified in the MFS source, the output message field is considered to be a filler or spacer. IMS Web formats the data of the output message field to the corresponding generated attribute, but it does not display on the output HTML form. Unless you override it with IMS Web Studio, the MFLD statement label is the attribute name.

LTH= IMS Web uses the LTH= keyword to determine the length of the input or output message field. If LTH= is omitted for a literal, IMS Web uses the literal length for the field length.

IMS Web supports the form (pp,nn), where pp specifies which byte of the attribute is to be used as the first byte of data for the input message field.

– Example:

  If an input MFLD is defined to have no attribute bytes and LTH=(2,5), the default HTML form has a maximum text entry field length of 5. If you enter 3 data characters in the entry field, the value of the corresponding attribute consists of the 3-character string. IMS Web uses the second and third bytes of the attribute for the data of the input message field.

JUST=

For input, IMS Web left- or right-justifies and truncates the data of the input message field depending on the amount of data in the corresponding attribute.

– Example:

  If an input MFLD is defined to have no attribute bytes, JUST=R, FILL=C' ', and LTH=5, the default HTML form has a maximum text entry field length of 5. If you enter 3 data characters in the entry field, the value of the corresponding attribute consists of the 3-character string. IMS Web builds a message input field of 5 characters, with the rightmost 3 being the attribute value and the leftmost 2 containing the blank fill character.

– Truncation is not necessary when the default HTML form is used, because the maximum length of a text entry field (MAXLENGTH) is set to the defined length of the input field.

  For more information about the JUST= keyword in input messages, see "Justify and fill for input messages" on page 31.

For output, IMS Web left- or right-justifies and truncates the data of the corresponding attribute depending on the amount of data in the output message field.

– Example:

  An output MFLD is defined to have no attribute bytes, JUST=R, LTH=5, and the output message statement (MSG) is defined with FILL=C' '. If the field in the output message contains 4 data characters followed by X'3F', IMS Web builds the attribute value as a string of 5 characters beginning with a blank fill character and followed by the 4 data characters from the ouput message field.

For more information about the JUST= keyword in output messages, see "Justify and fill for output messages" on page 32.

ATTR=

IMS Web uses the ATTR= keyword to determine the number of attribute

bytes to set aside in the input and output message. The maximum length (MAXLENGTH) of the corresponding text entry field on the generated input HTML is the defined field length minus the number of attribute bytes.

The attribute bytes are not present in the generated output HTML. However, the attribute bytes are available to the CGI-BIN program as an attribute of the generated output logical page class. The attribute name is attrname_Attr, where attrname is the name of the attribute that corresponds to the data portion of the field. Attribute bytes are not converted from the host to the server platform.

IMS Web initializes the attribute bytes to blanks on input. On output, the bytes are available to the CGI-BIN program unconverted, on the server platform.

FILL= IMS Web uses the `FILL=` keyword to determine which character to use to pad an input message field when the length of the data received from the HTML form/CGI-BIN program is less than the length of the field. IMS Web converts fill characters specified as C'c' from the server platform code page to the host platform code page. Fill characters of the form X'hh' are not converted.

For more information about the `FILL=` keyword in input messages, see "Justify and fill for input messages". For more information about the `FILL=` keyword in output messages, see "Justify and fill for output messages" on page 32.

EXIT= IMS Web does not provide field edit routines.

- ENDDO

  IMS Web uses the ENDDO statement and the associated DO statement to iteratively generate MFLD statements.
- MSGEND

  IMS Web uses the MSGEND statement to terminate an input or output message definition.

Justify and fill for input messages: The following table shows how IMS Web handles the `JUST=` and `FILL=` keywords in the MFS source of an input message. The assumptions are:

- SIZE and MAXLENGTH of the text entry field associated with the input message field is 5 in all cases. This is analogous to the DFLD always having length 5. For example, the following HTML (modified from the generated HTML)

  ```
  <strong>IN09</strong> <td><INPUT TYPE="text" NAME="IN09" SIZE=5 MAXLENGTH=5 VALUE="">
  ```

  would be used for the following input message field definition:

  ```
  MFLD     IN09,FILL=C'0',LTH=7,JUST=L
  ```
- The rightmost table entries reflect the field as the application program would receive it in an input segment.
- The letter 'b' in the table below represents a blank.

Input HTML text entry field contents	FILL=	LTH=	JUST=	Field rec'd by appl. prog.	Note
ABCDb	X'40'	5	L	ABCDb	1
ABCDb	X'40'	5	R	ABCDb	1
1234	C'0'	5	L	12340	2

Input HTML text entry field contents	FILL=	LTH=	JUST=	Field rec'd by appl. prog.	Note
1234	C'0'	5	R	01234	2
A1B1	NULL	5	L	A1B1?	3
A1B1	NULL	5	R	A1B1?	3
nothing	X'40'	7	L	bbbbbbb	4
nothing	X'40'	7	R	bbbbbbb	4
1234b	C'0'	7	L	1234b000	5
1234b	C'0'	7	R	0001234b	5
nothing	NULL	7	L	???????	6
nothing	NULL	7	R	???????	6
12345	X'40'	3	L	123	7
12345	X'40'	3	R	345	7

Notes:

1. No fill or justification is necessary since the data entered in the HTML text entry field fully provides for the MFLD definition.
2. Since the data entered in the HTML text entry field is shorter than the MFLD definition, fill and justification must be applied. Justification is performed first, then the positions left are filled with the fill character.
3. Justification has no effect in this example since no matter where null fill is applied (left or right), the result is the same--a short field in the message segment. The fifth position in the field would actually be filled by the first character from the next field (as defined by the following MFLD statement).
4. Even though no data was entered inthe device field, a FILL=X'40' provides that the field in the message segment is filled with blanks. This preserves the relationship between fields for the application program. It can test for blanks and know that no data was entered for this field.
5. Only the positions with no data will have a fill applied to them.
6. Nothing from the input HTML text entry field finds its way into the input message. All following fields are moved left seven positions so that data from subsequent message fields will occupy these positions. Remember that the application program must be prepared for this.
7. First justify, then fill (or truncate in this case).

Justify and fill for output messages: The following table shows how IMS Web handles the JUST= and FILL= keywords in the MFS source of an output message when the output message field contains the X'3F' character. The assumptions are:

- MFLD lengths are 6 in all cases.
- The letter 'b' in the table below represents a blank.
- The character '#' in the table below represents the X'3F' short field indicator.

Output message field content	JUST=	FILL=	Output HTML data	Note
1#3456	L	C'0'	100000	1
1#3456	R	C'0'	000001	1
b#3456	L	C'0'	00000	2

Output message field content	JUST=	FILL=	Output HTML data	Note
b#3456	R	C'0'	00000	3
#23456	L	C'0'	000000	4
#23456	R	C'0'	000000	4
1#3456	L	C''	1	5
1#3456	R	C''	1	6
b#3456	L	C''	no data displayed	7
b#3456	R	C''	no data displayed	7
#23456	L	C''	no data displayed	7
#23456	R	C''	no data displayed	7
1#3456	L	NULL or PT	1	8
1#3456	R	NULL or PT	1	8
b#3456	L	NULL or PT	no data displayed	9
b#3456	R	NULL or PT	no data displayed	9
#23456	L	NULL or PT	no data displayed	10
#23456	R	NULL or PT	no data displayed	10

Notes:

1. If you use a character fill, the output HTML data is justified and filled to the length in the MFLD definition.
2. In this case the output attribute and HTML data is 'b00000' but the browser does not display the leading blank.
3. In this case the output attribute and HTML data is '00000b' but the browser does not display the trailing blank.
4. There is no data in the output message field, since the X'3F' is in the first position, so the output HTML data is completely filled with the fill character (C'0') to the length in the MFLD definition.
5. In this case the output attribute and HTML data is '1bbbbb' but the browser does not display the trailing blanks.

   For release 2.1.0 and above, IMS Web uses FILL=NULL when formatting output data that is left justified.
6. In this case the output attribute and HTML data is 'bbbbb1' but the browser does not display the leading blanks.
7. Since the fill character is a blank (C' '), no data is displayed for the output HTML data. The corresponding attribute, however, consists of a string of 6 blanks.

   For release 2.1.0 and above, IMS Web uses FILL=NULL when formatting output data that is left justified.
8. The output attribute and HTML data is one-byte string consisting of the character '1'.
9. The output attribute is a one byte string consisting of the character blank, which the browser displays as no data.
10. The output attribute and HTML data contain no data.

FMT statement set: IMS Web does not use this statement set.

PDB statement set: IMS Web does not use this statement set.

TABLE statement set: IMS Web does not use this statement set.

Compilation statements: IMS Web does not use the following compilation statements:

- EJECT
- PRINT
- SPACE
- STACK
- TITLE
- UNSTACK

IMS Web handles other compilation statements as follows:

- ALPHA

  IMS Web Release 1 does not use the ALPHA compilation statement. If you use MFS source that contains the ALPHA statement, IMS Web could consider the MFS source to be unusable.
- COPY

  When downloading the MFS source for a transaction, you must also download all the COPY members that the MFS source uses.
- END

  The IMS Web MFS parser stops parsing any MFS source at the END compilation statement.
- EQU

  The EQU compilation statement defines a symbol as a substitution variable. All subsequent occurrences of the symbol used in the MFS source that is presented to IMS Web are replaced by the value specified in the operand field of the EQU statement.
- RESCAN

  The RESCAN compilation statement specifies whether (indicated by ON) or not (indicated by OFF) replacement text should be rescanned for further substitution. This statement controls the operation of EQU statements during replacement mode.

  The default value for RESCAN is OFF unless a number is specified. If ON is specified, replacement text can invoke further substitution within the substituted text, up to a maximum number of occurrences. The default number is 5.

Mapping transactions to classes

For a detailed description of IMS Web classes, see “Chapter 5. IMS Web classes” on page 17.

IMS Web Studio generates a set of C++ files that contain class definitions and implementations to enable IMS transactions. In general, the C++ classes are derived from the MFS source for the transaction as follows:

- A C++ class is generated for each logical page of the transaction.
- The generated class takes the logical page name as its name. You can override this name with IMS Web Studio.
- The fields of a logical page become attributes (or data members) of the generated class for that page.

- The name of a generated attribute is the field name. You can override this name with IMS Web Studio.
- Access methods are generated for each attribute. Both get and set methods are generated for input logical page attributes, while only get methods are generated for output logical page attributes.

Input logical page

The following methods are specific to a class that is generated for an input logical page:

- A setUser method to set the RACF userid
- A setGroup method to set the RACF group name
- A setHost method to set the hostname for TCP/IP
- An execute method to send the request to IMS and return a class instance for an output logical page

For a basic IMS Web application, the following files are generated specifically for an input logical page:

- An input HTML form that gathers required information and starts the CGI-BIN program.
- A C++ class that parses the input string from the input HTML form. The generated CGI-BIN program invokes this class to parse the input string from the Web browser.

In the case of a Net.Data application, a Net.Data macro is generated. This macro includes the input and output HTML and invocation of the transaction program by the IMS Web language environment.

Output logical page

The following methods are specific to a class that is generated for an output logical page:

- A genHTML method that returns a buffer with the generated HTML code to the Web client. In the case of a basic IMS Web application, the generated CGI-BIN program invokes this method to return the output message from the transaction to the Web browser.
- A genList method that returns the output message from the transaction in the form of a name-and-value pair list. You can use this method to customize the CGI-BIN program and the output HTML. The generated CGI-BIN program does not invoke this method.
- A genDisplayList method that returns the displayable fields of an output message in the form of a name-and-value pair list. This list produced by this method is used by the Net.Data IMS Web language environment to build a Net.Data table presented to the Web browser by the Net.Data macro.

CGI-BIN program

The CGI-BIN program of a basic IMS Web application uses the input logical page classes to submit the request to IMS as follows:

- Invokes the parser class to parse the input from the input HTML form.
- Invokes the execute method to send the request to IMS. The execute method returns an instance of one of the generated classes for output logical pages.

The CGI-BIN program uses the output logical page classes to produce the output HTML as follows:

- Invokes the genHTML method of the object instance returned from the execute method to obtain a buffer of HTML, and then returns the buffer to the Web client.

Transaction program

The transaction program invoked as a DLL by Net.Data's IMS Web language environment uses the input logical page classes to submit the request to IMS as follows:

- Uses the values input in the Net.Data input %HTML block to set the attributes of the input transaction object. These values are presented to the transaction program as a list prepared by the IMS Web language environment.
- Invokes the execute method to send the request to IMS. The execute method returns an instance of one of the generated classes for output logical pages.

The transaction program uses the output logical page classes to build a list of the displayable fields of the ouput logical page. This list is used by the IMS Web language environment to build a Net.Data table that is displayed on the Web client via the output %HTML block of the Net.Data macro.

The transaction code

The transaction code that IMS Web uses depends on the MFS source.

Example:

The first field of the input message is a literal, as follows:

```
MSGIN    MSG      TYPE=INPUT
                  MFLD     'TRANA '
                  ...
```

The transaction code is TRANA. This type of input message field is not displayed on the input HTML form.

Example:

The first field of the input message uses a default literal, as follows:

```
MSGIN    MSG      TYPE=INPUT
                  MFLD     (TRNFLD,'TRANA '),LTH=6
                  ...
```

The transaction code is TRANA if the text entry field TRNFLD on the input HTML form has no entry; otherwise, the transaction code is the value that is entered.

Example:

The first field of the input message is as follows:

```
 MSGIN    MSG      TYPE=INPUT
                   MFLD     TRNFLD,LTH=6
                   ...
```

The transaction code is the value that is entered into the text entry field TRNFLD on the input HTML form.

Generating code

The following files are generated for each project in the CGI direcory:

HTM*project_name***.HTM**
An input HTML form that is used to input data for the message that goes to the IMS application and that starts the CGI-BIN program.

CGI*project_name***.cpp**
A CGI-BIN program that uses the class generated for the input logical page, *input_lpage_name*, to submit a request (input message) to IMS as follows:

- Uses the parser class, HTM*project_name*, to parse the input from the generated input HTML form.
- Invokes the execute method of the class generated for the input logical page to send the request to IMS. The execute method returns one of the generated classes for output logical pages.

The generated CGI-BIN program then uses one of the classes generated for the output logical pages, *output_lpage_name*, to build and send the output HTML as follows:

- Invokes the genHTML method of the HWSTranOut object returned by the execute method to get a buffer with the generated HTML code to be sent back to the Web client.

HTM*project_name***.hpp and HTM***project_name***.cpp**
A C++ class that parses the input string from the generated input HTML form. The CGI-BIN program uses this class to parse the input from the generated input HTML, HTM*project_name*.HTM.

*input_lpage_name***.hpp and** *input_lpage_name***.cpp**
A single class is generated for the input logical page. This class includes the following:

- A setUser method to set the RACF user ID
- A setGroup method to set the RACF group name
- A setHost method to set the host name for TCP/IP
- An execute method to submit the request (input message) to IMS and return one of the generated classes for the output logical pages

*output_lpage_name***.hpp and** *output_lpage_name***.cpp**
A class is generated for each output logical page. Each class includes the following:

- A genHTML method to build a buffer of HTML to be returned to the Web client
- A genList method to build a name-and-value pair list that you use to generate customized HTML code to be returned to the Web client (for optional use if you want to modify the generated CGI-BIN program and HTML)

CGI*project_name***.MAK**
A makefile that builds the CGI*project_name*.EXE program to be run on the IMS Web Runtime component. You must move file CGI*project_name* to the Web server.

The following files are generated for each project in the DTW directory:

dtw*proj*.cpp
The C++ source code for the DLL or shared library that processes the IMS transaction using IMS Web.

dtw*proj*.def
The module definition file used in building the transaction DLL.

dtw*proj*.mac
The Net.Data Web macro used to interface with the IMS Web language environment. The language environment invokes the transaction DLL.

dtw*proj*.htm
A sample link that invokes Net.Data as a CGI-bin program and references the generated Web macro and input HTML form.

lpageIn.**cpp**
The C++ source (implementation file) for the class that represents the input logical page of the input message. This class includes a message that is used to submit the request (input message) to IMS and return one of the generated classes for the output logical pages.

lpageIn.**hpp**
The interface file for *lpageIn*.cpp.

lpageOut.**cpp**
The C++ source (implementation file) for an output logical page of an output message. A class is generated for each output logical page.

lpageOut.**hpp**
The interface file for *lpageOut*.cpp.

etc. A transaction might have cpp/hpp files for multiple output logical pages.

dtw*proj*.MAK
The makefile used to compile and link the source code, dtw*proj*.cpp, into a DLL or shared library.

Security

Overview

Release 1 of IMS Web does not provide security on the IMS Web Runtime component or browser platform. The generated CGI-BIN program passes, in the OTMA message that is sent to IMS, the user ID and group name that you specified in IMS Web Studio

As an IMS Web user, you can implement your own security on the IMS Web Runtime component platform so that only authorized users can execute certain CGI-BIN programs that map to IMS transactions.

If the IMS transaction that you are enabling on the Web is generally available to all users, probably all you need to define is a RACF user ID and group name with minimal security authorization. Users can then use this user ID and group name to generate the CGI-BIN program.

If the IMS transaction that you are enabling on the Web is not generally available, you can modify the CGI-BIN program to do the following:

- Obtain the current user's user ID

- Use this user ID to dynamically set the user ID by means of the setUser method of the transaction input object

Using the current user's user ID to set the user ID is successful when the user ID in the Web environment is identical to the user ID defined in RACF. When the user IDs are different, the CGI-BIN must also implement a user ID table lookup, or prompt the user dynamically. One way to prompt the user dynamically is to modify the input HTML form so that it has additional user ID and group name fields. Browser users can then specify their RACF user IDs and group names in these fields.

How IMS Web security works

The following sections describe how each of the following components implements IMS Web security:

- IMS Web Studio
- IMS Web Runtime component
- IMS OTMA

IMS Web Studio

IMS Web Studio lets you specify one RACF user ID and one group name per project. These names become constants in the generated CGI-BIN program. The CGI-BIN program sets the specified user ID and group name in the transaction input object using the setUser and setGroup methods and the generated user ID and group name constants.

IMS Web Runtime component

An IMS Web Runtime component message-formatting routine uses the user ID and group name in the transaction input object to set the user ID and group name in the OTMA message header.

IMS OTMA

IMS OTMA uses the user ID and group name in the OTMA message header to perform the following RACF operations:

- RACROUTE REQUEST=VERIFY PASSCHK=NO to get an ACEE (accessor environment element)
- RACROUTE REQUEST=
  - FASTAUTH, if the request is an IMS transaction
  - AUTH, if the request is an IMS command

Limitations and direction of IMS Web security

- The major security limitation of IMS Web, Release 1, is that users are not authenticated. In other words, the Web server is trusted, such that user ID and group name are passed to RACF without an associated password. Any user authentication must be done at the Web server platform.
- IBM plans to enable IMS Web, in the near future, to pass a password in the form of a RACF passticket to the IMS TCP/IP OTMA Connection. The plan is for the

IMS TCP/IP OTMA Connection to perform RACROUTE REQUEST=VERIFY to create a RACF UTOKEN. The UTOKEN will then be passed to IMS OTMA, rather than the user ID and group name.

- IBM also plans to implement IMS Web support for workstation-generated security certificates, as soon as both RACF and IMS support these certificates.

Tips

General IMS Web application development

The following tips assume that you are using a CGI-BIN program and .mak file that IMS Web has generated. If you modify either of these files, see “Non-Web applications” on page 41 for additional tips.

- If you want to study the generated code, look at the file hws.h, which contains many of the typedefs and defines that IMS Web uses. You can find hws.h as follows:
  - In path \include of the Windows NT IMS Web Studio tool, where path is the drive and directory where IMS Web Studio is installed
  - In path \include of the OS/2 IMS Web Development component, where path is the drive and directory where the development component is installed
  - In /usr/lpp/imsweb/include on an AIX, Sun, or OS/390 platform where the IMS Web Development component is installed
- If your Web server platform is OS/2, AIX, Sun, or OS/390, be sure to do the following:
  - Choose the appropriate target platform in the IMS Web Studio Web Info dialog box when you generate the code.
  - Transfer the generated .cpp, .hpp, and .mak files from the Windows NT or 95 platform where you ran the IMS Web Studio to your OS/2, AIX, Sun, or OS/390 IMS Web development platform.
  - Run NMAKE (for OS/2) or MAKE (for AIX, Sun, or OS/390) on your development platform to build the executable form of the generated CGI-BIN program.
- Remember to transfer the executable CGI-BIN program that you built using the generated .mak file, as well as the generated input HTML form, to the appropriate directories on your Web server.
- Ensure that the MID you selected in the IMS Web Studio MID Selection dialog box contains a transaction code in the first 8 bytes of the input message. The transaction code can be supplied either by the input HTML or by a default literal.
- If you are using Microsoft Internet Information Server and the generated CGI-BIN program fails with the following error:

  ```
  CGI Error

  The specified CGI application misbehaved by not returning a complete set of
  HTTP headers. The headers it did return are: headera,headerb...headerz
  ```

  To resolve this error, move the IMS Web Runtime component DLLs, as well as any runtime DLLs that the CGI-BIN program uses, to the CGI-BIN directory. IMS Web installs the runtime component DLLs in the Windows NT SYSTEM32 directory.

The following item pertains to the IMS TCP/IP OTMA Connection:

- If the IMS transaction enabled by IMS Web is returning multiple messages as output, these output messages must all use the same MOD. Otherwise, only the last MOD name is returned to IMS Web Runtime, and IMS Web fails to format the output messages correctly. This is the same restriction as LU 6.2 support of IMS.

The following items pertain to IMS Web and its prerequisite code:

- Until APAR PQ02806 is applied to OTMA, you must run with OTMA security NONE by entering `/SEC OTMA NONE` at the IMS prompt. If you do not do this the DFS1292 security violation message is returned to your IMS Web transaction.
- Until APAR PQ01918 is applied to OTMA, the browser does not receive the DFS0555 message when the MPP abends.

Non-Web applications

The following tips are for non-Web environments:

- In addition to the #include statements for the interfaces of the input and output logical pages, your non-Web application needs the following #include statements:
  - HWSDMOD.hpp

    The interface of the IMS Web class HWSErrOut, plus interfaces for the default output classes.
  - HWSLNode.hpp

    The interface of the IMS Web class HWSLNode. You need this class if you use the genList method of a transaction output object. The genList method returns an HWSLNode object for each attribute of a transaction output object.
- Remember to link your non-Web application with the IMS Web Runtime component. For example, if you are building your non-Web executable on a Windows NT platform, you link with the HWSTRAN.LIB, HWSFMT.LIB, HWSUTIL.LIB, HWSCOM.LIB, and HWSMFS.LIB library modules.
- Remember to use the appropriate system run-time libraries with your non-Web application. For example, if you are developing a Windows NT application and are building the debug configuration of your application executable, use the Debug Multithreaded DLL run-time libraries.
- If you modify an IMS Web-supplied .mak file or create your own .mak file, be sure that you have the correct preprocessor definitions, as explained in the following list.

  HWSNT4
  Must be defined if your non-Web application is to run on Windows NT 4.0.

  HWSOS2
  Must be defined if your non-Web application is to run on OS/2.

  HWSAIX
  Must be defined if your non-Web application is to run on AIX.

  HWSSUN
  Must be defined if your non-Web application is to run on SUN Solaris.

  HWS390
  Must be defined if your non-Web application is to run on OS/390.
- **Restriction regarding RUname**: You can use the getRUname method that IMS Web provides, in which case you must first instantiate class HWSUtil, where getRUname is a member function. Alternatively, you can generate a unique name

using your own algorithm. If you generate your own name without using getRUname, then you should ensure that your algorithm uses only the numerals 0 - 9 and upper case letters in the RUname. Otherwise, when you enter the CLIENTID parameter of an IMS TCP/IP OTMA Connection (ITOC) command on the MVS console, MVS will convert the lower case letters in the CLIENTID to upper case, which will prevent you from being able to enter the correct CLIENTID for that client if its CLIENTID contains any lower case letters.

Chapter 7. IMS Web samples

This section describes:

Basic IMS Web example

This example shows how you can use IMS Web to convert a legacy payroll application to a Web application. In Release 1, IMS Web applications are based on the MFS source of the legacy application. The application used in this example adds new employees to the payroll.

MFS source

The following MFS source shows the 3270 Model 1 screen format used in the legacy application, as well as the input and output message descriptors. First you type the employee information in the appropriate fields on the 3270 screen, and then press ENTER to send the input message to the IMS application program. The application program echoes the input data back to the user, and uses MSGFLD to present the status of the newly-added employee. If the user needs to take special notice of the message in MSGFLD, the employee number displays in red. (To cause the output data to display in red, the application program dynamically sets the extended attribute bytes of the EMPNO output message field.)

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/

PAYF    FMT
        DEF    TYPE=(3270,1),FEAT=IGNORE,DSCA=X'00A0',
               SYSMSG=MSGFLD
```

```
        DIV    TYPE=INOUT
        DPAGE  CURSOR=((3,13))
        DFLD   '****EMPLOYEE PAYROLL ****',POS=(1,8)
        DFLD   'LAST NAME:',POS=(4,2)
LNAME   DFLD   POS=(4,13),LTH=10
        DFLD   'FIRST NAME:',POS=(5,2)
FNAME   DFLD   POS=(5,14),LTH=10
        DFLD   'EMPL NO:',POS=(6,2)
EMPNO   DFLD   POS=(6,11),LTH=5,EATTR=(NEUTRAL)
        DFLD   'SOC SEC NO.:',POS=(7,2)
SSN     DFLD   POS=(7,15),LTH=11
        DFLD   'RATE OF PAY:$',POS=(8,2)
RATE    DFLD   POS=(8,16),LTH=9
MSGFLD  DFLD   POS=(10,2),LTH=50
DATE    DFLD   POS=(10,55),LTH=8
        FMTEND
PAYIN   MSG    TYPE=INPUT,SOR=(PAYF,IGNORE),NXT=PAYOUT
        SEG
        MFLD   'SKS1 '
        MFLD   LNAME,LTH=10
        MFLD   FNAME,LTH=10
        MFLD   EMPNO,LTH=7,ATTR=(NO,1)
        MFLD   SSN,LTH=11
        MFLD   RATE,LTH=9,JUST=R,FILL=C'0'
        MSGEND
PAYOUT  MSG    TYPE=OUTPUT,SOR=(PAYF,IGNORE),NXT=PAYIN,
               OPT=1,FILL=NULL
        SEG
        MFLD   LNAME,LTH=10
        MFLD   FNAME,LTH=10
        SEG
        MFLD   EMPNO,LTH=7,ATTR=(NO,1)
        MFLD   SSN,LTH=11
        SEG
        MFLD   RATE,LTH=9
        SEG
        MFLD   MSGFLD,LTH=50
        MFLD   (DATE,DATE2)
        MSGEND
```

Generated files

You use IMS Web Studio to generate the IMS Web application. After you have created an IMS Web Studio project and inserted the above MFS source file into the project, you are ready to generate and build your new IMS Web application. (Although the MFS source file contains both a FMT statement and MSG statements, IMS Web ignores the FMT statement and uses only the MSG statements in the generation process.)

For an IMS Web Studio project with the name HTMPAYF, IMS Web Studio generates the following files:

HTMPAYF.htm
: This file contains the HTML form that you use to input data to the IMS application. The generated HTML uses the POST method to invoke the generated CGI-BIN program and the Web server path that you provided to IMS Web Studio for its URL. To see the complete sample, see “HTMPAYF.htm” on page 55.

The following is the generated input form, as displayed by a browser.

CGIPAYF.cpp

This is the CGI-BIN program. Its executable form runs on your Web server. The CGI-BIN program uses the classes described below to parse the string from the HTML form, submit an input message containing the data from the HTML form to the IMS application, and then present the data in the output message from the IMS transaction to the Web browser. To see the complete sample, see “CGIPAYF.cpp” on page 56. You only need to build its executable form using the generated makefile (see “CGIPAYF.mak” on page 75).

HTMPAYF.hpp

This file contains the interface of the class that parses the string received from the input HTML form. The generated CGI-BIN program uses this class. To see the complete sample, see “HTMPAYF.hpp” on page 58.

HTMPAYF.cpp

This file contains the implementation of the class that parses the string received from the input HTML form. To see the complete sample, see “HTMPAYF.cpp” on page 59.

HWSLPG01.hpp

This file contains the interface of the input logical page class. Because the MFS source does not include an input LPAGE statement, IMS Web generates a default logical page. The generated CGI-BIN program uses this class. For a description of this class, see “Chapter 5. IMS Web classes” on page 17 . To see the complete sample, see “HWSLPG01.hpp” on page 62.

HWSLPG01.cpp

This file contains the implementation of the input logical page class. To see the complete sample, see “HWSLPG01.cpp” on page 63.

HWSLPG02.hpp

This file contains the interface of the output logical page class. Because the MFS source does not include an output LPAGE statement, IMS Web generates a default logical page. The generated CGI-BIN program uses this class. For a description of this class, see “Chapter 5. IMS Web classes” on page 17 . To see the complete sample, see “HWSLPG02.hpp” on page 70.

HWSLPG02.cpp

This file contains the implementation of the output logical page class. To see the complete sample, see “HWSLPG02.cpp” on page 72.

CGIPAYF.MAK

This file is used to build the executable form of the CGI-BIN program, CGIPAYF.EXE, for execution on a Windows NT Web server. To see the complete sample, see “CGIPAYF.mak” on page 75.

For the process of using the .mak file to create the executable form of the CGI-BIN program, see “Building an executable file” in the “IMS Web User’s Guide” or click here to link online.

Modifying input HTML

IMS Web does not use information in the MFS FMT statement that is associated with a transaction. Therefore, the literal information that is used to identify fields on the original 3270 screen is not available for the generated HTML. To create an input HTML form that closely resembles the original 3270 display screen, modify the generated HTML as follows:

1. Left align the table.
2. Replace the text preceding each text-entry field with the literal used in the MFS FMT statement.
3. Add a heading and horizontal rule.
4. Remove the IMS Web image.

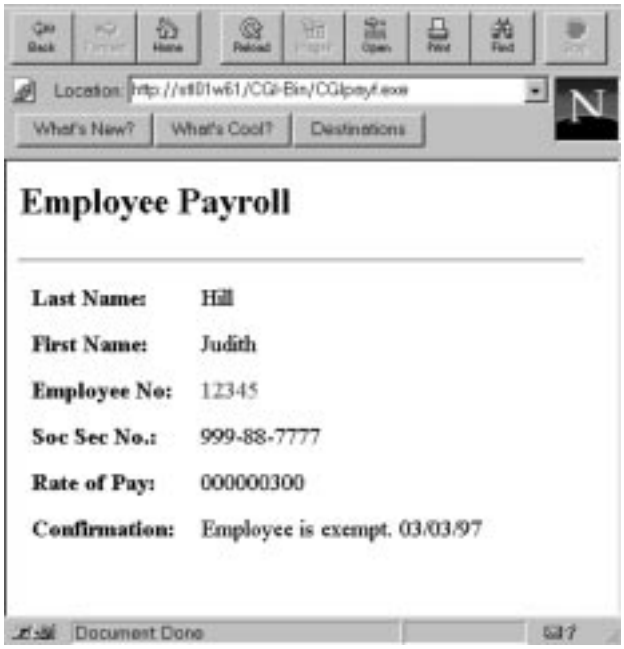
Following is the modified form and a link to the modified input HTML.

To see the complete modified input HTML sample, see “HTMPAYF2.HTM, modified input HTML,” on page 76.

Modifying output HTML

The CGI-BIN program uses the getHTML method of the output LPAGE to generate the output HTML. In this example, the following HTML is sent to the Web browser.

As is the case for input HTML, the output HTML is more meaningful if you modify it to resemble the original 3270 screen. Because the CGI-BIN program dynamically builds the output HTML, you must modify the code of the CGI-BIN program. The following code segments show how you can modify the CGI-BIN program and the genHTML method to produce the following output HTML.



To see the sample of a modified CGI-BIN program, see “Modified CGI-BIN program” on page 78.

The CGIPAYF.cpp file was modified to replace the default **Results** heading with the heading **Employee Payroll**.

To see the sample of a modified genHTML method, see “Modified genHTML method” on page 79.

Only the genHTML method of the HWSLPG02.cpp file is shown. Modifications are enclosed by the following comments.

```
// Modifications...
// End Modifications...
```

In order to display the employee number in red, the genHTML method had to be modified to examine the attribute bytes associated with the output message field, and to add HTML to set the color of the data appropriately. Ordinarily, genHTML skips attribute bytes.

For a detailed description of the customization process, see “Chapter 3. Customizing generated code” on page 5.

IMS Web Net.Data example

This example is similar to the basic IMS Web example in that it shows how you can modify the generated input and output HTML. Multiple output LPAGEs have been added to the message output descriptor for illustration purpose.

MFS source

The following MFS source shows the 3270 Model 1 screen format used in the legacy application, as well as the input and output message descriptors.

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
```

```
/*                                                                   */
/*********************************************************************/

PAYF    FMT
        DEF    TYPE=(3270,1),FEAT=IGNORE,DSCA=X'00A0',SYSMSG=MSGFLD
        DIV    TYPE=INOUT
        DPAGE  CURSOR=((3,13))
        DFLD   '****EMPLOYEE PAYROLL ****',POS=(1,8)
        DFLD   'LAST NAME:',POS=(4,2)
LNAME   DFLD   POS=(4,13),LTH=10
        DFLD   'FIRST NAME:',POS=(5,2)
FNAME   DFLD   POS=(5,14),LTH=10
        DFLD   'EMPL NO:',POS=(6,2)
EMPNO   DFLD   POS=(6,11),LTH=5,EATTR=(NEUTRAL)
        DFLD   'SOC SEC NO.:',POS=(7,2)
SSN     DFLD   POS=(7,15),LTH=11
        DFLD   'RATE OF PAY:$',POS=(8,2)
RATE    DFLD   POS=(8,16),LTH=9
MSGFLD  DFLD   POS=(10,2),LTH=50
DATE    DFLD   POS=(10,55),LTH=8
        FMTEND
PAYIN   MSG    TYPE=INPUT,SOR=(PAYF,IGNORE),NXT=PAYOUT
LPIN    LPAGE  SOR=(DUMMY)
        SEG
        MFLD   'SKS1 '
        MFLD   LNAME,LTH=10
        MFLD   FNAME,LTH=10
        MFLD   EMPNO,LTH=7,ATTR=(NO,1)
        MFLD   SSN,LTH=11
        MFLD   RATE,LTH=9,JUST=R,FILL=C'0'
        MSGEND
PAYOUT  MSG    TYPE=OUTPUT,SOR=(PAYF,IGNORE),NXT=PAYIN,OPT=1,FILL=NULL
LPOUTA  LPAGE  SOR=DUMMY,COND=(LASTN,=,'Hill      ')
        SEG
LASTN   MFLD   LNAME,LTH=10
        MFLD   FNAME,LTH=10
        SEG
        MFLD   EMPNO,LTH=7,ATTR=(NO,1)
        MFLD   SSN,LTH=11
        SEG
        MFLD   RATE,LTH=9
        SEG
        MFLD   MSGFLD,LTH=50
        MFLD   (DATE,DATE1)
LPOUTB  LPAGE  SOR=DUMMY,COND=(LASTN,=,'Lin       ')
        SEG
LASTN   MFLD   LNAME,LTH=10
        MFLD   FNAME,LTH=10
        SEG
        MFLD   EMPNO,LTH=7,ATTR=(NO,1)
        MFLD   SSN,LTH=11
        SEG
        MFLD   MSGFLD,LTH=50
        MFLD   (DATE,DATE2)
LPOUTC  LPAGE  SOR=DUMMY,COND=(LASTN,=,'Chung     ')
        SEG
LASTN   MFLD   LNAME,LTH=10
        MFLD   FNAME,LTH=10
        SEG
        MFLD   MSGFLD,LTH=50
        MFLD   (DATE,DATE3)
        MSGEND
PAYOUT9 MSG    TYPE=OUTPUT,SOR=(PAYF,IGNORE),NXT=PAYIN,OPT=1,FILL=NULL
LPOUT1  LPAGE  SOR=DUMMY,COND=(LASTN,=,'Hill      ')
        SEG
LASTN   MFLD   LNAME,LTH=10
        SEG
```

```
        MFLD   EMPNO,LTH=7,ATTR=(NO,1)
        MFLD   SSN,LTH=11
        SEG
        MFLD   RATE,LTH=9
        SEG
        MFLD   MSGFLD,LTH=50
        MFLD   (DATE,DATE1)
LPOUT2  LPAGE  SOR=DUMMY,COND=(LASTN,=,'Lin       ')
        SEG
LASTN   MFLD   LNAME,LTH=10
        SEG
        MFLD   EMPNO,LTH=7,ATTR=(NO,1)
        MFLD   SSN,LTH=11
        SEG
        MFLD   MSGFLD,LTH=50
        MFLD   (DATE,DATE2)
LPOUT3  LPAGE  SOR=DUMMY,COND=(LASTN,=,'Chung     ')
        SEG
LASTN   MFLD   LNAME,LTH=10
        SEG
        MFLD   MSGFLD,LTH=50
        MFLD   (DATE,DATE3)
        MSGEND
```

Generated files

You use IMS Web Studio to generate the IMS Web application. After you have created an IMS Web Studio project and inserted the above MFS source file into the project, you are ready to generate and build your new IMS Web application. (Although the MFS source file contains both a FMT statement and MSG statements, IMS Web ignores the FMT statement and uses only the MSG statements in the generation process.)

For an IMS Web Studio project with the name **payf**, IMS Web Studio generates a Net.Data macro in directory **payf\dtw**:

DTWpayf.mac
: The generated Net.Data macro contains %HTML blocks for the input and output of the IMS transaction, as well as a %FUNCTION block that defines a function that is used to submit the IMS transaction and receive its output. When the function is invoked, the data input in the input %HTML block, as well as the name of the generated transaction DLL, is passed to the Net.Data IMS Web language environment for processing. The output from the IMS transaction is returned to the transaction DLL, which in turn passes it to the IMS Web language environment. The language environment builds a Net.Data table from the transaction output, and the table is displayed by the output %HTML block of the Net.Data macro.

  In addition, IMS Web Studio generates the C++ code for the transaction program, a .mak file for building the a DLL from the transaction program, and .cpp and .hpp files for the input and output logical pages of the transaction.

  To see the complete sample, see “DTWpayf.mac” on page 81.

DTWpayf.cpp
: This is the transaction program. Its executable form is invoked by Net.Data’s IMS Web language environment. The transaction program uses the classes described below to submit an input message containing the data from the input %HTML block of the Net.Data macro to the IMS application, and then present the data in the output message from the IMS

transaction to the Web browser. You only need to build its executable form using the generated .mak file (see “DTWpayf.mak, generated make file” on page 92).

To see the complete sample, see “DTWpayf.cpp, generated transaction program” on page 88.

LPIN.hpp
This file contains the interface of the input logical page class. For a description of this class, see “Chapter 5. IMS Web classes” on page 17.

To see the complete sample, see “LPIN.hpp, generated input LPAGE interface file” on page 94.

LPIN.cpp
This file contains the implementation of the input logical page class.

To see the complete sample, see “LPIN.cpp, generated input LPAGE implementation file” on page 96.

LPOUTA.hpp
This file contains the interface of one of the output logical page classes. The generated transaction program uses this class. For a description of this class, see “Chapter 5. IMS Web classes” on page 17.

To see the complete sample, see “LPOUTA.hpp, generated output LPAGE interface file” on page 111.

LPOUTA.cpp
This file contains the implementation of one of the output logical page classes.

Additional .hpp and .cpp files are generated but not shown for output logical pages LPOUTB, LPOUTC, LPOUT1, LPOUT2, and LPOUT3.

To see the complete sample, see “LPOUTA.cpp, generated output LPAGE implementation file” on page 114.

DTWpayf.mak
This file is used to build the executable form of the transaction program, DTQpayf.dll, for execution on a Windows NT Web server.

For the process of using the .mak file to create the executable form of the transaction program, see “Buidling a CGI-BIN executable file” in the “IMS Web User’s Guide” or click here to link online.

To see the complete sample, see “DTWpayf.mak, generated make file” on page 92 .

Modifying input HTML

IMS Web does not use information in the MFS FMT statement that is associated with a transaction. Therefore, the literal information that is used to identify fields on the original 3270 screen is not available for the generated HTML. To create an input HTML form that more closely resembles the original 3270 display screen, modify the input %HTML block of the Net.Data macro as follows:

1. Left align the table.
2. Replace the text preceding each text-entry field with the literals similar to those used in the MFS FMT statement.
3. Add a heading and horizontal rule.
4. Remove the IMS Web image.

Following is the modified input HTML form from the input %HTML block of the Net.Data macro and a link to the modified macro.

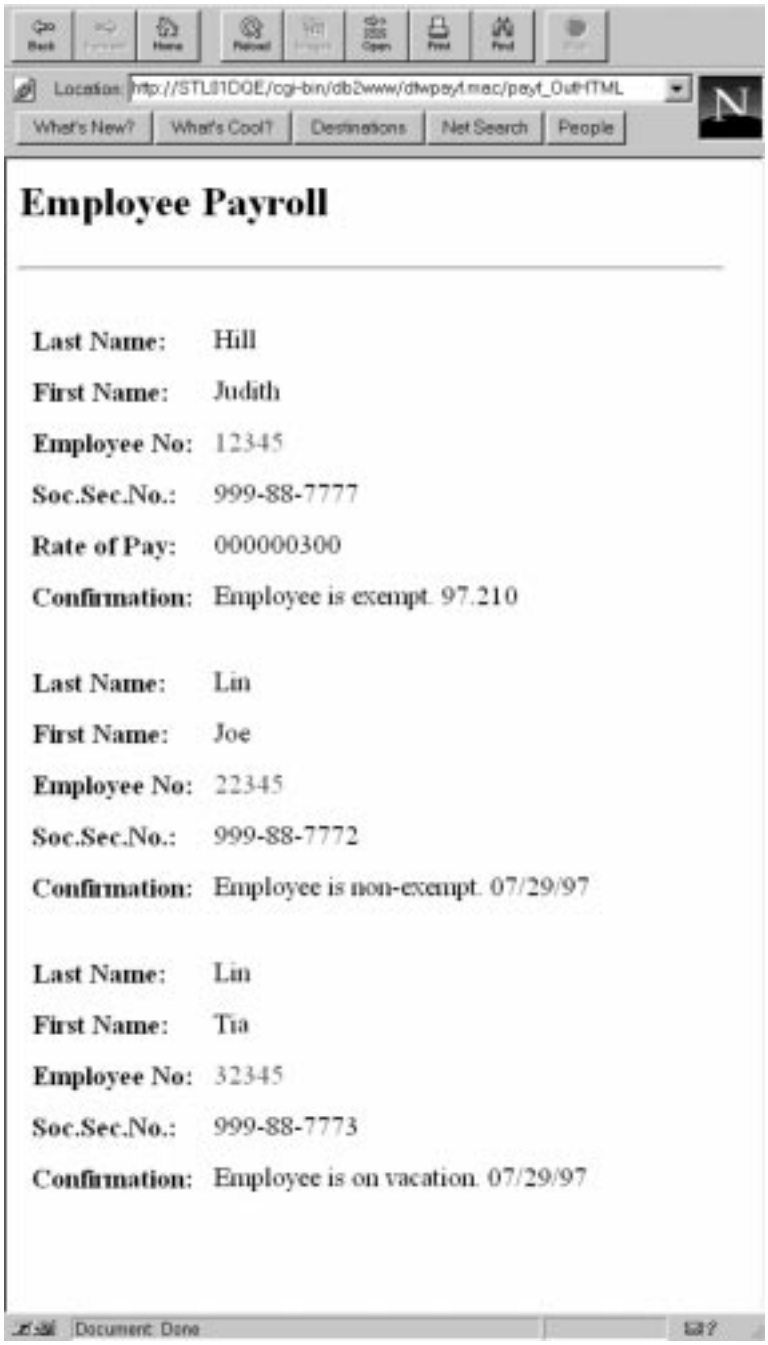
To see the complete modified macro sample, see “DTWpayf.mac” on page 81.

Modifying output HTML

The output of the IMS transaction is presented by the output %HTML block of the Net.Data macro. The output %HTML block, **payf_OutHTML**, contains an invocation of the function defined in the %FUNCTION block. This function is invoked with the values entered in the input %HTML block, as well as the name of the transaction DLL. Net.Data invokes the IMS Web language environment which, in turn, invokes the transaction DLL to submit the IMS transaction. The IMS Web language environment builds a Net.Data table, **payf_RESULTS**, from the output of the transaction. The %REPORT block of the function then presents the data in the table as part of the output HTML.

As is the case for input HTML, the output HTML is more meaningful if you modify it to resemble the original 3270 screen. Most changes can be made directly to the Net.Data macro. The %DEFINE section at the beginning of the Net.Data macro contains many of the heading values, and additional formatting, such as horizontal

rules, can be added to the output %HTML block and %REPORT block.



To see the complete modified macro sample, see “DTWpayf.mac” on page 81.

The table containing the output of the IMS transaction only contains ″displayable″ data of the output message. That is, any nontextual data, such as attribute bytes, SCA data, and so forth, is not represented in the table. If you wish to process this data you must modify the **genDisplayList** method to have the nondisplayable data added to the table. To see a description of the transaction output table, see “Sample IMS transaction table output” on page 126.

In order to display the employee number in red, the **genDisplayList** methods of output lpage classes LPOUTA and LPOUTB have been modified. In this case, the attribute bytes for the employee number (HWSMF008_ATTR), are added to the list that is used to build the table. Note that, when values are added to a row (LPage) of the transaction output table, the names of the following fields change accordingly. Hence, the %REPORT section for LPages LPOUTA and LPOUTB must be changed:

```
$(V_MFLD03Value) now contains attribute information, $(V_MFLD04Value) now contains the
Employee No. (formerly in $(V_MFLD03Value)), $(V_MFLD05Value) now contains the Soc.Sec.No.
(formerly in $(V_MFLD04Value)), and so forth.
```

To see the modified genDisplayList method in LPAGE LPOUTA sample, see "LPOUTA.cpp, modified output LPAGE implementation file" on page 121.

To see the modified %REPORT block in the Net.Data macro sample, see "DTWpayf.mac" on page 81.

Complete sample set

HTMPAYF.htm

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/


<!-- ********************************************************* -->
<!-- *                                                       * -->
<!-- *  IMS Web generated HTML input form.                   * -->
<!-- *                                                       * -->
<!-- *  This form :                                          * -->
<!-- *                                                       * -->
<!-- *   - has a table with label-entry fields pair for      * -->
<!-- *     each input field for the associated transaction.  * -->
<!-- *   - upon clicking on SUBMIT button, will POST the     * -->
<!-- *     corresponding CGI-Bin program.                    * -->
<!-- *                                                       * -->
```

```
<!-- *  You may modify this form as long as the names for    * -->
<!-- *  each entry fields are not changed.                   * -->
<!-- *                                                       * -->
<!-- ******************************************************** -->
<HTML>
<HEAD>
<TITLE>IMS.web.</TITLE>
</HEAD>
<BODY TEXT="000000">
<form method="POST" action="/CGI-Bin/CGIpayf.exe">
<center>
<table border="5" cellpadding="6">
<tr>
<td colspan=2 align="center" valign="center">
<strong>Enter the information below</strong><br>
<tr>
<td>
<strong>LNAME</strong>
<td>
<INPUT TYPE="text" NAME="LNAME" SIZE=10 MAXLENGTH=10 VALUE="">
<tr>
<td>
<strong>FNAME</strong>
<td>
<INPUT TYPE="text" NAME="FNAME" SIZE=10 MAXLENGTH=10 VALUE="">
<tr>
<td>
<strong>EMPNO</strong>
<td>
<INPUT TYPE="text" NAME="EMPNO" SIZE=5 MAXLENGTH=5 VALUE="">
<tr>
<td>
<strong>SSN</strong>
<td>
<INPUT TYPE="text" NAME="SSN" SIZE=11 MAXLENGTH=11 VALUE="">
<tr>
<td>
<strong>RATE</strong>
<td>
<INPUT TYPE="text" NAME="RATE" SIZE=9 MAXLENGTH=9 VALUE="">
</strong>
</td>
<tr>
<td colspan=2 align="center" valign="center">
<input type="submit" value="SUBMIT">
<input type="reset"  value="RESET">
</td>
</table>
</center>
</form>
<P>
<CENTER>
<img src="ims1.gif" align="center" hspace="5"
alt="Another Web Site Powered by IMS.web" border=0>
</CENTER>
</BODY>
</HTML>
```

CGIPAYF.cpp

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
```

```
/*                                                                  */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                              */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.           */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                        */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the     */
/* possibility of such damages.                                     */
/*                                                                  */
/* DISTRIBUTION.                                                    */
/*                                                                  */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:             */
/*   - It bears the above Copyright notice and DISCLAIMER intact    */
/*   - The software is not for resale                               */
/*                                                                  */
/********************************************************************/


#include <iostream.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <process.h>
#include "hwsutil.hpp"
#include "HTMpayf.hpp"
#include "HWSLPG01.hpp"
#include "HWSLPG02.hpp"

/*********************************************************/
/*                                                       */
/*  IMS Web generated CGI-Bin executable source code.    */
/*                                                       */
/*  INPUT : string generated by input HTML from STDIN    */
/*  OUTPUT: generated HTML to STDOUT                     */
/*                                                       */
/*  This routine :                                       */
/*                                                       */
/*   - parses the input for associated IMS transaction   */
/*     by using the constructor of HTML parsing class,   */
/*   - invokes the transaction by calling the execute    */
/*     method of the transaction input class,            */
/*   - and generates the output HTML from the output of  */
/*     the IMS transaction by using the transaction      */
/*     output class.                                     */
/*                                                       */
/*********************************************************/

/********************/
/*   MAIN PROGRAM   */
/********************/

int main (int argc, char *argv[])
{
// The length of the following arrays must be changed to
// reflect the actual possible length calculated during GEN
char input[1024];
char ruName[9];

HtmIn *request;

HWSUtil *util;
```

```
HWSLPG01In tranObj;

HWSTranOut *pLpg = NULL;
HWSTranOut *qLpg = NULL;

/*******************************************************/
/* input comes from POST action of the input HTML file */
/*******************************************************/
cin >> input;
request = new HtmIn (input);

util = new HWSUtil;

cout << "Content-type: text/html\n\n";
cout << "<html><head><title>IMS.web</title></head>\n";
cout << "<BODY TEXT=\"000000\">\n";

cout << "<h2>Results</h2>\n";

tranObj.setUser("GOFISHIN");
tranObj.setGroup("HARRY");
tranObj.setHost("CSDMEC13");
tranObj.setPort("9999");
tranObj.setIMS("SOCKEYE");
tranObj.setRUname(util->getRUname(ruName));

tranObj.setLNAME(request->LNAME);

tranObj.setFNAME(request->FNAME);

tranObj.setEMPNO(request->EMPNO);

tranObj.setSSN(request->SSN);

tranObj.setRATE(request->RATE);

pLpg = tranObj.execute();

if (!pLpg) {
cout << "Error Occured in the Server CGI-BIN program contact the WEB.Master<p>\n";
} else {
while (pLpg) {
cout << "   " << pLpg->genHTML() << "<p>";
qLpg = pLpg;
pLpg = qLpg->getNext();
free(qLpg);
} /* end while */
} /* end if */

cout << "</body></html>\n";

delete request;
return 0;

}
```

HTMPAYF.hpp

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
```

```
/* of IBM Corporation.                                                */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/

#ifndef HTMpayf_HPP
#define HTMpayf_HPP

#include <stddef.h>
#include <ctype.h>

/*************************************************************/
/*                                                           */
/*  IMS Web generated HTML input parsing class defintion.*/
/*                                                           */
/*  This class :                                             */
/*                                                           */
/*   - parses the input for associated IMS transaction   */
/*     passed in the constructor and populates its       */
/*     attributes with the parsed value.                 */
/*                                                           */
/*************************************************************/

class HtmIn
{
public:
HtmIn  (char * req);
~HtmIn ();

char LNAME[11];

char FNAME[11];

char EMPNO[6];

char SSN[12];

char RATE[10];

};

#endif
```

HTMPAYF.cpp

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
```

```
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/



...

/*********************************************************/
/*                                                       */
/*  IMS Web generated HTML input parsing class.          */
/*                                                       */
/*  This class :                                         */
/*                                                       */
/*   - parses the input for associated IMS transaction   */
/*     passed in the constructor and populates its       */
/*     attributes with the parsed value.                 */
/*                                                       */
/*********************************************************/

#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "hwsutil.hpp"
#include "HTMpayf.hpp"

/***************/
/* Constructor */
/***************/

HtmIn::HtmIn (char * req)
{
int i;
char *preq, *key, *value, *posamp, *poseq;
HWSUtil util;

/************************************/
/* Change all plusses back to spaces */
/************************************/

for(i=0; req[i]; i++) {
if(req[i] == '+') req[i] = ' ';
} /* end for */

poseq = strchr(req,'=');
if (poseq) {
*poseq = '\0';
key = req;
preq = poseq+1;
```

```
} else {
key = NULL;
} /* end if */

while (key) {
posamp = strchr(preq,'&');
value = preq;
if (posamp) {
*posamp = '\0';
preq = posamp+1;
} else {
preq = NULL;
} /* end if */
if (*value) {
util.unper(value) ;
} else {
value = NULL;
} /* end if */

if (!strcmp(key,"LNAME")) {
if (value) {
strcpy(LNAME,value);
} else {
memset(LNAME,'\0',sizeof(LNAME));
} /* end if */
} /* end if */
if (!strcmp(key,"FNAME")) {
if (value) {
strcpy(FNAME,value);
} else {
memset(FNAME,'\0',sizeof(FNAME));
} /* end if */
} /* end if */
if (!strcmp(key,"EMPNO")) {
if (value) {
strcpy(EMPNO,value);
} else {
memset(EMPNO,'\0',sizeof(EMPNO));
} /* end if */
} /* end if */
if (!strcmp(key,"SSN")) {
if (value) {
strcpy(SSN,value);
} else {
memset(SSN,'\0',sizeof(SSN));
} /* end if */
} /* end if */
if (!strcmp(key,"RATE")) {
if (value) {
strcpy(RATE,value);
} else {
memset(RATE,'\0',sizeof(RATE));
} /* end if */
} /* end if */

if (preq) {
poseq = strchr(preq,'=');
} else {
poseq = NULL;
} /* end if */
if (poseq) {
*poseq = '\0';
key = preq;
preq = poseq+1;
} else {
key = NULL;
} /* end if */
```

```
} /* end while */

}

/**************/
/* Destructor */
/**************/

HtmIn::~HtmIn ()
{
}
```

HWSLPG01.hpp

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/

...

#ifndef HWSLPG01_HPP
#define HWSLPG01_HPP

#include "HWSTIn.hpp"

#define HWSLPG01IN_MAX_HWSMF001 5
#define HWSLPG01IN_MAX_LNAME 10
#define HWSLPG01IN_MAX_FNAME 10
#define HWSLPG01IN_MAX_EMPNO 5
#define HWSLPG01IN_MAX_SSN 11
#define HWSLPG01IN_MAX_RATE 9

/*********************************************************/
/*                                                       */
/*  IMS Web generated transaction input class definition.*/
/*                                                       */
/*  This class :                                         */
/*                                                       */
/*   - provides the execute method which compiles a      */
/*     list of all attribute values and calls the server */
/*     DLL for building and passing the message to IMS.  */
```

```
/*                                                          */
/************************************************************/

class HWSLPG01In : public HWSTranIn {

public:

// LifeCycle
HWSLPG01In();
~HWSLPG01In();

// Overloaded Method
virtual HWSTranOut * execute();

// Access Methods for Fields
void setLNAME(char *);
inline char * getLNAME()    {return(LNAME);};
void setFNAME(char *);
inline char * getFNAME()    {return(FNAME);};
void setEMPNO(char *);
inline char * getEMPNO()    {return(EMPNO);};
void setSSN(char *);
inline char * getSSN()    {return(SSN);};
void setRATE(char *);
inline char * getRATE()    {return(RATE);};

private:

virtual HWSDList * export();

char HWSMF001[HWSLPG01IN_MAX_HWSMF001+1];
char LNAME[HWSLPG01IN_MAX_LNAME+1];
char FNAME[HWSLPG01IN_MAX_FNAME+1];
char EMPNO[HWSLPG01IN_MAX_EMPNO+1];
char SSN[HWSLPG01IN_MAX_SSN+1];
char RATE[HWSLPG01IN_MAX_RATE+1];

};

#endif
```

HWSLPG01.cpp

```
/********************************************************************/
/*                                                                  */
/* (c) Copyright IBM Corp. 1996                                     */
/* All Rights Reserved                                              */
/* Licensed Materials - Property of IBM                             */
/*                                                                  */
/* DISCLAIMER OF WARRANTIES.                                        */
/*                                                                  */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                              */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.           */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                        */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                     */
/*                                                                  */
/* DISTRIBUTION.                                                    */
/*                                                                  */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:             */
```

```
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/


...

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "HWS.h"
#include "HWSTran.hpp"
#include "HWSMFSF.hpp"
#include "HWSMsgF.hpp"
#include "HWSLPGF.hpp"
#include "HWSSegF.hpp"
#include "HWSFldF.hpp"
#include "HWSDMod.hpp"
#include "HWSDList.hpp"
#include "HWSLNode.hpp"
#include "HWSLPG01.hpp"
#include "HWSLPG02.hpp"

/*********************************************************/
/*                                                       */
/*  IMS Web generated transaction input class.           */
/*                                                       */
/*  This>class :                                         */
/*                                                       */
/*   - provides the execute method which compiles a      */
/*     list of all attribute values and calls the server */
/*     DLL for building and passing the message to IMS.  */
/*                                                       */
/*********************************************************/

// LifeCycle
HWSLPG01In::HWSLPG01In():
HWSTranIn("CGIpayf")
{

HWSMFSFormat   *aFmt;
HWSMsgFormat *aMsg;
HWSLPageFormat *aLpg;
HWSSegFormat *aSeg;
HWSFldFormat *aFld;
Condition   *aCond;

aFmt = new HWSMFSFormat();

aMsg = new HWSMsgFormat("PAYIN",OPT1,HWSIN);
aFmt->addMID(aMsg);

aCond = NULL;
aLpg = new HWSLPageFormat("HWSLPG01",NOCOND,aCond);
aMsg->addLPage(aLpg);

aSeg = new HWSSegFormat("HWSSEG01");
aLpg->addSeg(aSeg);

memcpy(HWSMF001,"SKS1 ",5);
memset(HWSMF001+5, '\0', 1);

aFld = new HWSFldFormat("HWSMF001",FLDLIT,5);

aFld->pp = 0;
aFld->justify = HWSLEFT;
```

```
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 5;
aFld->literalVal = (char *) malloc (5+1);
memcpy(aFld->literalVal,"SKS1 ",5);
memset(aFld->literalVal+5, '\0', 1);

aSeg->addFld(aFld);

LNAME[0] = '\0';

aFld = new HWSFldFormat("HWSMF002",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

FNAME[0] = '\0';

aFld = new HWSFldFormat("HWSMF003",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

EMPNO[0] = '\0';

aFld = new HWSFldFormat("HWSMF004",STANDARD,7);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 1;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

SSN[0] = '\0';

aFld = new HWSFldFormat("HWSMF005",STANDARD,11);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
```

```
aFld->literalVal = NULL;

aSeg->addFld(aFld);

RATE[0] = '\0';

aFld = new HWSFldFormat("HWSMF006",STANDARD,9);

aFld->pp = 0;
aFld->justify = HWSRIGHT;
aFld->fillType = HWSCHAR;
aFld->fill = '0';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

aMsg = new HWSMsgFormat("PAYOUT",OPT1,HWSOUT);
aFmt->addMOD(aMsg);

aCond = NULL;
aLpg = new HWSLPageFormat("HWSLPG02",NOCOND,aCond);
aMsg->addLPage(aLpg);

aSeg = new HWSSegFormat("HWSSEG02");
aLpg->addSeg(aSeg);

aFld = new HWSFldFormat("HWSMF007",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

aFld = new HWSFldFormat("HWSMF008",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

aSeg = new HWSSegFormat("HWSSEG03");
aLpg->addSeg(aSeg);

aFld = new HWSFldFormat("HWSMF009",STANDARD,7);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 1;
aFld->literalLen = 0;
```

```
aFld->literalVal = NULL;

aSeg->addFld(aFld);

aFld = new HWSFldFormat("HWSMF010",STANDARD,11);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

aSeg = new HWSSegFormat("HWSSEG04");
aLpg->addSeg(aSeg);

aFld = new HWSFldFormat("HWSMF011",STANDARD,9);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

aSeg = new HWSSegFormat("HWSSEG05");
aLpg->addSeg(aSeg);

aFld = new HWSFldFormat("HWSMF012",STANDARD,50);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

aFld = new HWSFldFormat("HWSMF013",SLITDAT2,8);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

formatObj = aFmt;

}

HWSLPG01In::~HWSLPG01In()
```

```
{
}

// Overloaded Method
HWSTranOut * HWSLPG01In::execute()
{
HWSTranOut * aMod = NULL;
HWSTranOut * pMod = NULL;
HWSTranOut * qMod = NULL;
HWSTran    aTran(this);
HWSDList   * pList;
HWSDList   * qList;
int errNum = 0;

pList = aTran.doIMS(export(),formatObj);

if (pList) {
while (pList) {
if (strcmp(pList->MDName,"HWSErrOut") == 0) {
pMod = new HWSErrOut(pList);
} else if (strcmp(pList->MDName,"PAYOUT&HWSLPG02") == 0) {
pMod = new HWSLPG02Out(pList);
} else if (strcmp(pList->MDName,"DFSM01&HWSMO1") == 0) {
pMod = new HWSMO1(pList);
} else if (strcmp(pList->MDName,"DFSM02&HWSMO2") == 0) {
pMod = new HWSMO2(pList);
} else if (strcmp(pList->MDName,"DFSM03&HWSMO3") == 0) {
pMod = new HWSMO3(pList);
} else if (strcmp(pList->MDName,"DFSM04&HWSMO4") == 0) {
pMod = new HWSMO4(pList);
} else if (strcmp(pList->MDName,"DFSM05&HWSMO5") == 0) {
pMod = new HWSMO5(pList);
} else if (strcmp(pList->MDName,"DFSDSP01&HWSDSPO1") == 0) {
pMod = new HWSDSPO1(pList);
} else {
pMod = new HWSErrOut(1000,"AN UNKNOWN MOD RETURNED BY HOST");
} /* end if */
if (!aMod) aMod = pMod;
if (qMod) qMod->setNext(pMod);
qMod = pMod;
qList = pList;
pList = qList->getNextList();
} /* end while */
} else {
pMod = new HWSErrOut(1001,"SEVERE ERROR NULL RETURNED FROM DLLs");
} /* end if */

return (aMod);

}

// Private Method Export
HWSDList * HWSLPG01In::export()
{
HWSDList  * expList;
HWSLNode  * aFld;
char   * iStr;
int      iStrLen;

expList = new HWSDList(formatObj->firstMID->name);

iStrLen = strlen(HWSMF001);
if (iStrLen > 0) {
iStr = (char *)malloc(iStrLen + 1);
strcpy(iStr,HWSMF001);
} else {
iStr = NULL;
```

```
iStrLen = 0;
}
aFld = new HWSLNode("HWSMF001",iStrLen,iStr);
expList->addNode(aFld);

iStrLen = strlen(LNAME);
if (iStrLen > 0) {
iStr = (char *)malloc(iStrLen + 1);
strcpy(iStr,LNAME);
} else {
iStr = NULL;
iStrLen = 0;
}
aFld = new HWSLNode("HWSMF002",iStrLen,iStr);
expList->addNode(aFld);

iStrLen = strlen(FNAME);
if (iStrLen > 0) {
iStr = (char *)malloc(iStrLen + 1);
strcpy(iStr,FNAME);
} else {
iStr = NULL;
iStrLen = 0;
}
aFld = new HWSLNode("HWSMF003",iStrLen,iStr);
expList->addNode(aFld);

iStrLen = strlen(EMPNO);
if (iStrLen > 0) {
iStr = (char *)malloc(iStrLen + 1);
strcpy(iStr,EMPNO);
} else {
iStr = NULL;
iStrLen = 0;
}
aFld = new HWSLNode("HWSMF004",iStrLen,iStr);
expList->addNode(aFld);

iStrLen = strlen(SSN);
if (iStrLen > 0) {
iStr = (char *)malloc(iStrLen + 1);
strcpy(iStr,SSN);
} else {
iStr = NULL;
iStrLen = 0;
}
aFld = new HWSLNode("HWSMF005",iStrLen,iStr);
expList->addNode(aFld);

iStrLen = strlen(RATE);
if (iStrLen > 0) {
iStr = (char *)malloc(iStrLen + 1);
strcpy(iStr,RATE);
} else {
iStr = NULL;
iStrLen = 0;
}
aFld = new HWSLNode("HWSMF006",iStrLen,iStr);
expList->addNode(aFld);

return (expList);

}

// Access Methods for Fields
void HWSLPG01In::setLNAME(char * value)
{
```

```
memset(LNAME,'\0',HWSLPG01IN_MAX_LNAME+1);
if (*value) {
memcpy(LNAME,value,(strlen(value)<=HWSLPG01IN_MAX_LNAME)?strlen(value):HWSLPG01IN_MAX_LNAME);
} /* end if */
}

void HWSLPG01In::setFNAME(char * value)
{
memset(FNAME,'\0',HWSLPG01IN_MAX_FNAME+1);
if (*value) {
memcpy(FNAME,value,(strlen(value)<=HWSLPG01IN_MAX_FNAME)?strlen(value):HWSLPG01IN_MAX_FNAME);
} /* end if */
}

void HWSLPG01In::setEMPNO(char * value)
{
memset(EMPNO,'\0',HWSLPG01IN_MAX_EMPNO+1);
if (*value) {
memcpy(EMPNO,value,(strlen(value)<=HWSLPG01IN_MAX_EMPNO)?strlen(value):HWSLPG01IN_MAX_EMPNO);
} /* end if */
}

void HWSLPG01In::setSSN(char * value)
{
memset(SSN,'\0',HWSLPG01IN_MAX_SSN+1);
if (*value) {
memcpy(SSN,value,(strlen(value)<=HWSLPG01IN_MAX_SSN)?strlen(value):HWSLPG01IN_MAX_SSN);
} /* end if */
}

void HWSLPG01In::setRATE(char * value)
{
memset(RATE,'\0',HWSLPG01IN_MAX_RATE+1);
if (*value) {
memcpy(RATE,value,(strlen(value)<=HWSLPG01IN_MAX_RATE)?strlen(value):HWSLPG01IN_MAX_RATE);
} /* end if */
}
```

HWSLPG02.hpp

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
```

```
/*********************************************************************/

...

#ifndef HWSLPG02_HPP
#define HWSLPG02_HPP

#include "HWSTOut.hpp"

#define HWSLPG02OUT_MAX_LNAME 10
#define HWSLPG02OUT_MAX_FNAME 10

#define HWSLPG02OUT_MAX_EMPNO 5
#define HWSLPG02OUT_MAX_EMPNO_ATTR 2
#define HWSLPG02OUT_MAX_SSN 11

#define HWSLPG02OUT_MAX_RATE 9

#define HWSLPG02OUT_MAX_MSGFLD 50
#define HWSLPG02OUT_MAX_DATE 8

/*********************************************************/
/*                                                       */
/*  IMS Web generated transaction output class definition.*/
/*                                                       */
/*  This class :                                         */
/*                                                       */
/*   - provides the genHTML method which generates the   */
/*     HTML output based on the result of IMS tran.      */
/*                                                       */
/*********************************************************/

class HWSLPG02Out : public HWSTranOut {

public:

// LifeCycle
HWSLPG02Out(HWSDList * fldList);
~HWSLPG02Out();

// Overloaded Method
virtual char     * genHTML();
virtual HWSLNode * genList();

// Access Methods for Fields
inline char * getLNAME()    {return(LNAME);};
inline char * getFNAME()    {return(FNAME);};
inline char * getEMPNO()    {return(EMPNO);};
inline char * getEMPNO_Attr()    {return(EMPNO_Attr);};
inline char * getSSN()    {return(SSN);};
inline char * getRATE()    {return(RATE);};
inline char * getMSGFLD()    {return(MSGFLD);};
inline char * getDATE()    {return(DATE);};

private:

HWSDList * fList;

char LNAME[HWSLPG02OUT_MAX_LNAME+1];
char FNAME[HWSLPG02OUT_MAX_FNAME+1];
char EMPNO[HWSLPG02OUT_MAX_EMPNO+1];
char EMPNO_Attr[HWSLPG02OUT_MAX_EMPNO_ATTR+1];
char SSN[HWSLPG02OUT_MAX_SSN+1];
char RATE[HWSLPG02OUT_MAX_RATE+1];
char MSGFLD[HWSLPG02OUT_MAX_MSGFLD+1];
char DATE[HWSLPG02OUT_MAX_DATE+1];
```

```
};

#endif
```

HWSLPG02.cpp

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/

...

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "HWS.h"
#include "HWSTran.hpp"
#include "HWSMFSF.hpp"
#include "HWSMsgF.hpp"
#include "HWSLPGF.hpp"
#include "HWSSegF.hpp"
#include "HWSFldF.hpp"
#include "HWSDMod.hpp"
#include "HWSDList.hpp"
#include "HWSLNode.hpp"
#include "HWSLPG02.hpp"

/*********************************************************/
/*                                                       */
/*  IMS Web generated transaction output class.          */
/*                                                       */
/*  This class :                                         */
/*                                                       */
/*   - provides the genHTML method which generates the   */
/*     HTML output based on the result of IMS tran.      */
/*                                                       */
/*********************************************************/

// LifeCycle
HWSLPG02Out::HWSLPG02Out(HWSDList * fldList):
HWSTranOut("HWSLPG02Out")
```

```
{

HWSLNode * aFld;

fList = fldList;

aFld = fldList->getFirst();
memset(LNAME,'\0',HWSLPG02OUT_MAX_LNAME+1);
memcpy(LNAME,aFld->data,(aFld->length <= HWSLPG02OUT_MAX_LNAME)?aFld->length:HWSLPG02OUT_MAX
_LNAME);
aFld = aFld->next;

memset(FNAME,'\0',HWSLPG02OUT_MAX_FNAME+1);
memcpy(FNAME,aFld->data,(aFld->length <= HWSLPG02OUT_MAX_FNAME)?aFld->length:HWSLPG02OUT_MAX
_FNAME);
aFld = aFld->next;

memset(EMPNO_Attr,'\0',HWSLPG02OUT_MAX_EMPNO_ATTR+1);
memcpy(EMPNO_Attr,aFld->data,(aFld->length <= HWSLPG02OUT_MAX_EMPNO_ATTR)?aFld->length:
HWSLPG02OUT_MAX_EMPNO);
aFld = aFld->next;
memset(EMPNO,'\0',HWSLPG02OUT_MAX_EMPNO+1);
memcpy(EMPNO,aFld->data,(aFld->length <= HWSLPG02OUT_MAX_EMPNO)?aFld->length:HWSLPG02OUT
_MAX_EMPNO);
aFld = aFld->next;

memset(SSN,'\0',HWSLPG02OUT_MAX_SSN+1);
memcpy(SSN,aFld->data,(aFld->length <= HWSLPG02OUT_MAX_SSN)?aFld->length:HWSLPG02OUT_MAX_SSN);
aFld = aFld->next;

memset(RATE,'\0',HWSLPG02OUT_MAX_RATE+1);
memcpy(RATE,aFld->data,(aFld->length <= HWSLPG02OUT_MAX_RATE)?aFld->length:HWSLPG02OUT
_MAX_RATE);
aFld = aFld->next;

memset(MSGFLD,'\0',HWSLPG02OUT_MAX_MSGFLD+1);
memcpy(MSGFLD,aFld->data,(aFld->length <= HWSLPG02OUT_MAX_MSGFLD)?aFld->length:HWSLPG02OUT
_MAX_MSGFLD);
aFld = aFld->next;

memset(DATE,'\0',HWSLPG02OUT_MAX_DATE+1);
memcpy(DATE,aFld->data,(aFld->length <= HWSLPG02OUT_MAX_DATE)?aFld->length:HWSLPG02OUT
_MAX_DATE);
aFld = aFld->next;

}

HWSLPG02Out::~HWSLPG02Out()
{
}

// Overloaded Method genHTML
char * HWSLPG02Out::genHTML()
{
char     * buf;
HWSLNode * aFld;

buf = (char *)malloc(sizeof(HWSLPG02Out)*10);
strcpy(buf,"<table border=5 cellpadding=6>\n");

aFld = fList->getFirst();
strcat(buf,"<tr><td><strong>");
strcat(buf,"LNAME");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
```

```
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"FNAME");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
/* skipping the extended attribute data*/
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"EMPNO");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"SSN");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"RATE");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"MSGFLD");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"DATE");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
strcat(buf,"</table>\n");
return buf;

}

// Overloaded Method genList
HWSLNode * HWSLPG02Out::genList()
{
HWSLNode * fldList  = NULL;
HWSLNode * aNode    = NULL;
HWSLNode * newNode  = NULL;
HWSLNode * prevNode  = NULL;
if (fList) {
for (aNode = fList->getFirst();
     aNode != NULL;
     aNode = aNode->next) {
newNode = new HWSLNode(aNode->name,aNode->length,aNode->data);
if (prevNode) {
prevNode->next = newNode;
} else {
fldList = newNode;
} /* end if */
```

```
prevNode = newNode;
} /* end for */
} /* end if */
return fldList;

}
```

CGIPAYF.mak

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/


####################################
# IMS.web Studio Generated NMAKE File #
####################################

!IF "$(CFG)" != "HWSRel" && "$(CFG)" != "HWSDbg"
!MESSAGE Invalid configuration "$(CFG)" specified.
!MESSAGE You can specify a configuration when running NMAKE on this makefile
!MESSAGE by defining the macro CFG on the command line.  For example:
!MESSAGE
!MESSAGE NMAKE /f "CGIpayf.mak" CFG="HWSDbg"
!MESSAGE
!MESSAGE Possible choices for configuration are:
!MESSAGE
!MESSAGE "HWSRel" : build a PRODUCTION version
!MESSAGE "HWSDbg" : build a DEBUG version
!MESSAGE
!ERROR An invalid configuration is specified.
!ENDIF

!IF "$(OS)" == "Windows_NT"
NULL=
!ELSE
NULL=nul
!ENDIF

!IF "$(CFG)" == "HWSRel"
CFLAGS= /nologo /MD /W3 /GX /O2 /D "WIN32" /D "NDEBUG" /D "_CONSOLE" /D "HWSNT4" /YX /c
LFLAGS= /nologo /subsystem:console /incremental:no /machine:IX86 /out:".\CGIpayf.exe"
```

```
!ENDIF
!IF "$(CFG)" == "HWSDbg"
CFLAGS=/nologo /MDd /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D "_CONSOLE" /D "HWSNT4"
 /YX /c
LFLAGS= /nologo /subsystem:console /incremental:no /debug /machine:IX86 /out:".\CGIpayf.exe"
!ENDIF
!IF "$(CFG)" == "HWSTrap"
CFLAGS=/nologo /MDd /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D "_TRAP" /D "_CONSOLE" /D
"HWSNT4" /YX /c
LFLAGS= /nologo /subsystem:console /incremental:no /debug /machine:IX86 /out:".\CGIpayf.exe"
!ENDIF

################################################################################
# Begin Project

ALL : ".\CGIpayf.exe"

LINK_OBJS= \
"HWSLPG01.obj"\
"HWSLPG02.obj"\
"HTMpayf.obj"\
"CGIpayf.obj"

LINK=link.exe
COMPILE=cl.exe

LINK_LIBS= \
 HWSTran.lib HWSCom.lib HWSUtil.lib HWSFMT.lib HWSMFS.lib

.CPP.obj:
 $(COMPILE) $(CFLAGS) %s

".\CGIpayf.exe" : \
    $(LINK_OBJS)
    $(LINK) @<<
     $(LFLAGS)
     $(LINK_LIBS)
     $(LINK_OBJS)
<<

############  END OF FILE  #############
```

HTMPAYF2.HTM, modified input HTML,

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
```

```
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/


<!-- ********************************************************* -->
<!-- *                                                       * -->
<!-- *  IMS Web generated HTML input form.                   * -->
<!-- *                                                       * -->
<!-- *  This form :                                          * -->
<!-- *                                                       * -->
<!-- *   - has a table with label-entry fields pair for      * -->
<!-- *     each input field for the associated transaction.  * -->
<!-- *   - upon clicking on SUBMIT button, will POST the     * -->
<!-- *     corresponding CGI-Bin program.                    * -->
<!-- *                                                       * -->
<!-- *  You may modify this form as long as the names for    * -->
<!-- *  each entry fields are not changed.                   * -->
<!-- *                                                       * -->
<!-- ********************************************************* -->
<HTML>
<HEAD>
<TITLE>IMS.web.</TITLE>
</HEAD>
<BODY TEXT="000000">
<form method="POST" action="/CGI-Bin/CGIpayf.exe">
<h2>Employee Payroll</h2>
<hr>
<center>
<table align="left" cellpadding="6">
<tr>
<td colspan=2 align="left" valign="center">
<strong>Enter the information below</strong><br>
<tr>
<td>
<strong>Last Name:</strong>
<td>
<INPUT TYPE="text" NAME="LNAME" SIZE=10 MAXLENGTH=10 VALUE="">
<tr>
<td>
<strong>First Name:</strong>
<td>
<INPUT TYPE="text" NAME="FNAME" SIZE=10 MAXLENGTH=10 VALUE="">
<tr>
<td>
<strong>Employee No:</strong>
<td>
<INPUT TYPE="text" NAME="EMPNO" SIZE=5 MAXLENGTH=5 VALUE="">
<tr>
<td>
<strong>Soc.Sec.No.:</strong>
<td>
<INPUT TYPE="text" NAME="SSN" SIZE=11 MAXLENGTH=11 VALUE="">
<tr>
<td>
<strong>Rate of Pay:</strong>
<td>
<INPUT TYPE="text" NAME="RATE" SIZE=9 MAXLENGTH=9 VALUE="">
</strong>
</td>
<tr>
<td colspan=2 align="left" valign="center">
<input type="submit" value="SUBMIT">
<input type="reset"  value="RESET">
</td>
```

```
</table>
</center>
</form>
<P>
</BODY>
</HTML>
```

Modified CGI-BIN program

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/


CGI-BIN Program

       '''

/*******************************************************/
/* input comes from POST action of the input HTML file */
/*******************************************************/
cin >> input;
request = new HtmIn (input);

util = new HWSUtil;

cout << "Content-type: text/html\n\n";
cout << "<html><head><title>IMS.web</title></head>\n";
cout << "<BODY TEXT=\"000000\">\n";

        // Modification...
// cout << "<h2>Results</h2>\n";
        // End Modification...

tranObj.setUser("GOFISHIN");
tranObj.setGroup("HARRY");
tranObj.setHost("CSDMEC13");
tranObj.setPort("9999");
tranObj.setIMS("SOCKEYE");
tranObj.setRUname(util->getRUname(ruName));

tranObj.setLNAME(request->LNAME);
```

```
tranObj.setFNAME(request->FNAME);

tranObj.setEMPNO(request->EMPNO);

tranObj.setSSN(request->SSN);

tranObj.setRATE(request->RATE);

pLpg = tranObj.execute();

if (!pLpg) {
cout << "Error Occured in the Server CGI-BIN program contact the WEB.Master<p>\n";
} else {
while (pLpg) {

         // Modification...
         cout << "<H2>Employee Payroll</H2>\n";
         cout << "<HR>\n";
         // End Modification...

cout << "   " << pLpg->genHTML() << "<p>";
qLpg = pLpg;
pLpg = qLpg->getNext();
free(qLpg);
} /* end while */
} /* end if */

cout << "</body></html>\n";

delete request;
return 0;

}
```

Modified genHTML method

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if they have been advised of the      */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not for resale                                */
/*                                                                   */
/*********************************************************************/
```

```
...

// Overloaded Method genHTML
char * HWSLPG02Out::genHTML()
{
char     * buf;
HWSLNode * aFld;

buf = (char *)malloc(sizeof(HWSLPG02Out)*10);

        // Modification...
// strcpy(buf,"<table border=5 cellpadding=6>\n");
strcpy(buf,"<table cellpadding=6>\n");
        // End Modification...

aFld = fList->getFirst();
strcat(buf,"<tr><td><strong>");
strcat(buf,"Last Name:");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"First Name:");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;

        // Modification...
        /*-------------------------------------------------------------------*/
/* Process extended attribute data.                                  */
        /*-------------------------------------------------------------------*/
        char colorRed[] = "\xC2\xF2";   /* Extended attribute is type is   */
                                        /* color and color is red.         */

        if( (memcmp( aFld->data, colorRed, 2) == 0) )
        {
           aFld = aFld->next;
      strcat(buf,"<tr><td><strong>");
    strcat(buf,"Employee No:");
    strcat(buf,"</strong>\n");
    strcat(buf,"<td><FONT COLOR=#FF0000>");
    strncat(buf,(char *)aFld->data,aFld->length);
    strcat(buf,"\n");
        }
        else
        {
    aFld = aFld->next;
    strcat(buf,"<tr><td><strong>");
    strcat(buf,"Employee No:");
    strcat(buf,"</strong>\n");
    strcat(buf,"<td>");
    strncat(buf,(char *)aFld->data,aFld->length);
    strcat(buf,"\n");
        }
        // End Modification...

aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"Soc Sec No.:");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
```

```
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"Rate of Pay:");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
strcat(buf,"<tr><td><strong>");
strcat(buf,"Confirmation:");
strcat(buf,"</strong>\n");
strcat(buf,"<td>");
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
        // Modification...
// strcat(buf,"<tr><td><strong>");
// strcat(buf,"DATE");
// strcat(buf,"</strong>\n");
// strcat(buf,"<td>");
        // End Modification...
strncat(buf,(char *)aFld->data,aFld->length);
strcat(buf,"\n");
aFld = aFld->next;
strcat(buf,"</table>\n");
return buf;

}

...
```

DTWpayf.mac

```
%DEFINE {
   payf_page_title      = "IMS Web payf"
   payf_input_prompt    = "Enter the information below"
   payf_output_heading  = "Employee Payroll"
   payf_alt_message     = "Another Web Site Powered by IMS Web"
   payf_RESULTS         = %TABLE(ALL)
%} %FUNCTION (HWS_LE)  payf (
                           IN    parm_LNAME,
                           IN    parm_FNAME,
                           IN    parm_EMPNO,
                           IN    parm_SSN,
                           IN    parm_RATE,
                           IN    parm_MAXCOLS,
                           IN    parm_TRANPGM,
                           INOUT parm_results ) {
   %REPORT (payf_RESULTS) {
      %ROW {

<TABLE CELLPADDING=6>

<BR>
     %IF ($(V_LPageName) =="HWSErrOut")
     <TR><TD>
<STRONG>Error Code</Strong><TD>$(V_MFLD01Value)
     <TR><TD>
<STRONG>Error String</STRONG><TD>$(V_MFLD02Value)
     <TR><TD>
<STRONG>Error Buffer</STRONG><TD>$<V_MFLD03Value)

<BR>
     %ELIF (($(V_LPageName) == "PAYOUT&LPOUTA") && ($(V_MFLD03Value)
== "RED"))
     <TR><TD>
<STRONG>Last Name:</STRONG><TD>$(V_MFLD01Value)
```

```
     <TR><TD>
<STRONG>First Name:</STRONG><TD>$(V_MFLD02Value)
     <TR><TD>
<STRONG>Employee No:</STRONG><TD><FONT COLOR=#FF0000>$(V_MFLD04Value)

 <TR><TD>
<STRONG>Soc.Sec.No.:</STRONG><TD>$(V_MFLD05Value)
      <TR><TD>
<STRONG>Rate of Pay:</STRONG><TD>$(V_MFLD06Value)
      <TR><TD>
<STRONG>Confirmation:</STRONG><TD>$(V_MFLD07Value) $(V_MFLD08Value)
      <BR>
      %ELIF (($(V_LPageName) == "PAYOUT&LPOUTA") && ($(V_MFLD03Value)

== "DFLT"))
      <TR><TD>
<STRONG>Last Name:</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>First Name:</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>Employee No:</STRONG><TD>$(V_MFLD04Value)
      <TR><TD>
<STRONG>Soc.Sec.No.:</STRONG><TD>$(V_MFLD05Value)
      <TR><TD>
<STRONG>Rate of Pay:</STRONG><TD>$(V_MFLD06Value)
      <TR><TD>
<STRONG>Confirmation:</STRONG><TD>$(V_MFLD07Value) $(V_MFLD08Value)
      <BR>
      %ELIF (($(V_LPageName) == "PAYOUT&LPOUTB") && ($(V_MFLD03Value)

== "RED"))
      <TR><TD>
<STRONG>Last Name:</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>First Name:</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>Employee No:</STRONG><TD><FONT COLOR=#FF0000>$(V_MFLD04Value)
      <TR><TD>
<STRONG>Soc.Sec.No.:</STRONG><TD>$(V_MFLD05Value)
      <TR><TD>
<STRONG>Confirmation:</STRONG><TD>$(V_MFLD06Value) $(V_MFLD07Value)
      <BR>
      %ELIF
      (($(V_LPageName) == "PAYOUT&LPOUTB") && ($(V_MFLD03Value)

== "DFLT"))
      <TR><TD>
<STRONG>Last Name:</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>First Name:</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>Employee No:</STRONG><TD>$(V_MFLD04Value)
      <TR><TD>
<STRONG>Soc.Sec.No.:</STRONG><TD>$(V_MFLD05Value)
      <TR><TD>
<STRONG>Confirmation:</STRONG><TD>$(V_MFLD06Value) $(V_MFLD07Value)
      <BR>
      %ELIF ($(V_LPageName) == "PAYOUT&LPOUTC")
      <TR><TD>
<STRONG>Last Name:</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>First Name:</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>Confirmation:</STRONG><TD>$(V_MFLD03Value) $(V_MFLD04Value)
    <BR>
      %ELIF ($(V_LPageName) == "PAYOUT9&LPOUT1")
      <TR><TD>
```

```
<STRONG>Last Name:</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>Employee No:</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>Soc.Sec.No.:</STRONG><TD>$(V_MFLD03Value)
      <TR><TD>
<STRONG>Rate of Pay:</STRONG><TD>$(V_MFLD04Value)
      <TR><TD>
<STRONG>Confirmation:</STRONG><TD>$(V_MFLD05Value) $(V_MFLD06Value)
     <BR>
      %ELIF ($(V_LPageName) == "PAYOUT9&LPOUT2")
      <TR><TD>
<STRONG>Last Name:</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>Employee No:</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>Soc.Sec.No.:</STRONG><TD>$(V_MFLD03Value)
      <TR><TD>
<STRONG>Confirmation:</STRONG><TD>$(V_MFLD04Value) $(V_MFLD05Value)
     <BR>
      %ELIF ($(V_LPageName) == "PAYOUT9&LPOUT3")
      <TR><TD>
<STRONG>Last Name:</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>Confirmation:</STRONG><TD>$(V_MFLD02Value) $(V_MFLD03Value)
      <BR>
      %ELIF
     ($(V_LPageName) == "DFSMO1&HWSMO1")
      <TR><TD>
<STRONG>OUTLINE</STRONG><TD>$(V_MFLD01Value)

<BR>
      %ELIF ($(V_LPageName) == "DFSMO2&HWSMO2")
      <TR><TD>
<STRONG>OUTL01</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>OUTL02</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>OUTL03</STRONG><TD>$(V_MFLD03Value)
      <TR><TD>
<STRONG>OUTL04</STRONG><TD>$(V_MFLD04Value)
      <TR><TD>
<STRONG>OUTL05</STRONG><TD>$(V_MFLD05Value)
      <TR><TD>
<STRONG>OUTL06</STRONG><TD>$(V_MFLD06Value)
      <TR><TD>
<STRONG>OUTL07</STRONG><TD>$(V_MFLD07Value)
      <TR><TD>
<STRONG>OUTL08</STRONG><TD>$(V_MFLD08Value)
      <TR><TD>
<STRONG>OUTL09</STRONG><TD>$(V_MFLD09Value)
      <TR><TD>
<STRONG>OUTL10</STRONG><TD>$(V_MFLD10Value)
      <TR><TD>
<STRONG>OUTL11</STRONG><TD>$(V_MFLD11Value)
      <TR><TD>
<STRONG>OUTL12</STRONG><TD>$(V_MFLD12Value)
      <TR><TD>
<STRONG>OUTL13</STRONG><TD>$(V_MFLD13Value)
      <TR><TD>
<STRONG>OUTL14</STRONG><TD>$(V_MFLD14Value)
      <TR><TD>
<STRONG>OUTL15</STRONG><TD>$(V_MFLD15Value)
      <TR><TD>
<STRONG>OUTL16</STRONG><TD>$(V_MFLD16Value)
      <TR><TD>
<STRONG>OUTL17</STRONG><TD>$(V_MFLD17Value)
```

```
      <TR><TD>
<STRONG>OUTL18</STRONG><TD>$(V_MFLD18Value)
      <TR><TD>
<STRONG>OUTL19</STRONG><TD>$(V_MFLD19Value)
      <TR><TD>
<STRONG>OUTL20</STRONG><TD>$(V_MFLD20Value)
      <TR><TD>
<STRONG>OUTL21</STRONG><TD>$(V_MFLD21Value)
      <TR><TD>
<STRONG>OUTL22</STRONG><TD>$(V_MFLD22Value)
      <BR>
      %ELIF ($(V_LPageName) == "DFSMO3&HWSMO3")
      <TR><TD>
<STRONG>OUTL01</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>OUTL02</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>OUTL03</STRONG><TD>$(V_MFLD03Value)
      <TR><TD>
<STRONG>OUTL04</STRONG><TD>$(V_MFLD04Value)
      <TR><TD>
<STRONG>OUTL05</STRONG><TD>$(V_MFLD05Value)
      <TR><TD>
<STRONG>OUTL06</STRONG><TD>$(V_MFLD06Value)
      <TR><TD>
<STRONG>OUTL07</STRONG><TD>$(V_MFLD07Value)
      <TR><TD>
<STRONG>OUTL08</STRONG><TD>$(V_MFLD08Value)
      <TR><TD>
<STRONG>OUTL09</STRONG><TD>$(V_MFLD09Value)
      <TR><TD>
<STRONG>OUTL10</STRONG><TD>$(V_MFLD10Value)
      <TR><TD>
<STRONG>OUTL11</STRONG><TD>$(V_MFLD11Value)
      <TR><TD>
<STRONG>OUTL12</STRONG><TD>$(V_MFLD12Value)
      <TR><TD>
<STRONG>OUTL13</STRONG><TD>$(V_MFLD13Value)
      <TR><TD>
<STRONG>OUTL14</STRONG><TD>$(V_MFLD14Value)
      <TR><TD>
<STRONG>OUTL15</STRONG><TD>$(V_MFLD15Value)
      <TR><TD>
<STRONG>OUTL16</STRONG><TD>$(V_MFLD16Value)
      <TR><TD>
<STRONG>OUTL17</STRONG><TD>$(V_MFLD17Value)
      <TR><TD>
<STRONG>OUTL18</STRONG><TD>$(V_MFLD18Value)
      <TR><TD>
<STRONG>OUTL19</STRONG><TD>$(V_MFLD19Value)
      <TR><TD>
<STRONG>OUTL20</STRONG><TD>$(V_MFLD20Value)
      <TR><TD>
<STRONG>OUTL21</STRONG><TD>$(V_MFLD21Value)
      <TR><TD>
<STRONG>OUTL22</STRONG><TD>$(V_MFLD22Value)

<BR>
      %ELIF ($(V_LPageName) == "DFSMO4&HWSMO4")
      <TR><TD>
<STRONG>OUTL01</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>OUTL02</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>OUTL03</STRONG><TD>$(V_MFLD03Value)
      <TR><TD>
<STRONG>OUTL04</STRONG><TD>$(V_MFLD04Value)
```

```
      <BR>
      %ELIF ($(V_LPageName) == "DFSDSPO1&HWSDSPO1")
      <TR><TD>
<STRONG>OUTL01</STRONG><TD>$(V_MFLD01Value)
      <TR><TD>
<STRONG>OUTL02</STRONG><TD>$(V_MFLD02Value)
      <TR><TD>
<STRONG>OUTL03</STRONG><TD>$(V_MFLD03Value)
      <TR><TD>
<STRONG>OUTL04</STRONG><TD>$(V_MFLD04Value)
      <TR><TD>
<STRONG>OUTL05</STRONG><TD>$(V_MFLD05Value)
      <TR><TD>
<STRONG>OUTL06</STRONG><TD>$(V_MFLD06Value)
      <TR><TD>
<STRONG>OUTL07</STRONG><TD>$(V_MFLD07Value)
      <TR><TD>
<STRONG>OUTL08</STRONG><TD>$(V_MFLD08Value)
      <TR><TD>
<STRONG>OUTL09</STRONG><TD>$(V_MFLD09Value)
      <TR><TD>
<STRONG>OUTL10</STRONG><TD>$(V_MFLD10Value)
      <TR><TD>
<STRONG>OUTL11</STRONG><TD>$(V_MFLD11Value)
      <TR><TD>
<STRONG>OUTL12</STRONG><TD>$(V_MFLD12Value)
      <TR><TD>
<STRONG>OUTL13</STRONG><TD>$(V_MFLD13Value)
      <TR><TD>
<STRONG>OUTL14</STRONG><TD>$(V_MFLD14Value)
      <TR><TD>
<STRONG>OUTL15</STRONG><TD>$(V_MFLD15Value)
      <TR><TD>
<STRONG>OUTL16</STRONG><TD>$(V_MFLD16Value)
      <TR><TD>
<STRONG>OUTL17</STRONG><TD>$(V_MFLD17Value)
      <TR><TD>
<STRONG>OUTL18</STRONG><TD>$(V_MFLD18Value)
      <TR><TD>
<STRONG>OUTL19</STRONG><TD>$(V_MFLD19Value)
      <TR><TD>
<STRONG>OUTL20</STRONG><TD>$(V_MFLD20Value)
      <TR><TD>
<STRONG>OUTL21</STRONG><TD>$(V_MFLD21Value)
      <TR><TD>
<STRONG>OUTL22</STRONG><TD>$(V_MFLD22Value)

<BR>
      %ENDIF


</TABLE>
            %}

   %}
   %MESSAGE {
       9000     : "Unable to load IMS Web transaction program
                  DTWpayf" : exit
       9001     : "Unable to locate doTran entry point in IMS Web
                  transaction program DTWpayf" : exit
      +default  : {

<P>Internal error $(RETURN_CODE) in the IMS Web Language Environment.

<P>Contact your IBM service representative.
                %} : exit
   %}
```

```
%} %HTML (payf_InHTML) {

<HTML>

<HEAD>

<TITLE>
      $(payf_page_title)
      </TITLE>

</HEAD>

<BODY TEXT="000000">

<FORM METHOD="POST" ACTION="payf_OutHTML">

<h2>Employee Payroll</h2>

<hr>

<CENTER>

<TABLE ALIGN="LEFT" CELLPADDING="6">
   <TR><TD COLSPAN=2 ALIGN="LEFT" VALIGN="CENTER">

<STRONG>
   $(payf_input_prompt)

</STRONG>

<BR>
   <TR>
      <TD>

<STRONG>Last Name:</STRONG>
   </TD>
      <TD>

<INPUT TYPE="TEXT" NAME="LNAME" SIZE=10 MAXLENGTH=10 VALUE="">
   </TD>
  </TR>
  <TR>
      <TD>

<STRONG>First Name:</STRONG>
      </TD>
      <TD>

<INPUT TYPE="TEXT" NAME="FNAME" SIZE=10 MAXLENGTH=10  VALUE="">
      </TD>
     </TR>

     <TR>
      <TD>

<STRONG>Employee No:</STRONG>
      </TD>
      <TD>

<INPUT TYPE="TEXT" NAME="EMPNO" SIZE=5 MAXLENGTH=5 VALUE="">
      </TD>
   </TR>
   <TR>
     <TD>

<STRONG>Soc.Sec.No.:</STRONG>
     </TD>
```

```
     <TD>

<INPUT TYPE="TEXT" NAME="SSN" SIZE=11 MAXLENGTH=11 VALUE="">
     </TD>
   </TR>
   <TR>
     <TD>

<STRONG>Rate of Pay:</STRONG>
     </TD>
     <TD>

<INPUT TYPE="TEXT" NAME="RATE" SIZE=9 MAXLENGTH=9 VALUE="">

     </TD>
    </TR>
    <TR>
       <TD COLSPAN=2 ALIGN="LEFT" VALIGN="CENTER">

<INPUT TYPE="SUBMIT" VALUE="SUBMIT">

<INPUT TYPE="RESET" VALUE="RESET">
     </TD>
    </TR>

</TABLE>

</CENTER>

</FORM>

<p>

</BODY>

</HTML>
%}
%HTML (payf_OutHTML) {

<html>

<head>

<title>$(payf_page_title)</title>

</head>

<BODY TEXT="000000">

<h2>$(payf_output_heading)</h2>

<hr>

   @payf(
          LNAME,
          FNAME,
          EMPNO,
          SSN,
          RATE,
          "23",
          "DTWpayf",
          payf_RESULTS )

<p>
```

```
</BODY>

%}
```

DTWpayf.cpp, generated transaction program

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if it has been advised of the         */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not offered for resale                        */
/*                                                                   */
/*********************************************************************/

#include #include #include #include "hwsutil.hpp"
#include "hwsdlist.hpp"
#include "hwslnode.hpp"
#include "hwsdtw.h"
            // DEBUG: HWSMsgGen::genCGIIncl
            // DEBUG: HWSLPageGen::genCGIIncl
#include "LPIN.hpp"
            // DEBUG: HWSMsgGen::genCGIIncl
            // DEBUG: HWSLPageGen::genCGIIncl
#include "LPOUTA.hpp"
            // DEBUG: HWSLPageGen::genCGIIncl
#include "LPOUTB.hpp"
            // DEBUG: HWSLPageGen::genCGIIncl
#include "LPOUTC.hpp"
            // DEBUG: HWSMsgGen::genCGIIncl
            // DEBUG: HWSLPageGen::genCGIIncl
#include "LPOUT1.hpp"
            // DEBUG: HWSLPageGen::genCGIIncl
#include "LPOUT2.hpp"
            // DEBUG: HWSLPageGen::genCGIIncl
#include "LPOUT3.hpp"
            // DEBUG: HWSMFSGen::genDTWCPP

/*********************************************************/
/*                                                       */
/*  IMS Web generated transaction DLL source code.       */
/*                                                       */
/*  INPUT : HWSDList of parameter values representing    */
/*          HTML form input from Net.Data LEI.           */
/*  OUTPUT: HWSDList of logical pages output by the      */
/*          transaction to Net.Data LEI.                 */
```

```
/*                                                          */
/*  This routine :                                          */
/*                                                          */
/*   - uses the nodes of the input HWSDList object to       */
/*     set the attributes of the transaction input          */
/*     object.                                              */
/*   - invokes the transaction by calling the execute       */
/*     method of the transaction input class,               */
/*   - and generates the output HTML from the output of     */
/*     the IMS transaction by using the transaction         */
/*     output class.                                        */
/*                                                          */
/************************************************************/

/************************************************************/
/*   doTran entry point.                                    */
/************************************************************/

HWSDList  *doTran( HWSDList *parmList )
{
   char      ruName[9];
   HWSUtil   *util      = NULL;
   HWSLNode  *aParm     = NULL;
   HWSDList  *LPageList = NULL;
   HWSDList  *allLPages = NULL;
   HWSLNode  *aNode     = NULL;
   HWSLNode  *newNode   = NULL;
   char      *clName    = NULL;
   char      errCode[40];

             // DEBUG: HWSMsgGen::genCGIDecl
             // DEBUG: HWSLPageGen::genCGIDecl
   LPINIn tranObj;

             // DEBUG: HWSMFSGen::genCGI
#ifdef _TRAP
   char *trap = NULL;
   *trap = 'Y';
#endif
   HWSTranOut *pLpg = NULL;
   HWSTranOut *qLpg = NULL;

   util = new HWSUtil;

   /*******************************************/
   /* Setting up the transaction input object.  */
   /*******************************************/
   /* Set the RACF userid. */
   tranObj.setUser("GOFISHIN");
   /* Set the RACF groupname. */
   tranObj.setGroup("HARRY");
   /* Set the hostname for TCP/IP OTMA Connection. */
   tranObj.setHost("CSDMEC13");
   /* Set the port address for TCP/IP OTMA Connection. */
   tranObj.setPort("9999");
   /* Set the Datastore id. */
   tranObj.setIMS("SOCKEYE");
   /* Set the requester unique name which is used as LTERM name. */
   tranObj.setRUname(util->getRUname(ruName));

   /************************************************************/
   /* Set the input attributes of tranObj from the nodes of    */
   /* the input parameter list.                                */
   /*                                                          */
   /* Note: In the case of parmList, data is a null            */
   /*       terminated string.  length does not include the    */
   /*       null.                                              */
```

```
/*                                                          */
/*       If parameter is null string, set method sets       */
/*       attribute to null.                                 */
/************************************************************/
for( aParm  = parmList->removeFirst();
     aParm != NULL;
     aParm  = parmList->removeFirst() ) {
         // DEBUG: HWSMsgGen::genDTWCPP
         // DEBUG: HWSLPageGen::genDTWCPP
         // DEBUG: HWSSegGen::genDTWCPP
         // DEBUG: HWSFldGen::genDTWCPP
         // DEBUG: HWSFldGen::genDTWCPP
if( !strcmp( aParm->name, "parm_LNAME" ) ) {
   if( aParm->data ) {
      tranObj.setLNAME( (char *)aParm->data );
   } /* end if */
} /* end if */

         // DEBUG: HWSFldGen::genDTWCPP
if( !strcmp( aParm->name, "parm_FNAME" ) ) {
   if( aParm->data ) {
      tranObj.setFNAME( (char *)aParm->data );
   } /* end if */
} /* end if */

         // DEBUG: HWSFldGen::genDTWCPP
if( !strcmp( aParm->name, "parm_EMPNO" ) ) {
   if( aParm->data ) {
      tranObj.setEMPNO( (char *)aParm->data );
   } /* end if */
} /* end if */

         // DEBUG: HWSFldGen::genDTWCPP
if( !strcmp( aParm->name, "parm_SSN" ) ) {
   if( aParm->data ) {
      tranObj.setSSN( (char *)aParm->data );
   } /* end if */
} /* end if */

         // DEBUG: HWSFldGen::genDTWCPP
if( !strcmp( aParm->name, "parm_RATE" ) ) {
   if( aParm->data ) {
      tranObj.setRATE( (char *)aParm->data );
   } /* end if */
} /* end if */

} /* end for */
         // DEBUG: HWSMFSGen::genDTWCPP
/************************************************************/
/* Calling the execute method to send the request to host.  */
/************************************************************/
pLpg = tranObj.execute();

/*********************************/
/* Processing output from host.  */
/*********************************/
if (!pLpg) {
   memset( errCode, 0, sizeof( errCode ) );
   _itoa( HWS_NULL_TRANOUT_OBJ, errCode, 10 );
   LPageList = new HWSDList( "DTW_Error" );
   newNode   = new HWSLNode( "DTW_Error", strlen( errCode ), errCode );
   LPageList->addNode( newNode );
   allLPages = LPageList;
} else {
   while (pLpg) {
      /*****************************************************/
      /* Build an HWSDList object for each transaction     */
```

```
         /* output object (lpage).                              */
         /*                                                     */
         /* The nodes of the HWSDList object correspond to the */
         /* displayable attributes of the output transaction   */
         /* object.  The excluded attributes are those         */
         /* containing                                         */
         /*                                                     */
         /*     - 3270 or extended attribute data              */
         /*     - SCA data                                     */
         /*     - attributes with no DFLD association          */
         /*       (filler attributes)                          */
         /*****************************************************/

         switch( pLpg->getClassType()) {
         case HWS_USER:
            clName = pLpg->getClassName();
            // DEBUG: HWSMsgGen::genDTWCPPLPageList
            // DEBUG: HWSLPageGen::genDTWCPPLPageList
            if( !strcmp( clName, "LPOUTAOut" ) ) {
               LPageList = new HWSDList( "PAYOUT&LPOUTA" );
            }
            // DEBUG: HWSLPageGen::genDTWCPPLPageList
            if( !strcmp( clName, "LPOUTBOut" ) ) {
               LPageList = new HWSDList( "PAYOUT&LPOUTB" );
            }
            // DEBUG: HWSLPageGen::genDTWCPPLPageList
            if( !strcmp( clName, "LPOUTCOut" ) ) {
               LPageList = new HWSDList( "PAYOUT&
LPOUTC" );
            }
            /*****************************************************/
            /* Add HWSLNode objects for displayable fields.      */
            /*****************************************************/
            if( LPageList ) {
               aNode = pLpg->genDisplayList();
               while( aNode ) {
                  newNode = new HWSLNode( aNode->name,
                                          aNode->length,
                                          aNode->data );
                  LPageList->addNode( newNode );
                  aNode = aNode->next;
               } /* end while */
            } /* end if */
         break;
         case HWS_ERROUT:
            LPageList = new HWSDList( "HWSErrOut" );
            /*****************************************************/
            /* Add HWSLNode objects for displayable fields.      */
            /* (OK to use genList because all fields in HWSErrOut */
            /* are displayable fields).                          */
            /*****************************************************/
            if( LPageList ) {
               aNode = pLpg->genList();
               while( aNode ) {
                  newNode = new HWSLNode( aNode->name,
                                          aNode->length,
                                          aNode->data );
                  LPageList->addNode( newNode );
                  aNode = aNode->next;
               } /* end while */
            } /* end if */
         break;
         case HWS_MO1:
            LPageList = new HWSDList( "DFSM01&
HWSMO1" );
            /*****************************************************/
            /* Add HWSLNode objects for displayable fields.      */
```

```
            /* (OK to use genList because all fields in HWSMO1    */
            /* are displayable fields).                           */
            /******************************************************/
            if( LPageList ) {
               aNode = pLpg->genList();
               while( aNode ) {
                  newNode = new HWSLNode( aNode->name,
                                          aNode->length,
                                          aNode->data );
                  LPageList->addNode( newNode );
                  aNode = aNode->next;
               } /* end while */
            } /* end if */
         break;
         } /* end switch */

         /******************************************************/
         /* Add HWSLNode objects for displayable fields.       */
         /******************************************************/
         if( !LPageList ) {
            memset( errCode, 0, sizeof( errCode ) );
            _itoa( HWS_NULL_TRANOUT_LIST, errCode, 10 );

            LPageList = new HWSDList( "DTW_Error" );
            newNode   = new HWSLNode( "DTW_Error", strlen( errCode ), errCode );

            LPageList->addNode( newNode );
            allLPages = LPageList;
            break;
         }

         /******************************************************/
         /* Add newly created lpage list to chain of DLists.   */
         /******************************************************/
         if( allLPages ) {
            allLPages->addList( LPageList );
         } else {
            allLPages = LPageList;
         }
         qLpg = pLpg;
         pLpg = qLpg->getNext();
         free(qLpg);

      } /* end while */

   } /* end if */

   return allLPages;

}
```

DTWpayf.mak, generated make file

```
# DEBUG: HWSMFSGen::genDTWMAK
#####################################
# IMS Web Studio Generated NMAKE File #
#####################################

#####################################
# This makefile was generated        #
# for NT and MS VC++ compiler.       #
#####################################

!IF "$(CFG)" != "HWSRel" && "$(CFG)" != "HWSDbg" && "$(CFG)" != "HWSTrap"
!MESSAGE Invalid configuration "$(CFG)" specified.
!MESSAGE You can specify a configuration when running NMAKE on this makefile
!MESSAGE by defining the macro CFG on the command line.  For example:
```

```
!MESSAGE
!MESSAGE NMAKE /f "DTWpayf.mak" CFG="HWSDbg"
!MESSAGE
!MESSAGE Possible choices for configuration are:
!MESSAGE
!MESSAGE "HWSRel" : build a PRODUCTION version
!MESSAGE "HWSDbg" : build a DEBUG version
!MESSAGE "HWSTrap" : build with TRAP and DEBUG
!MESSAGE
!ERROR An invalid configuration is specified.
!ENDIF

!IF "$(OS)" == "Windows_NT"
NULL=
!ELSE
NULL=nul
!ENDIF

!IF "$(CFG)" == "HWSRel"
CFLAGS= /nologo /MD /W3 /GX /O2 /D "WIN32" /D "NDEBUG" /D "_CONSOLE" /D "HWSNT4" /YX /c
LFLAGS= /nologo /subsystem:windows /dll /incremental:no /machine:
IX86 /def:"DTWpayf.def" /implib:"DTWpayf.lib" /out:".\DTWpayf.dll"
!ENDIF
!IF "$(CFG)" == "HWSDbg"
CFLAGS=/nologo /MDd /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D "_CONSOLE" /D "HWSNT4" /YX /c
LFLAGS= /nologo /subsystem:windows /dll /incremental:no
/debug /machine:IX86 /def:"DTWpayf.def" /implib:"DTWpayf.lib" /out:".\DTWpayf.dll"
!ENDIF
!IF "$(CFG)" == "HWSTrap"
CFLAGS=/nologo /MDd /W3 /Gm /GX /Zi /Od /D "WIN32" /D "_DEBUG" /D "_TRAP"
/D "_CONSOLE" /D "HWSNT4" /YX /c
LFLAGS= /nologo /subsystem:windows /dll /incremental:no /debug
/machine:IX86 /def:"DTWpayf.def" /implib:"DTWpayf.lib" /out:".\DTWpayf.dll"
!ENDIF

################################################################################
# Begin Project

ALL : ".\DTWpayf.dll"

LINK_OBJS= \
   "LPIN.obj"\
   "LPOUTA.obj"\
   "LPOUTB.obj"\
   "LPOUTC.obj"\
   "LPOUT1.obj"\
   "LPOUT2.obj"\
   "LPOUT3.obj"\
   "DTWpayf.obj"

LINK=link.exe
COMPILE=cl.exe

LINK_LIBS= \
 HWSTran.lib HWSCom.lib HWSUtil.lib HWSFMT.lib HWSMFS.lib

.cpp.obj:
 $(COMPILE) $(CFLAGS) %s

".\DTWpayf.dll" : \
    $(LINK_OBJS)
    $(LINK) @<<
     $(LFLAGS)
     $(LINK_LIBS)
```

```
    $(LINK_OBJS)
<<

#############  END OF FILE  #############
```

LPIN.hpp, generated input LPAGE interface file

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if it has been advised of the         */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not offered for resale                        */
/*                                                                   */
/*********************************************************************/

            // DEBUG: HWSLPageGen::genHHead
#ifndef LPIN_HPP
#define LPIN_HPP

#include "hwstin.hpp"
#include "hwsdmod.hpp"

            // DEBUG: HWSSegGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSLPageGen::genHDefs
            // DEBUG: HWSSegGen::genHDefs
            // DEBUG: HWSFldGen::genHDefs
#define LPININ_MAX_HWSMF001 5
            // DEBUG: HWSFldGen::genHDefs
#define LPININ_MAX_LNAME 10
            // DEBUG: HWSFldGen::genHDefs
#define LPININ_MAX_FNAME 10
            // DEBUG: HWSFldGen::genHDefs
#define LPININ_MAX_EMPNO 5
            // DEBUG: HWSFldGen::genHDefs
#define LPININ_MAX_SSN 11
            // DEBUG: HWSFldGen::genHDefs
#define LPININ_MAX_RATE 9

            // DEBUG: HWSLPageGen::genHDecl
/*********************************************************/
```

```
/*                                                          */
/*  IMS Web generated transaction input class definition.*/
/*                                                          */
/*  This class :                                            */
/*                                                          */
/*   - provides the execute method which compiles a         */
/*     list of all attribute values and calls the server */
/*     DLL for building and passing the message to IMS.  */
/*                                                          */
/************************************************************/

class LPINIn : public HWSTranIn {

            // DEBUG: HWSSegGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSLPageGen::genHPublic
   public:

      // LifeCycle
      LPINIn();
      ~LPINIn();

      // Overloaded Method
      virtual HWSTranOut * execute();

      // Access Methods for Fields
            // DEBUG: HWSSegGen::genHPublic
            // DEBUG: HWSFldGen::genHPublic
            // DEBUG: HWSFldGen::genHPublic
      void setLNAME(char *);
      inline char * getLNAME()    {return(LNAME);};
            // DEBUG: HWSFldGen::genHPublic
      void setFNAME(char *);
      inline char * getFNAME()    {return(FNAME);};
            // DEBUG: HWSFldGen::genHPublic
      void setEMPNO(char *);
      inline char * getEMPNO()    {return(EMPNO);};
            // DEBUG: HWSFldGen::genHPublic
      void setSSN(char *);
      inline char * getSSN()    {return(SSN);};
            // DEBUG: HWSFldGen::genHPublic
      void setRATE(char *);
      inline char * getRATE()    {return(RATE);};

            // DEBUG: HWSLPageGen::genHPrivate
   private:

      virtual HWSDList * export();

            // DEBUG: HWSSegGen::genHPrivate
            // DEBUG: HWSFldGen::genHPrivate
      char HWSMF001[LPININ_MAX_HWSMF001+1];
            // DEBUG: HWSFldGen::genHPrivate
      char LNAME[LPININ_MAX_LNAME+1];
            // DEBUG: HWSFldGen::genHPrivate
      char FNAME[LPININ_MAX_FNAME+1];
            // DEBUG: HWSFldGen::genHPrivate
      char EMPNO[LPININ_MAX_EMPNO+1];
            // DEBUG: HWSFldGen::genHPrivate
      char SSN[LPININ_MAX_SSN+1];
            // DEBUG: HWSFldGen::genHPrivate
      char RATE[LPININ_MAX_RATE+1];
```

```
            // DEBUG: HWSLPageGen::genHTail
            // DEBUG: HWSSegGen::genHTail
            // DEBUG: HWSFldGen::genHTail
            // DEBUG: HWSFldGen::genHTail
            // DEBUG: HWSFldGen::genHTail
            // DEBUG: HWSFldGen::genHTail
            // DEBUG: HWSFldGen::genHTail
            // DEBUG: HWSLPageGen::genHTail
};

#endif
```

LPIN.cpp, generated input LPAGE implementation file

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if it has been advised of the         */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not offered for resale                        */
/*                                                                   */
/*********************************************************************/

            // DEBUG: HWSLPageGen::genCHead
#include #include #include #include "hws.h"
#include "hwstran.hpp"
#include "hwsmfsf.hpp"
#include "hwsmsgf.hpp"
#include "hwslpgf.hpp"
#include "hwssegf.hpp"
#include "hwsfldf.hpp"
#include "hwsdlist.hpp"
#include "hwslnode.hpp"
#include "LPIN.hpp"
            // DEBUG: HWSMFSGen::genInclMod
            // DEBUG: HWSMsgGen::genInclMod
            // DEBUG: HWSLPageGen::genInclMod
#include "LPOUTA.hpp"
            // DEBUG: HWSLPageGen::genInclMod
#include "LPOUTB.hpp"
            // DEBUG: HWSLPageGen::genInclMod
#include "LPOUTC.hpp"
            // DEBUG: HWSMsgGen::genInclMod
            // DEBUG: HWSLPageGen::genInclMod
#include "LPOUT1.hpp"
```

```
            // DEBUG: HWSLPageGen::genInclMod
#include "LPOUT2.hpp"
            // DEBUG: HWSLPageGen::genInclMod
#include "LPOUT3.hpp"

            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
/*********************************************************/
/*                                                       */
/*  IMS Web generated transaction input class.           */
/*                                                       */
/*  This class :                                         */
/*                                                       */
/*   - provides the execute method which compiles a      */
/*     list of all attribute values and calls the server */
/*     DLL for building and passing the message to IMS.  */
/*                                                       */
/*********************************************************/

            // DEBUG: HWSLPageGen::genCConst
// LifeCycle
LPINIn::LPINIn():
   HWSTranIn("CGIpayf")
{

   HWSMFSFormat   *aFmt;
   HWSMsgFormat *aMsg;
   HWSLPageFormat *aLpg;
   HWSSegFormat *aSeg;
   HWSFldFormat *aFld;
   Condition   *aCond;

            // DEBUG: HWSMFSGen::genConst
   aFmt = new HWSMFSFormat();

   aFmt->verLevel = HWS110;

   aFmt->forLevel = HWSFMT1;

   aFmt->convType = USENGLISH;

            // DEBUG: HWSMsgGen::genConst
   aMsg = new HWSMsgFormat("PAYIN",OPT1,HWSIN);
   aFmt->addMID(aMsg);

            // DEBUG: HWSLPageGen::genConst
   aCond = NULL;
   aLpg = new HWSLPageFormat("LPIN",NOCOND,aCond);
   aMsg->addLPage(aLpg);

            // DEBUG: HWSSegGen::genConst
   aSeg = new HWSSegFormat("HWSSEG01");
   aLpg->addSeg(aSeg);

            // DEBUG: HWSFldGen::genConst
   memcpy(HWSMF001,"SKS1 ",5);
   memset(HWSMF001+5, '\0', 1);

   aFld = new HWSFldFormat("HWSMF001",FLDLIT,5);

   aFld->pp = 0;
   aFld->justify = HWSLEFT;
```

```
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 5;
aFld->literalVal = (char *) malloc (5+1);
memcpy(aFld->literalVal,"SKS1 ",5);
memset(aFld->literalVal+5, '\0', 1);

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
LNAME[0] = '\0';

aFld = new HWSFldFormat("HWSMF002",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
FNAME[0] = '\0';

aFld = new HWSFldFormat("HWSMF003",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
EMPNO[0] = '\0';

aFld = new HWSFldFormat("HWSMF004",STANDARD,7);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSCHAR;
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 1;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
SSN[0] = '\0';

aFld = new HWSFldFormat("HWSMF005",STANDARD,11);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSCHAR;
```

```
aFld->fill = ' ';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
RATE[0] = '\0';

aFld = new HWSFldFormat("HWSMF006",STANDARD,9);

aFld->pp = 0;
aFld->justify = HWSRIGHT;
aFld->fillType = HWSCHAR;
aFld->fill = '0';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSMsgGen::genConst
aMsg = new HWSMsgFormat("PAYOUT",OPT1,HWSOUT);
aFmt->addMOD(aMsg);

         // DEBUG: HWSLPageGen::genConst
aCond = (Condition *)malloc(sizeof(Condition));
aCond->MFLDName = (char *)malloc(strlen("LASTN")+1);
strcpy(aCond->MFLDName,"LASTN");
aCond->pp = 0;
aCond->op = HWSEQ;
aCond->value = (char *)malloc(strlen("Hill      ")+1);
strcpy(aCond->value,"Hill      ");
aLpg = new HWSLPageFormat("LPOUTA",MFLDNM,aCond);
aMsg->addLPage(aLpg);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG02");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("LASTN",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF007",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
```

```
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG03");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF008",STANDARD,7);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 1;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF009",STANDARD,11);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG04");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF010",STANDARD,9);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG05");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF011",STANDARD,50);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
```

```
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF012",SLITDAT1,6);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSLPageGen::genConst
aCond = (Condition *)malloc(sizeof(Condition));
aCond->MFLDName = (char *)malloc(strlen("LASTN")+1);
strcpy(aCond->MFLDName,"LASTN");
aCond->pp = 0;
aCond->op = HWSEQ;
aCond->value = (char *)malloc(strlen("Lin       ")+1);
strcpy(aCond->value,"Lin       ");
aLpg = new HWSLPageFormat("LPOUTB",MFLDNM,aCond);
aMsg->addLPage(aLpg);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG06");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("LASTN",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF013",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG07");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
```

```
aFld = new HWSFldFormat("HWSMF014",STANDARD,7);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 1;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF015",STANDARD,11);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG08");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF016",STANDARD,50);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF017",SLITDAT2,8);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSLPageGen::genConst
aCond = (Condition *)malloc(sizeof(Condition));
aCond->MFLDName = (char *)malloc(strlen("LASTN")+1);
strcpy(aCond->MFLDName,"LASTN");
aCond->pp = 0;
aCond->op = HWSEQ;
aCond->value = (char *)malloc(strlen("Chung     ")+1);
strcpy(aCond->value,"Chung     ");
```

```
aLpg = new HWSLPageFormat("LPOUTC",MFLDNM,aCond);
aMsg->addLPage(aLpg);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG09");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("LASTN",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF018",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG10");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF019",STANDARD,50);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF020",SLITDAT3,8);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);
```

```
         // DEBUG: HWSMsgGen::genConst
aMsg = new HWSMsgFormat("PAYOUT9",OPT1,HWSOUT);
aFmt->addMOD(aMsg);

         // DEBUG: HWSLPageGen::genConst
aCond = (Condition *)malloc(sizeof(Condition));
aCond->MFLDName = (char *)malloc(strlen("LASTN")+1);
strcpy(aCond->MFLDName,"LASTN");
aCond->pp = 0;
aCond->op = HWSEQ;
aCond->value = (char *)malloc(strlen("Hill      ")+1);
strcpy(aCond->value,"Hill      ");
aLpg = new HWSLPageFormat("LPOUT1",MFLDNM,aCond);
aMsg->addLPage(aLpg);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG11");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("LASTN",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG12");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF021",STANDARD,7);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 1;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF022",STANDARD,11);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG13");
```

```
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF023",STANDARD,9);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG14");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF024",STANDARD,50);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF025",SLITDAT1,6);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSLPageGen::genConst
aCond = (Condition *)malloc(sizeof(Condition));
aCond->MFLDName = (char *)malloc(strlen("LASTN")+1);
strcpy(aCond->MFLDName,"LASTN");
aCond->pp = 0;
aCond->op = HWSEQ;
aCond->value = (char *)malloc(strlen("Lin       ")+1);
strcpy(aCond->value,"Lin       ");
aLpg = new HWSLPageFormat("LPOUT2",MFLDNM,aCond);
aMsg->addLPage(aLpg);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG15");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("LASTN",STANDARD,10);

aFld->pp = 0;
```

```
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG16");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF026",STANDARD,7);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 1;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF027",STANDARD,11);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG17");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF028",STANDARD,50);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF029",SLITDAT2,8);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
```

```
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSLPageGen::genConst
aCond = (Condition *)malloc(sizeof(Condition));
aCond->MFLDName = (char *)malloc(strlen("LASTN")+1);
strcpy(aCond->MFLDName,"LASTN");
aCond->pp = 0;
aCond->op = HWSEQ;
aCond->value = (char *)malloc(strlen("Chung     ")+1);
strcpy(aCond->value,"Chung     ");
aLpg = new HWSLPageFormat("LPOUT3",MFLDNM,aCond);
aMsg->addLPage(aLpg);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG18");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("LASTN",STANDARD,10);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSSegGen::genConst
aSeg = new HWSSegFormat("HWSSEG19");
aLpg->addSeg(aSeg);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF030",STANDARD,50);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);

         // DEBUG: HWSFldGen::genConst
aFld = new HWSFldFormat("HWSMF031",SLITDAT3,8);

aFld->pp = 0;
aFld->justify = HWSLEFT;
aFld->fillType = HWSNULL;
aFld->fill = '\x3F';
aFld->A3270Attr = NO3270;
aFld->eAttr = 0;
aFld->literalLen = 0;
aFld->literalVal = NULL;

aSeg->addFld(aFld);
```

```
            // DEBUG: HWSLPageGen::genCConst
   formatObj = aFmt;

}

            // DEBUG: HWSLPageGen::genCDest
LPINIn::~LPINIn()
{
            // DEBUG: HWSSegGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSLPageGen::genCDest
}

            // DEBUG: HWSLPageGen::genCExec
// Overloaded Method
HWSTranOut * LPINIn::execute()
{
   HWSTranOut * aMod = NULL;
   HWSTranOut * pMod = NULL;
   HWSTranOut * qMod = NULL;
   HWSTran    aTran(this);
   HWSDList   * pList;
   HWSDList   * qList;
   int errNum = 0;

   pList = aTran.doIMS(export(),formatObj);

   if (pList) {
      while (pList) {
         if (strcmp(pList->MDName,"HWSErrOut") == 0) {
            pMod = new HWSErrOut(pList);
            // DEBUG: HWSMFSGen::genCExecOut
            // DEBUG: HWSMsgGen::genCExecOut
            // DEBUG: HWSLPageGen::genCExecOut
         } else if (strcmp(pList->MDName,"PAYOUT&LPOUTA") == 0) {
            pMod = new LPOUTAOut(pList);
            // DEBUG: HWSLPageGen::genCExecOut
         } else if (strcmp(pList->MDName,"PAYOUT&LPOUTB") == 0) {
            pMod = new LPOUTBOut(pList);
            // DEBUG: HWSLPageGen::genCExecOut
         } else if (strcmp(pList->MDName,"PAYOUT&LPOUTC") == 0) {
            pMod = new LPOUTCOut(pList);
            // DEBUG: HWSMsgGen::genCExecOut
            // DEBUG: HWSLPageGen::genCExecOut
         } else if (strcmp(pList->MDName,"PAYOUT9&LPOUT1") == 0) {
            pMod = new LPOUT1Out(pList);
            // DEBUG: HWSLPageGen::genCExecOut
         } else if (strcmp(pList->MDName,"PAYOUT9&LPOUT2") == 0) {
            pMod = new LPOUT2Out(pList);
            // DEBUG: HWSLPageGen::genCExecOut
         } else if (strcmp(pList->MDName,"PAYOUT9&LPOUT3") == 0) {
            pMod = new LPOUT3Out(pList);
            // DEBUG: HWSLPageGen::genCExec
         } else if (strcmp(pList->MDName,"DFSM01&HWSMO1") == 0) {
            pMod = new HWSMO1(pList);
         } else if (strcmp(pList->MDName,"DFSM02&HWSMO2") == 0) {
            pMod = new HWSMO2(pList);
         } else if (strcmp(pList->MDName,"DFSM03&HWSMO3") == 0) {
            pMod = new HWSMO3(pList);
         } else if (strcmp(pList->MDName,"DFSM04&HWSMO4") == 0) {
            pMod = new HWSMO4(pList);
```

```
         } else if (strcmp(pList->MDName,"DFSM05&HWSMO5") == 0) {
            pMod = new HWSMO5(pList);
         } else if (strcmp(pList->MDName,"DFSDSP01&
HWSDSPO1") == 0) {
            pMod = new HWSDSPO1(pList);
         } else {
            pMod = new HWSErrOut(1000,"AN UNKNOWN MOD RETURNED BY HOST");
         } /* end if */
         if (!aMod) aMod = pMod;
         if (qMod) qMod->setNext(pMod);
         qMod = pMod;
         qList = pList;
         pList = qList->getNextList();
      } /* end while */
   } else {
      pMod = new HWSErrOut(1001,"SEVERE ERROR NULL RETURNED FROM DLLs");
   } /* end if */

            // DEBUG: HWSSegGen::genCExec
            // DEBUG: HWSFldGen::genCExec
            // DEBUG: HWSFldGen::genCExec
            // DEBUG: HWSFldGen::genCExec
            // DEBUG: HWSFldGen::genCExec
            // DEBUG: HWSFldGen::genCExec
            // DEBUG: HWSFldGen::genCExec
            // DEBUG: HWSLPageGen::genCExec
   return (aMod);

}

            // DEBUG: HWSLPageGen::genCExport
// Private Method Export
HWSDList * LPINIn::export()
{
   HWSDList  * expList;
   HWSLNode  * aFld;
   char   * iStr;
   int      iStrLen;

   expList = new HWSDList(formatObj->firstMID->name);

            // DEBUG: HWSSegGen::genCExport
            // DEBUG: HWSFldGen::genCExport
   iStrLen = strlen(HWSMF001);
   if (iStrLen > 0) {
      iStr = (char *)malloc(iStrLen + 1);
      strcpy(iStr,HWSMF001);
   } else {
      iStr = NULL;
      iStrLen = 0;
   }
   aFld = new HWSLNode("HWSMF001",iStrLen,iStr);
   expList->addNode(aFld);

            // DEBUG: HWSFldGen::genCExport
   iStrLen = strlen(LNAME);
   if (iStrLen > 0) {
      iStr = (char *)malloc(iStrLen + 1);
      strcpy(iStr,LNAME);
   } else {
      iStr = NULL;
      iStrLen = 0;
   }
   aFld = new HWSLNode("HWSMF002",iStrLen,iStr);
   expList->addNode(aFld);

            // DEBUG: HWSFldGen::genCExport
```

```
   iStrLen = strlen(FNAME);
   if (iStrLen > 0) {
      iStr = (char *)malloc(iStrLen + 1);
      strcpy(iStr,FNAME);
   } else {
      iStr = NULL;
      iStrLen = 0;
   }
   aFld = new HWSLNode("HWSMF003",iStrLen,iStr);
   expList->addNode(aFld);

            // DEBUG: HWSFldGen::genCExport
   iStrLen = strlen(EMPNO);
   if (iStrLen > 0) {
      iStr = (char *)malloc(iStrLen + 1);
      strcpy(iStr,EMPNO);
   } else {
      iStr = NULL;
      iStrLen = 0;
   }
   aFld = new HWSLNode("HWSMF004",iStrLen,iStr);
   expList->addNode(aFld);

            // DEBUG: HWSFldGen::genCExport
   iStrLen = strlen(SSN);
   if (iStrLen > 0) {
      iStr = (char *)malloc(iStrLen + 1);
      strcpy(iStr,SSN);
   } else {
      iStr = NULL;
      iStrLen = 0;
   }
   aFld = new HWSLNode("HWSMF005",iStrLen,iStr);
   expList->addNode(aFld);

            // DEBUG: HWSFldGen::genCExport
   iStrLen = strlen(RATE);
   if (iStrLen > 0) {
      iStr = (char *)malloc(iStrLen + 1);
      strcpy(iStr,RATE);
   } else {
      iStr = NULL;
      iStrLen = 0;
   }
   aFld = new HWSLNode("HWSMF006",iStrLen,iStr);
   expList->addNode(aFld);

            // DEBUG: HWSLPageGen::genCExport
   return (expList);

}

            // DEBUG: HWSLPageGen::genCHtml
            // DEBUG: HWSLPageGen::genCList
            // DEBUG: HWSLPageGen::genCDispList
            // DEBUG: HWSLPageGen::genCSetOper
// Access Methods for Fields
            // DEBUG: HWSSegGen::genCSetOper
            // DEBUG: HWSFldGen::genCSetOper
            // DEBUG: HWSFldGen::genCSetOper
void LPINIn::setLNAME(char * value)
{
   memset(LNAME,'\0',LPININ_MAX_LNAME+1);
   if (*value) {
      memcpy(LNAME,value,(strlen(value)<=LPININ_MAX_LNAME)?strlen(value):LPININ_MAX_LNAME);
   } /* end if */
}
```

```
            // DEBUG: HWSFldGen::genCSetOper
void LPINIn::setFNAME(char * value)
{
   memset(FNAME,'\0',LPININ_MAX_FNAME+1);
   if (*value) {
      memcpy(FNAME,value,(strlen(value)<=LPININ_MAX_FNAME)?strlen(value):LPININ_MAX_FNAME);
   } /* end if */
}

            // DEBUG: HWSFldGen::genCSetOper
void LPINIn::setEMPNO(char * value)
{
   memset(EMPNO,'\0',LPININ_MAX_EMPNO+1);
   if (*value) {
      memcpy(EMPNO,value,(strlen(value)<=LPININ_MAX_EMPNO)?strlen(value):LPININ_MAX_EMPNO);
   } /* end if */
}

            // DEBUG: HWSFldGen::genCSetOper
void LPINIn::setSSN(char * value)
{
   memset(SSN,'\0',LPININ_MAX_SSN+1);
   if (*value) {
      memcpy(SSN,value,(strlen(value)<=LPININ_MAX_SSN)?strlen(value):LPININ_MAX_SSN);
   } /* end if */
}

            // DEBUG: HWSFldGen::genCSetOper
void LPINIn::setRATE(char * value)
{
   memset(RATE,'\0',LPININ_MAX_RATE+1);
   if (*value) {
      memcpy(RATE,value,(strlen(value)<=LPININ_MAX_RATE)?strlen(value):LPININ_MAX_RATE);
   } /* end if */
}
```

LPOUTA.hpp, generated output LPAGE interface file

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if it has been advised of the         */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not offered for resale                        */
/*                                                                   */
/*********************************************************************/
```

```
            // DEBUG: HWSLPageGen::genHHead
#ifndef LPOUTA_HPP
#define LPOUTA_HPP

#include "hwstout.hpp"

            // DEBUG: HWSSegGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSSegGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSSegGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSSegGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSFldGen::genHHead
            // DEBUG: HWSLPageGen::genHDefs
            // DEBUG: HWSSegGen::genHDefs
            // DEBUG: HWSFldGen::genHDefs
#define LPOUTAOUT_MAX_LNAME 10
            // DEBUG: HWSFldGen::genHDefs
#define LPOUTAOUT_MAX_FNAME 10

            // DEBUG: HWSSegGen::genHDefs
            // DEBUG: HWSFldGen::genHDefs
#define LPOUTAOUT_MAX_EMPNO 5
#define LPOUTAOUT_MAX_EMPNO_ATTR 2
            // DEBUG: HWSFldGen::genHDefs
#define LPOUTAOUT_MAX_SSN 11

            // DEBUG: HWSSegGen::genHDefs
            // DEBUG: HWSFldGen::genHDefs
#define LPOUTAOUT_MAX_RATE 9

            // DEBUG: HWSSegGen::genHDefs
            // DEBUG: HWSFldGen::genHDefs
#define LPOUTAOUT_MAX_MSGFLD 50
            // DEBUG: HWSFldGen::genHDefs
#define LPOUTAOUT_MAX_DATE 6

            // DEBUG: HWSLPageGen::genHDecl
/*********************************************************/
/*                                                       */
/*  IMS Web generated transaction output class definition.*/
/*                                                       */
/*  This class :                                         */
/*                                                       */
/*   - provides the genHTML method which generates the   */
/*     HTML output based on the result of IMS tran.      */
/*                                                       */
/*********************************************************/

class LPOUTAOut : public HWSTranOut {

            // DEBUG: HWSSegGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSSegGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSSegGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSSegGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
            // DEBUG: HWSFldGen::genHDecl
```

```
        // DEBUG: HWSLPageGen::genHPublic
public:

   // LifeCycle
   LPOUTAOut(HWSDList * fldList);
   ~LPOUTAOut();

   // Overloaded Method
   virtual char     * genHTML();
   virtual HWSLNode * genList();
   virtual HWSLNode * genDisplayList();

   // Access Methods for Fields
         // DEBUG: HWSSegGen::genHPublic
         // DEBUG: HWSFldGen::genHPublic
   inline char * getLNAME()    {return(LNAME);};
         // DEBUG: HWSFldGen::genHPublic
   inline char * getFNAME()    {return(FNAME);};
         // DEBUG: HWSSegGen::genHPublic
         // DEBUG: HWSFldGen::genHPublic
   inline char * getEMPNO_Attr()    {return(EMPNO_Attr);};
   inline char * getEMPNO()    {return(EMPNO);};
         // DEBUG: HWSFldGen::genHPublic
   inline char * getSSN()    {return(SSN);};
         // DEBUG: HWSSegGen::genHPublic
         // DEBUG: HWSFldGen::genHPublic
   inline char * getRATE()    {return(RATE);};
         // DEBUG: HWSSegGen::genHPublic
         // DEBUG: HWSFldGen::genHPublic
   inline char * getMSGFLD()    {return(MSGFLD);};
         // DEBUG: HWSFldGen::genHPublic
   inline char * getDATE()    {return(DATE);};

         // DEBUG: HWSLPageGen::genHPrivate
private:

   HWSDList * fList;

         // DEBUG: HWSSegGen::genHPrivate
         // DEBUG: HWSFldGen::genHPrivate
   char LNAME[LPOUTAOUT_MAX_LNAME+1];
         // DEBUG: HWSFldGen::genHPrivate
   char FNAME[LPOUTAOUT_MAX_FNAME+1];
         // DEBUG: HWSSegGen::genHPrivate
         // DEBUG: HWSFldGen::genHPrivate
   char EMPNO_Attr[LPOUTAOUT_MAX_EMPNO_ATTR+1];
   char EMPNO[LPOUTAOUT_MAX_EMPNO+1];
         // DEBUG: HWSFldGen::genHPrivate
   char SSN[LPOUTAOUT_MAX_SSN+1];
         // DEBUG: HWSSegGen::genHPrivate
         // DEBUG: HWSFldGen::genHPrivate
   char RATE[LPOUTAOUT_MAX_RATE+1];
         // DEBUG: HWSSegGen::genHPrivate
         // DEBUG: HWSFldGen::genHPrivate
   char MSGFLD[LPOUTAOUT_MAX_MSGFLD+1];
         // DEBUG: HWSFldGen::genHPrivate
   char DATE[LPOUTAOUT_MAX_DATE+1];

         // DEBUG: HWSLPageGen::genHTail
         // DEBUG: HWSSegGen::genHTail
         // DEBUG: HWSFldGen::genHTail
         // DEBUG: HWSFldGen::genHTail
         // DEBUG: HWSSegGen::genHTail
         // DEBUG: HWSFldGen::genHTail
         // DEBUG: HWSFldGen::genHTail
         // DEBUG: HWSSegGen::genHTail
         // DEBUG: HWSFldGen::genHTail
```

```
            // DEBUG: HWSSegGen::genHTail
            // DEBUG: HWSFldGen::genHTail
            // DEBUG: HWSFldGen::genHTail
            // DEBUG: HWSLPageGen::genHTail
};

#endif
```

LPOUTA.cpp, generated output LPAGE implementation file

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if it has been advised of the         */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not offered for resale                        */
/*                                                                   */
/*********************************************************************/
          // DEBUG: HWSLPageGen::genCHead
#include #include #include #include "hws.h"
#include "hwstran.hpp"
#include "hwsmfsf.hpp"
#include "hwsmsgf.hpp"
#include "hwslpgf.hpp"
#include "hwssegf.hpp"
#include "hwsfldf.hpp"
#include "hwsdlist.hpp"
#include "hwslnode.hpp"
#include "LPOUTA.hpp"

            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
/*****************************************************/
/*                                                   */
/*  IMS Web generated transaction output class.      */
/*                                                   */
/*  This class :                                     */
/*                                                   */
```

```
/*   - provides the genHTML method which generates the   */
/*     HTML output based on the result of IMS tran.      */
/*                                                       */
/*******************************************************/

            // DEBUG: HWSLPageGen::genCConst
// LifeCycle
LPOUTAOut::LPOUTAOut(HWSDList * fldList):
   HWSTranOut("LPOUTAOut")
{

   HWSLNode * aFld;

   fList = fldList;

   aFld = fldList->getFirst();
            // DEBUG: HWSSegGen::genCConst
            // DEBUG: HWSFldGen::genCConst
   memset(LNAME,'\0',LPOUTAOUT_MAX_LNAME+1);
   memcpy(LNAME,aFld->data,(aFld->length <=
LPOUTAOUT_MAX_LNAME)?aFld->length:LPOUTAOUT_MAX_LNAME);
   aFld = aFld->next;

            // DEBUG: HWSFldGen::genCConst
   memset(FNAME,'\0',LPOUTAOUT_MAX_FNAME+1);
   memcpy(FNAME,aFld->data,(aFld->length <=
LPOUTAOUT_MAX_FNAME)?aFld->length:LPOUTAOUT_MAX_FNAME);
   aFld = aFld->next;

            // DEBUG: HWSSegGen::genCConst
            // DEBUG: HWSFldGen::genCConst
   memset(EMPNO_Attr,'\0',LPOUTAOUT_MAX_EMPNO_ATTR+1);
   memcpy(EMPNO_Attr,aFld->data,(aFld->length <=
LPOUTAOUT_MAX_EMPNO_ATTR)?aFld->length:LPOUTAOUT_MAX_EMPNO);
   aFld = aFld->next;
   memset(EMPNO,'\0',LPOUTAOUT_MAX_EMPNO+1);
   memcpy(EMPNO,aFld->data,(aFld->length <=
LPOUTAOUT_MAX_EMPNO)?aFld->length:LPOUTAOUT_MAX_EMPNO);
   aFld = aFld->next;

            // DEBUG: HWSFldGen::genCConst
   memset(SSN,'\0',LPOUTAOUT_MAX_SSN+1);
   memcpy(SSN,aFld->data,(aFld->length <= LPOUTAOUT_MAX_SSN)?aFld->length:LPOUTAOUT_MAX_SSN);
   aFld = aFld->next;
            // DEBUG: HWSSegGen::genCConst
            // DEBUG: HWSFldGen::genCConst
   memset(RATE,'\0',LPOUTAOUT_MAX_RATE+1);
   memcpy(RATE,aFld->data,(aFld->length <= LPOUTAOUT_MAX_RATE)?aFld->length:LPOUTAOUT_MAX_RATE);
   aFld = aFld->next;

            // DEBUG: HWSSegGen::genCConst
            // DEBUG: HWSFldGen::genCConst
   memset(MSGFLD,'\0',LPOUTAOUT_MAX_MSGFLD+1);
   memcpy(MSGFLD,aFld->data,(aFld->length <=
LPOUTAOUT_MAX_MSGFLD)?aFld->length:LPOUTAOUT_MAX_MSGFLD);
   aFld = aFld->next;

            // DEBUG: HWSFldGen::genCConst
   memset(DATE,'\0',LPOUTAOUT_MAX_DATE+1);
   memcpy(DATE,aFld->data,(aFld->length <=
LPOUTAOUT_MAX_DATE)?aFld->length:LPOUTAOUT_MAX_DATE);
   aFld = aFld->next;

            // DEBUG: HWSLPageGen::genCConst
}

            // DEBUG: HWSLPageGen::genCDest
```

```
LPOUTAOut::~LPOUTAOut()
{            // DEBUG: HWSSegGen::genCDest
             // DEBUG: HWSFldGen::genCDest
             // DEBUG: HWSFldGen::genCDest
             // DEBUG: HWSSegGen::genCDest
             // DEBUG: HWSFldGen::genCDest
             // DEBUG: HWSFldGen::genCDest
             // DEBUG: HWSSegGen::genCDest
             // DEBUG: HWSFldGen::genCDest
             // DEBUG: HWSSegGen::genCDest
             // DEBUG: HWSFldGen::genCDest
             // DEBUG: HWSFldGen::genCDest
             // DEBUG: HWSLPageGen::genCDest
}

             // DEBUG: HWSLPageGen::genCExec
             // DEBUG: HWSLPageGen::genCExport
             // DEBUG: HWSLPageGen::genCHtml
// Overloaded Method genHTML
char * LPOUTAOut::genHTML()
{
 char     * buf;
   HWSLNode * aFld;

   buf = (char *)malloc(sizeof(LPOUTAOut)*10);
   strcpy(buf,"<table border=5 cellpadding=6>\n");

   aFld = fList->getFirst();
            // DEBUG: HWSSegGen::genCHtml
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"<tr><td><strong>");
   strcat(buf,"LNAME");
   strcat(buf,"\n");
   strcat(buf,"<td>");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"<tr><td><strong>");
   strcat(buf,"FNAME");
   strcat(buf,"\n");
   strcat(buf,"<td>");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSSegGen::genCHtml
            // DEBUG: HWSFldGen::genCHtml
   /* skipping the extended attribute data*/
   aFld = aFld->next;
   strcat(buf,"<tr><td><strong>");
   strcat(buf,"EMPNO");
   strcat(buf,"\n");
   strcat(buf,"<td>");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"<tr><td><strong>");
   strcat(buf,"SSN");
   strcat(buf,"\n");
   strcat(buf,"<td>");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
```

```
   aFld = aFld->next;
            // DEBUG: HWSSegGen::genCHtml
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"<tr><td><strong>");
   strcat(buf,"RATE");
   strcat(buf,"\n");
   strcat(buf,"<td>");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSSegGen::genCHtml
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"<tr><td><strong>");
   strcat(buf,"MSGFLD");
   strcat(buf,"\n");
   strcat(buf,"<td>");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"<tr><td><strong>");
   strcat(buf,"DATE");
   strcat(buf,"\n");
   strcat(buf,"<td>");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSLPageGen::genCHtml
   strcat(buf,"\n");
   return buf;
}
            // DEBUG: HWSLPageGen::genCList
// Overloaded Method genList
HWSLNode * LPOUTAOut::genList()
{
   HWSLNode * fldList  = NULL;
   HWSLNode * aNode    = NULL;
   HWSLNode * newNode  = NULL;
   HWSLNode * prevNode  = NULL;
            // DEBUG: HWSSegGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSSegGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSSegGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSSegGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSLPageGen::genCList
   if (fList) {
      for (aNode = fList->getFirst();
           aNode != NULL;
           aNode = aNode->next) {
         newNode = new HWSLNode(aNode->name,aNode->length,aNode->data);
         if (prevNode) {
            prevNode->next = newNode;
         } else {
            fldList = newNode;
         } /* end if */
```

```
         prevNode = newNode;
      } /* end for */
   } /* end if */
   return fldList;

}

            // DEBUG: HWSLPageGen::genCDispList
// Overloaded Method genList
HWSLNode * LPOUTAOut::genDisplayList()
{
   HWSLNode * fldList  = NULL;
   HWSLNode * aNode    = NULL;
   HWSLNode * newNode  = NULL;
   HWSLNode * prevNode = NULL;
   int        addIt    = 0;

   if (fList) {
      for (aNode = fList->getFirst();
           aNode != NULL;
           aNode = aNode->next) {

         addIt = 0;
            // DEBUG: HWSSegGen::genCDispList
            // DEBUG: HWSFldGen::genCDispList
         if( !strcmp( aNode->name, "LASTN" ) ) {
            newNode = new HWSLNode( "LASTN",
                                    aNode->length,
                                    aNode->data );
            addIt = 1;
         }
            // DEBUG: HWSFldGen::genCDispList
         if( !strcmp( aNode->name, "HWSMF007" ) ) {
            newNode = new HWSLNode( "HWSMF007",
                                    aNode->length,
                                    aNode->data );
            addIt = 1;
         }
            // DEBUG: HWSSegGen::genCDispList
            // DEBUG: HWSFldGen::genCDispList
         if( !strcmp( aNode->name, "HWSMF008_ATTR" ) ) {
            addIt = 0;
         }
         if( !strcmp( aNode->name, "HWSMF008" ) ) {
            newNode = new HWSLNode( "HWSMF008",
                                    aNode->length,
                                    aNode->data );
            addIt = 1;
         }
            // DEBUG: HWSFldGen::genCDispList
         if( !strcmp( aNode->name, "HWSMF009" ) ) {
            newNode = new HWSLNode( "HWSMF009",
                                    aNode->length,
                                    aNode->data );
            addIt = 1;
         }
            // DEBUG: HWSSegGen::genCDispList
            // DEBUG: HWSFldGen::genCDispList
         if( !strcmp( aNode->name, "HWSMF010" ) ) {
            newNode = new HWSLNode( "HWSMF010",
                                    aNode->length,
```

```
                                    aNode->data );
            addIt = 1;
         }
            // DEBUG: HWSSegGen::genCDispList
            // DEBUG: HWSFldGen::genCDispList
         if( !strcmp( aNode->name, "HWSMF011" ) ) {
            newNode = new HWSLNode( "HWSMF011",
                                    aNode->length,
                                    aNode->data );
            addIt = 1;
         }
            // DEBUG: HWSFldGen::genCDispList
         if( !strcmp( aNode->name, "HWSMF012" ) ) {
            newNode = new HWSLNode( "HWSMF012",
                                    aNode->length,
                                    aNode->data );
            addIt = 1;
         }
            // DEBUG: HWSLPageGen::genCDispList
         if ( addIt ) {
            if( prevNode ) {
               prevNode->next = newNode;
            } else {
               fldList = newNode;
            } /* end if */
            prevNode = newNode;
         } /* end if */
      } /* end for */
   } /* end if */
   return fldList;

}

            // DEBUG: HWSLPageGen::genCSetOper
```

DTWpayf.mak

```
# DEBUG: HWSMFSGen::genDTWMAK
####################################
# IMS Web Studio Generated NMAKE File #
####################################

####################################
# This makefile was generated        #
# for NT and MS VC++ compiler.       #
####################################

!IF "$(CFG)" != "HWSRel" && "$(CFG)" != "HWSDbg" && "$(CFG)" != "HWSTrap"
!MESSAGE Invalid configuration "$(CFG)" specified.
!MESSAGE You can specify a configuration when running NMAKE on this makefile
!MESSAGE by defining the macro CFG on the command line.  For example:
!MESSAGE
!MESSAGE NMAKE /f "DTWpayf.mak" CFG="HWSDbg"
!MESSAGE
!MESSAGE Possible choices for configuration are:
!MESSAGE
!MESSAGE "HWSRel" : build a PRODUCTION version
!MESSAGE "HWSDbg" : build a DEBUG version
!MESSAGE "HWSTrap" : build with TRAP and DEBUG
!MESSAGE
!ERROR An invalid configuration is specified.
!ENDIF
```

```
!IF "$(OS)" == "Windows_NT"
NULL=
!ELSE
NULL=nul
!ENDIF

!IF "$(CFG)" == "HWSRel"
CFLAGS= /nologo /MD /W3 /GX /O2 /D "WIN32" /D "NDEBUG"
/D "_CONSOLE" /D "HWSNT4" /YX /c
LFLAGS= /nologo /subsystem:windows /dll /incremental:no
/machine:IX86 /def:"DTWpayf.def"
/implib:"DTWpayf.lib" /out:".
\DTWpayf.dll"
!ENDIF
!IF "$(CFG)" == "HWSDbg"
CFLAGS=/nologo /MDd /W3 /Gm /GX /Zi /Od /D "WIN32"
/D "_DEBUG" /D "_CONSOLE" /D "HWSNT4" /YX /c
LFLAGS= /nologo /subsystem:windows /dll /incremental:no
/debug /machine:IX86 /def:"DTWpayf.def"
/implib:"DTWpayf.lib" /out:".\DTWpayf.dll"
!ENDIF
!IF "$(CFG)" == "HWSTrap"
CFLAGS=/nologo /MDd /W3 /Gm /GX /Zi /Od /D "WIN32"
/D "_DEBUG" /D "_TRAP" /D "_CONSOLE"
/D "HWSNT4" /YX /c
LFLAGS= /nologo /subsystem:windows /dll /incremental:no
/debug /machine:IX86 /def:"DTWpayf.def" /implib:"DTWpayf.lib"
/out:".\DTWpayf.dll"
!ENDIF

################################################################################
# Begin Project

ALL : ".\DTWpayf.dll"

LINK_OBJS= \
   "LPIN.obj"\
   "LPOUTA.obj"\
   "LPOUTB.obj"\
   "LPOUTC.obj"\
   "LPOUT1.obj"\
   "LPOUT2.obj"\
   "LPOUT3.obj"\
   "DTWpayf.obj"

LINK=link.exe
COMPILE=cl.exe

LINK_LIBS= \
 HWSTran.lib HWSCom.lib HWSUtil.lib HWSFMT.lib HWSMFS.lib

.cpp.obj:
 $(COMPILE) $(CFLAGS) %s

".\DTWpayf.dll" : \
    $(LINK_OBJS)
    $(LINK) @<<
     $(LFLAGS)
     $(LINK_LIBS)
     $(LINK_OBJS)
<<

#############  END OF FILE  #############
```

LPOUTA.cpp, modified output LPAGE implementation file

```
/*********************************************************************/
/*                                                                   */
/* (c) Copyright IBM Corp. 1996                                      */
/* All Rights Reserved                                               */
/* Licensed Materials - Property of IBM                              */
/*                                                                   */
/* DISCLAIMER OF WARRANTIES.                                         */
/*                                                                   */
/* The following [enclosed] code is generated by a software product  */
/* of IBM Corporation.                                               */
/* This generated code is provided to you solely for the purpose of  */
/* assisting you in the development of your applications.            */
/* The code is provided "AS IS." IBM MAKES NO WARRANTIES, EXPRESS OR */
/* IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF   */
/* MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, REGARDING   */
/* THE FUNCTION OR PERFORMANCE OF THIS CODE.                         */
/* IBM shall not be liable for any damages arising out of your use   */
/* of the generated code, even if it has been advised of the         */
/* possibility of such damages.                                      */
/*                                                                   */
/* DISTRIBUTION.                                                     */
/*                                                                   */
/* This generated code can be freely distributed, copied, altered,   */
/* and incorporated into other software, provided that:              */
/*   - It bears the above Copyright notice and DISCLAIMER intact     */
/*   - The software is not offered for resale                        */
/*                                                                   */
/*********************************************************************/
            // DEBUG: HWSLPageGen::genCHead
#include #include #include #include "hws.h"
#include "hwstran.hpp"
#include "hwsmfsf.hpp"
#include "hwsmsgf.hpp"
#include "hwslpgf.hpp"
#include "hwssegf.hpp"
#include "hwsfldf.hpp"
#include "hwsdlist.hpp"
#include "hwslnode.hpp"
#include "LPOUTA.hpp"

            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSSegGen::genCHead
            // DEBUG: HWSFldGen::genCHead
            // DEBUG: HWSFldGen::genCHead
/*********************************************************/
/*                                                       */
/*  IMS Web generated transaction output class.          */
/*                                                       */
/*  This class :                                         */
/*                                                       */
/*   - provides the genHTML method which generates the   */
/*     HTML output based on the result of IMS tran.      */
/*                                                       */
/*********************************************************/


            // DEBUG: HWSLPageGen::genCConst
// LifeCycle
```

```
LPOUTAOut::LPOUTAOut(HWSDList * fldList):
   HWSTranOut("LPOUTAOut")
{

   HWSLNode * aFld;

   fList = fldList;

   aFld = fldList->getFirst();
            // DEBUG: HWSSegGen::genCConst
            // DEBUG: HWSFldGen::genCConst
   memset(LNAME,'\0',LPOUTAOUT_MAX_LNAME+1);
   memcpy(LNAME,aFld->data,(aFld->length <
= LPOUTAOUT_MAX_LNAME)?aFld->length:LPOUTAOUT_MAX_LNAME);
   aFld = aFld->next;

           // DEBUG: HWSFldGen::genCConst
   memset(FNAME,'\0',LPOUTAOUT_MAX_FNAME+1);
   memcpy(FNAME,aFld->data,(aFld->length <
= LPOUTAOUT_MAX_FNAME)?aFld->length:LPOUTAOUT_MAX_FNAME);
   aFld = aFld->next;

            // DEBUG: HWSSegGen::genCConst
            // DEBUG: HWSFldGen::genCConst
   memset(EMPNO_Attr,'\0',LPOUTAOUT_MAX_EMPNO_ATTR+1);
   memcpy(EMPNO_Attr,aFld->data,(aFld->length <
= LPOUTAOUT_MAX_EMPNO_ATTR)?aFld->length:LPOUTAOUT_MAX_EMPNO);
   aFld = aFld->next;
   memset(EMPNO,'\0',LPOUTAOUT_MAX_EMPNO+1);
   memcpy(EMPNO,aFld->data,(aFld->length <
= LPOUTAOUT_MAX_EMPNO)?aFld->length:LPOUTAOUT_MAX_EMPNO);
   aFld = aFld->next;

          // DEBUG: HWSFldGen::genCConst
   memset(SSN,'\0',LPOUTAOUT_MAX_SSN+1);
   memcpy(SSN,aFld->data,(aFld->length <
= LPOUTAOUT_MAX_SSN)?aFld->length:LPOUTAOUT_MAX_SSN);
   aFld = aFld->next;

            // DEBUG: HWSSegGen::genCConst
            // DEBUG: HWSFldGen::genCConst
   memset(RATE,'\0',LPOUTAOUT_MAX_RATE+1);
   memcpy(RATE,aFld->data,(aFld->length <
= LPOUTAOUT_MAX_RATE)?aFld->length:LPOUTAOUT_MAX_RATE);
   aFld = aFld->next;

           // DEBUG: HWSSegGen::genCConst
            // DEBUG: HWSFldGen::genCConst
   memset(MSGFLD,'\0',LPOUTAOUT_MAX_MSGFLD+1);
   memcpy(MSGFLD,aFld->data,(aFld->length <
= LPOUTAOUT_MAX_MSGFLD)?aFld->length:LPOUTAOUT_MAX_MSGFLD);
   aFld = aFld->next;

            // DEBUG: HWSFldGen::genCConst
   memset(DATE,'\0',LPOUTAOUT_MAX_DATE+1);
   memcpy(DATE,aFld->data,(aFld->length <
= LPOUTAOUT_MAX_DATE)?aFld->length:LPOUTAOUT_MAX_DATE);
   aFld = aFld->next;

            // DEBUG: HWSLPageGen::genCConst
}

           // DEBUG: HWSLPageGen::genCDest
LPOUTAOut::~LPOUTAOut()
{
            // DEBUG: HWSSegGen::genCDest
            // DEBUG: HWSFldGen::genCDest
```

```
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSSegGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSSegGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSSegGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSFldGen::genCDest
            // DEBUG: HWSLPageGen::genCDest
}

            // DEBUG: HWSLPageGen::genCExec
            // DEBUG: HWSLPageGen::genCExport
            // DEBUG: HWSLPageGen::genCHtml
// Overloaded Method genHTML
char * LPOUTAOut::genHTML()
{
   char     * buf;
   HWSLNode * aFld;

   buf = (char *)malloc(sizeof(LPOUTAOut)*10);
   strcpy(buf,"<table border=5 cellpadding=6>/n");

 aFld = fList->getFirst();
            // DEBUG: HWSSegGen::genCHtml
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"");
   strcat(buf,"LNAME");
   strcat(buf,"\n");
   strcat(buf,"");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"");
   strcat(buf,"FNAME");
   strcat(buf,"\n");
   strcat(buf,"");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
          // DEBUG: HWSSegGen::genCHtml
            // DEBUG: HWSFldGen::genCHtml
   /* skipping the extended attribute data*/
   aFld = aFld->next;
   strcat(buf,"");
   strcat(buf,"EMPNO");
   strcat(buf,"\n");
   strcat(buf,"");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"");
   strcat(buf,"SSN");
   strcat(buf,"\n");
   strcat(buf,"");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
         // DEBUG: HWSSegGen::genCHtml
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"");
   strcat(buf,"RATE");
   strcat(buf,"\n");
   strcat(buf,"");
```

```
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSSegGen::genCHtml
            // DEBUG: HWSFldGen::genCHtml
   strcat(buf,"");
   strcat(buf,"MSGFLD");
   strcat(buf,"\n");
   strcat(buf,"");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSFldGen::genCHtml
  strcat(buf,"");
   strcat(buf,"DATE");
   strcat(buf,"\n");
   strcat(buf,"");
   strncat(buf,(char *)aFld->data,aFld->length);
   strcat(buf,"\n");
   aFld = aFld->next;
            // DEBUG: HWSLPageGen::genCHtml
   strcat(buf,"</table>\n");
   return buf;

}

           // DEBUG: HWSLPageGen::genCList
// Overloaded Method genList
HWSLNode * LPOUTAOut::genList()
{
   HWSLNode * fldList  = NULL;
   HWSLNode * aNode    = NULL;
   HWSLNode * newNode  = NULL;
   HWSLNode * prevNode  = NULL;
            // DEBUG: HWSSegGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSSegGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSSegGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSSegGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSFldGen::genCList
            // DEBUG: HWSLPageGen::genCList
   if (fList) {
     for (aNode = fList->getFirst();
           aNode != NULL;
           aNode = aNode->next) {
         newNode = new HWSLNode(aNode->name,aNode->length,aNode->data);
         if (prevNode) {
            prevNode->next = newNode;
         } else {
            fldList = newNode;
         } /* end if */
         prevNode = newNode;
      } /* end for */
   } /* end if */
   return fldList;

}

           // DEBUG: HWSLPageGen::genCDispList
// Overloaded Method genList
HWSLNode * LPOUTAOut::genDisplayList()
{
```

```
HWSLNode * fldList  = NULL;
HWSLNode * aNode    = NULL;
HWSLNode * newNode  = NULL;
HWSLNode * prevNode = NULL;
int        addIt    = 0;

if (fList) {
   for (aNode = fList->getFirst();
        aNode != NULL;
        aNode = aNode->next) {

      addIt = 0;
         // DEBUG: HWSSegGen::genCDispList
         // DEBUG: HWSFldGen::genCDispList
      if( !strcmp( aNode->name, "LASTN" ) ) {
         newNode = new HWSLNode( "LASTN",
                                 aNode->length,
                                 aNode->data );
         addIt = 1;
      }
         // DEBUG: HWSFldGen::genCDispList
      if( !strcmp( aNode->name, "HWSMF007" ) ) {
         newNode = new HWSLNode( "HWSMF007",
                                 aNode->length,
                                 aNode->data );
         addIt = 1;
      }
         // DEBUG: HWSSegGen::genCDispList
         // DEBUG: HWSFldGen::genCDispList
      if( !strcmp( aNode->name, "HWSMF008_ATTR" ) ) {

        /*-------------------------------------------------------------------*/
         /* Process extended attribute data.                                  */
         /*-------------------------------------------------------------------*/
         char colorRed[] = "\xC2\xF2";   /* Extended attribute is type is   */
                                         /* color and color is red.         */
         char red[]      = "RED";
         char dflt[]     = "DEFAULT";

         if( (memcmp( aNode->data, colorRed, 2) == 0) )
         {
            newNode = new HWSLNode( "HWSMF008_ATTR",
                                    sizeof(red)-1,
                                    red );
         }
         else
         {
            newNode = new HWSLNode( "HWSMF008_ATTR",
                                    sizeof(dflt)-1,
                                    dflt );
         }
       addIt = 1;
      }
      if( !strcmp( aNode->name, "HWSMF008" ) ) {
         newNode = new HWSLNode( "HWSMF008",
                                 aNode->length,
                                 aNode->data );
         addIt = 1;
      }
         // DEBUG: HWSFldGen::genCDispList
      if( !strcmp( aNode->name, "HWSMF009" ) ) {
         newNode = new HWSLNode( "HWSMF009",
                                 aNode->length,
                                 aNode->data );
         addIt = 1;
      }
         // DEBUG: HWSSegGen::genCDispList
```

```
            // DEBUG: HWSFldGen::genCDispList
          if( !strcmp( aNode->name, "HWSMF010" ) ) {
            newNode = new HWSLNode( "HWSMF010",
                                    aNode->length,
                                    aNode->data );
           addIt = 1;
          }
            // DEBUG: HWSSegGen::genCDispList
            // DEBUG: HWSFldGen::genCDispList
          if( !strcmp( aNode->name, "HWSMF011" ) ) {
            newNode = new HWSLNode( "HWSMF011",
                                    aNode->length,
                                    aNode->data );
            addIt = 1;
          }
            // DEBUG: HWSFldGen::genCDispList
          if( !strcmp( aNode->name, "HWSMF012" ) ) {
            newNode = new HWSLNode( "HWSMF012",
                                    aNode->length,
                                    aNode->data );
            addIt = 1;
          }
            // DEBUG: HWSLPageGen::genCDispList
          if ( addIt ) {
            if( prevNode ) {
               prevNode->next = newNode;
            } else {
               fldList = newNode;
            } /* end if */
           prevNode = newNode;
          } /* end if */
        } /* end for */
      } /* end if */
      return fldList;

}

            // DEBUG: HWSLPageGen::genCSetOper
```

Sample IMS transaction table output

The dtw_execute function of the IMS Web language environment returns, in parm_results, a table representing the output of the IMS transaction. Each row of the table corresponds to a logical page of the output message. The first column of each row contains the MOD name and LPage name, separated by an ampersand. The remaining columns contain the values of the output fields of the lpage. The following example shows an output message containing two logical pages, each with a different number of fields. The first row of the example table represents the column headings of the table returned in parm_results.

LPAGEName MFLD01Value MFLD02Value MFLD03Value ... ...
MOD1&LPGA Smith John 099740 ... ...
MOD1&LPGB Chung 99-888-7777 null ... ...

The number of columns in the table is determined by the input parameter parm_MAXCOLS; the number of rows in the table is determined dynamically by the number of lpages returned by the IMS transaction.