Microsoft Windows®

Microsoft Windows 95™

Microsoft Windows NT™

COBOL Cartridge
User's Guide

FUJITSU

First Edition: April 1998

Preface

The COBOL Cartridge is a set of subroutines that allow COBOL programs to interact with Oracle's Web Application Server. The COBOL Cartridge allows programs running on Oracle's Web Application Server to receive requests from Web clients and send back responses to clients in HTML pages.

Audience

This manual is intended for programmers using the COBOL Cartridge to develop Oracle Web Application Server applications in COBOL. We recommend that you also refer to the Oracle documentation on the Web Application Server.

How this Manual is Organized

This manual contains the following information:

Chapter 1	Describes how to install and configure the COBOL Cartridge.
Chapter 2	Details the COBOL Cartridge API Reference.
Chapter 3	Steps through the creation of a simple COBOL Cartridge application.
Chapter 4	Describes how to use the Debugger, set breakpoints, and perform other debugging functions.
Appendix A	Lists the contents of the COBW3.CBL file.
Appendix B	Lists the contents of the CALCULATE_BIG.COB file.
Appendix C	Describes how to use the PL/SQL Web Toolkit.

Conventions Used in This Manual

Example of convention	Description
setup	Characters you enter appear in bold.
Program-name	Underlined text indicates a placeholder for information you supply.
ENTER	Small capital letters are used for the name of keys and key sequences such as ENTER and CTRL+R. A plus sign (+) indicates a combination of keys.
…	Ellipses indicate the item immediately preceding can be specified repeatedly.
Edit, Literal	Names of pulldown menus and options appear with the initial letter capitalized.
[def]	Indicates that the enclosed item may be omitted.
{ABC | DEF}	Indicates that one of the enclosed items delimited by | is to be selected.
CHECK WITH PASCAL LINKAGE ALL PARAGRAPH-ID COBOL ALL	Commands, statements, clauses, and options you enter or select appear in uppercase. Program section names, and some proper names also appear in uppercase. Underlined text indicates the default.

`PROCEDURE DIVISION`
`:`
`ADD 1 TO POW-FONTSIZE OF LABEL1.`
`IF POW-FONTSIZE OF LABEL1 > 70 THEN`
`MOVE 1 TOW POW-FONTSIZE OF LABEL1.`
`END-IF.`
	This font is used for examples of program code.

The *form* acts as an application creation window.	Italics are occasionally used for emphasis.
"PowerCOBOL Reference" See "Compile Options" in Chapter 5.	References to other publications or sections within publications are in quotation marks.

Related Manuals

Getting Started with Fujitsu COBOL
Fujitsu COBOL User's Guide for Windows
Fujitsu COBOL Debugging Guide
Fujitsu COBOL Language Reference -- Volume 1
Fujitsu COBOL Language Reference -- Volume 2

Trademarks

Contents

Chapter 1. Installing and Configuring the COBOL Cartridge

This chapter explains how to install and configure the COBOL Cartridge on the Oracle Web Application Server that is running under the Windows NT operating system. Information that is specific to UNIX or Windows NT is noted in the chapter. The steps for installing the COBOL Cartridge involve setting up the Oracle Web Application listener to recognize the virtual directory paths to the COBOL Cartridge binary files, and other directories that you may choose to deploy associated Web content. A prerequisite to installing the COBOL Cartridge is to have the Oracle Web Application Server installed and running properly.

Configuring the Listener

This section explains how to configure the Oracle Web Application listener. In this example, you will be using the svzip.cfg file, located in directory `C:\orant\ows\admin\ows\zip\httpd_spice\zip\config`. Your path and configuration file name should be similar, depending on how the Web Application Server was installed and the name of your listener. To configure the listener:

1. Log in by entering the URL of your server, and connecting to port 8888. (Port 8888 is the administrative port for Oracle Web Application Server 3.0.) You be presented with a login screen where you must enter the administrator password. After you have correctly entered the login information, you should see a display similar to the following figure.

Figure 1.1. Administrative page for the Oracle Web Application Server

2. Navigate to the web listener configuration page by selecting the following hypertext links: *Web Application Server Manager - Oracle Web Listener*. When you scroll down to the bottom of the page, you should see a list of available listeners similar to those shown in the figure below that displays the bottom half of the Oracle Web Listener Administration page. This example will use the zip listener. The zip listener listens on port 7777.

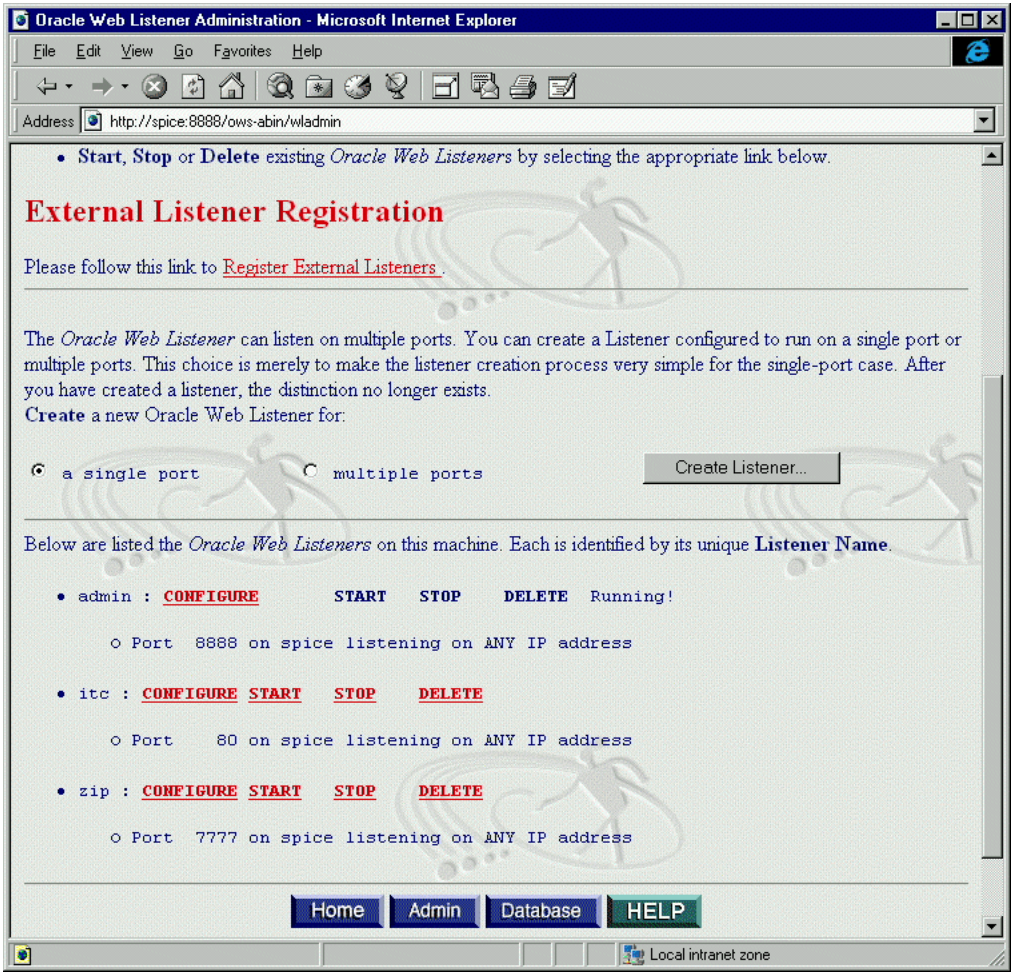

Figure 1.2. The bottom half of the Oracle Web Listener Administration page, showing available listeners

3. For the listener you intend to use with your COBOL cartridge, select *Configure* to access the listener configuration parameters.

4. You will now set up a virtual path for your COBOL Web files. Your choice for virtual path names should be driven by your

desire to organize your Web project so that it is easily understood and maintained by your organization.

On the vertical frame to the left, select Directory to jump to the Directory Mappings section. Scroll down to the Form fields so you can enter the physical path and the virtual path to your COBOL files.

This example will set the virtual path name to /zip-cobol/in order to be consistent with the arbitrary naming convention already chosen. The following figure shows the form filled out for your COBOL settings, as well as settings for other activities. Note that you could dump all your files into one location, but this would defeat the purpose of the directory mappings. The goal is to make the management of your files orderly and easy to maintain.

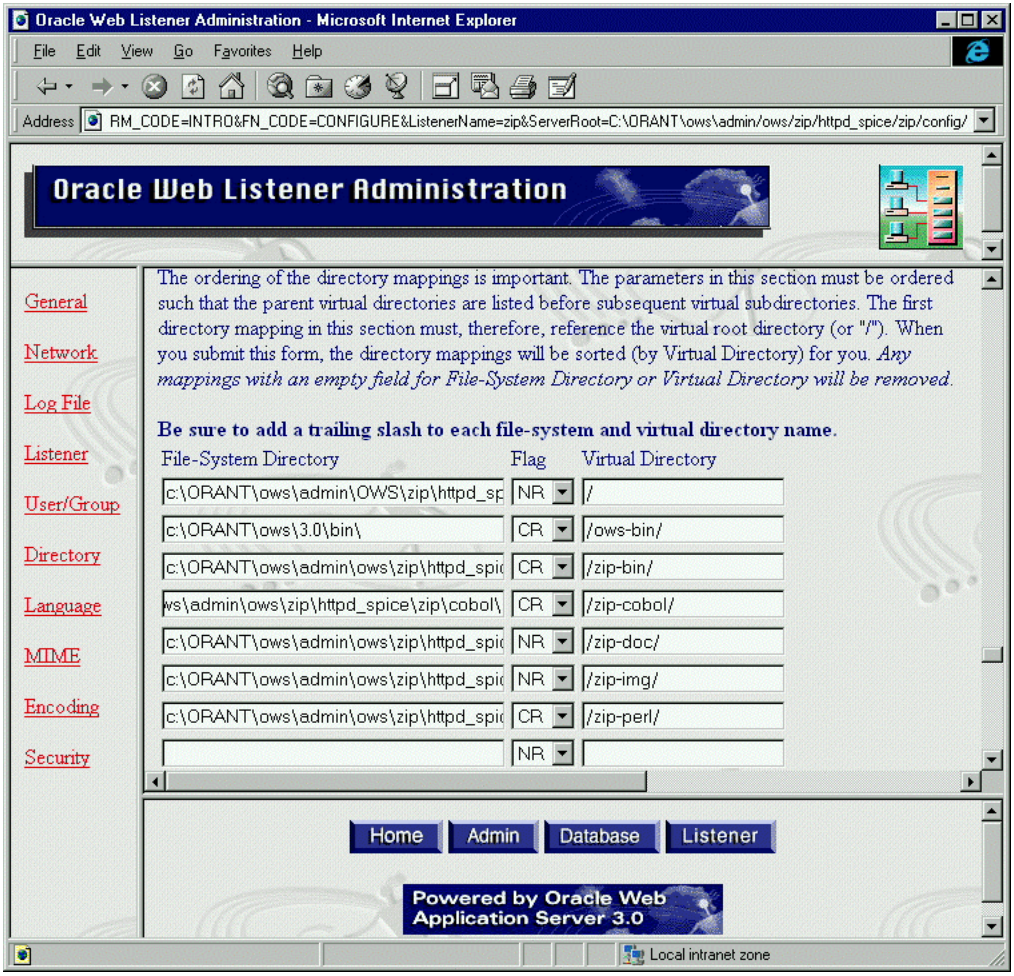

Figure 1.3. The Virtual Directory section of the Listener Configuration page

The directory information has been scrolled to the left in the above figure so you can see the detail on the /zip-cobol/ virtual directory.

5. Click on the Modify Listener button to conclude the listener configuration. This will write the changes to the configuration file. If properly done, the return page will show you the

green letters of “Success!”. If there are errors, this is usually because of an input error, such as an invalid directory.

The changes will not become effective until you stop and start the listener. This will be the case any time you change the listener configuration file.

It is always a good practice to stop and start the listener immediately after making any changes to the configuration file. If you wait to restart the listener over the weekend, you might not remember the last thing you did to the configuration file (a valuable clue to what went wrong). You will eventually discover this by looking into the listener error log file.

The listener error file for the zip listener described in this chapter is located at `C:\ORANT\ows\admin\ows\zip\httpd_spice\zip\log\svzip.err`. You can find where yours is by entering the Listener Configuration Web page again, clicking on the *Log File* hypertext link, and looking at the entry for *Log Admin File*.

Installing the COBOL Cartridge

Installing the COBOL Cartridge is an easy task. However, care must be taken putting the Cartridge DLL file (.so file for UNIX) in the correct virtual directory, and correctly typing in all the information to register it with the Web Request Broker.

Initial Cartridge Configuration

If you are not logged in as the administrator, do so now. (See "Configuring Your Listener" for a detailed explanation of this step.) Complete the following steps for initial cartridge configuration:

1. From the Cartridge Administration page click on the following hypertext links: *Web Application Server Manager* - *Oracle Web Application Server* - *Cartridge Administration*. The Cartridge Administration page is shown in the following figure.

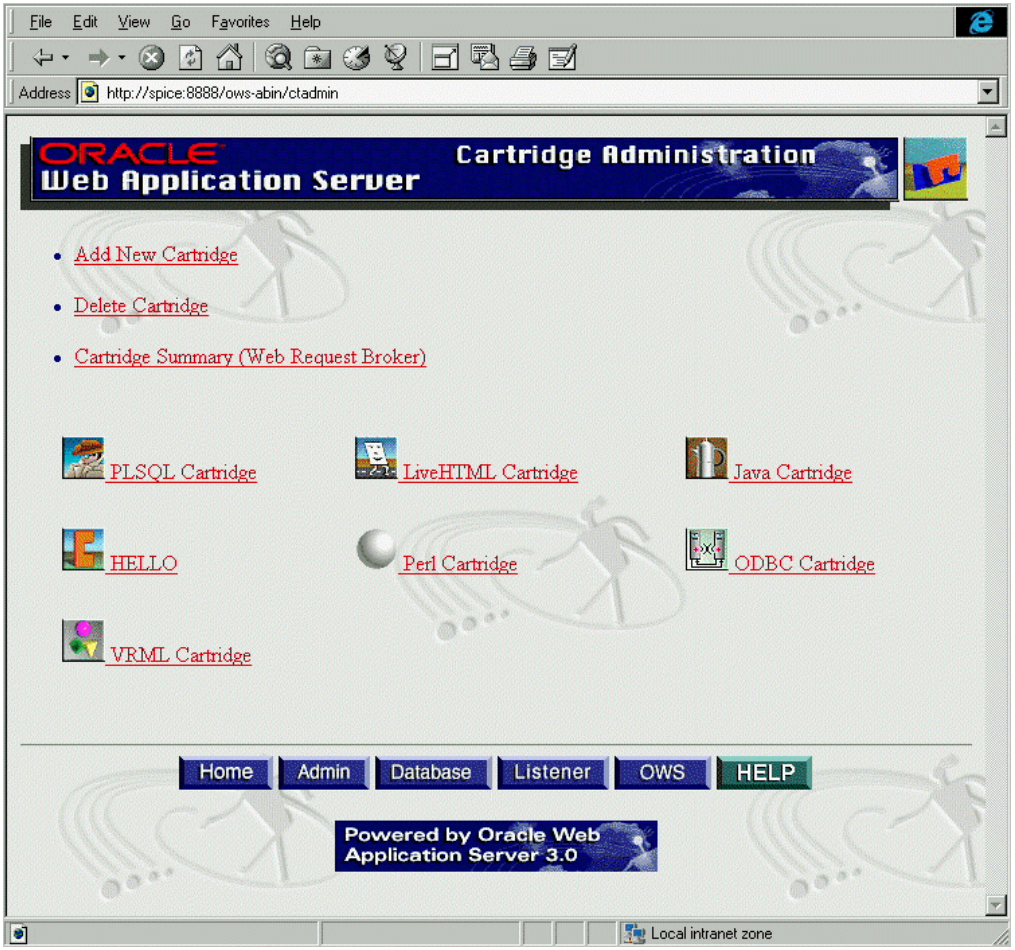

Figure 1.4. The Cartridge Administration page

2. Click on Add New Cartridge to bring up the New Cartridge Configuration page. Click on the hypertext link *Add New Cartridge with Manual Configuration*. The New Cartridge Configuration page is displayed.

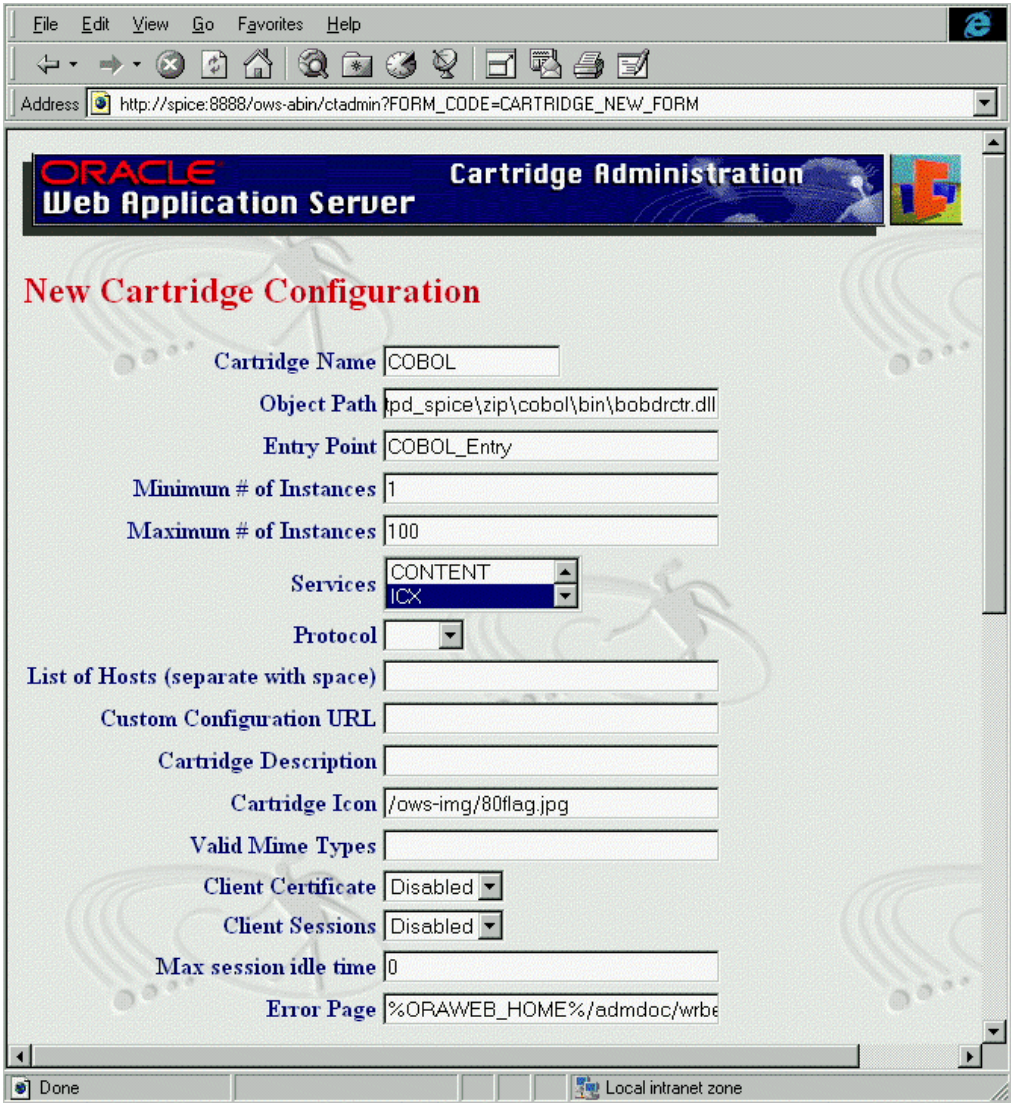

Figure 1.5. The top half of the New Cartridge Configuration page

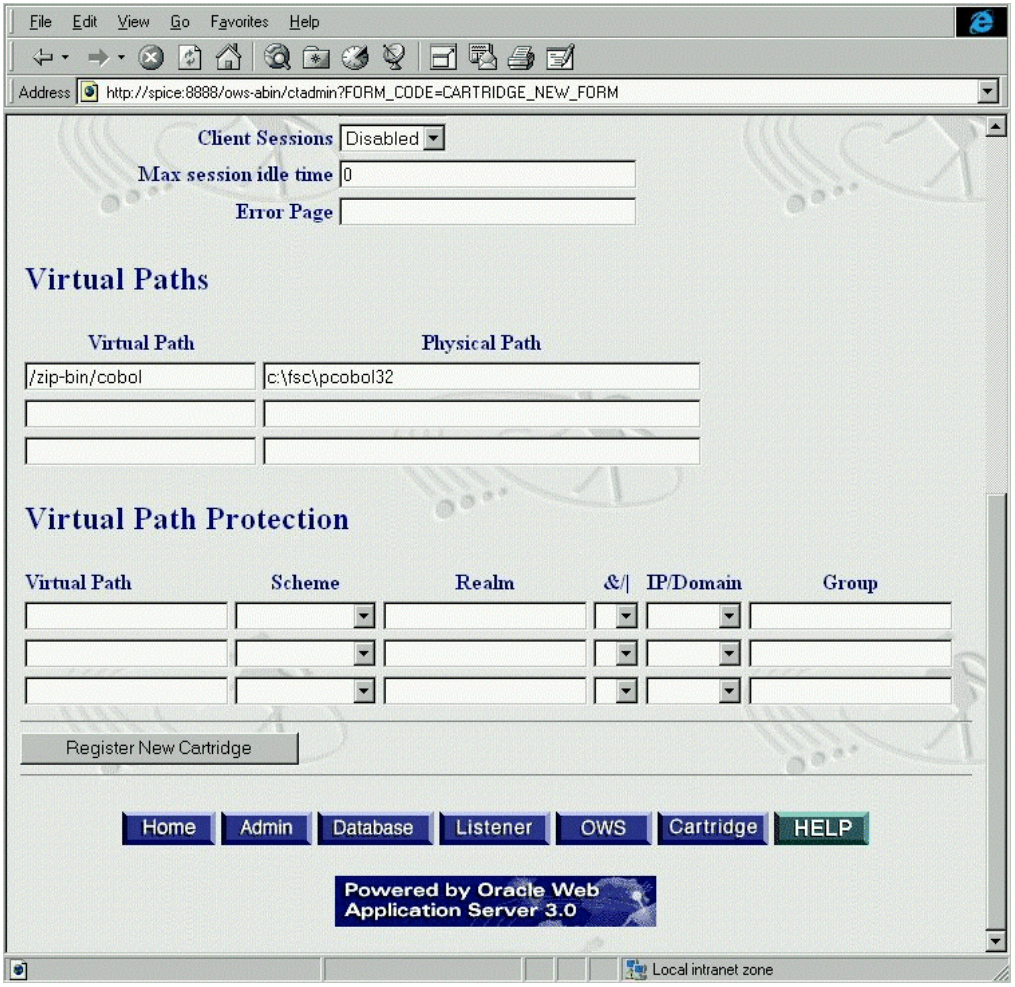

Figure 1.6. The bottom half of the New Cartridge Configuration page

You need to fill in the blanks to describe the cartridge. **Note**: The above figures are already filled out. Exercise care here, as path names and file names must be absolutely correct. If you make an error, the cartridge will not run. The following table lists the parameters, inputs, and explanations for the input.

The inputs reflect the configuration used in this example, yours will likely be similar.

Table 1.1. New Cartridge Configuration Parameters for Windows NT

Parameter Name	Input Name	Meaning
Cartridge Name	COBOL	The name of the cartridge. This will be the name that will appear on the Cartridge Administration page. (see Figure 1.4)
Object Path	C:\orant\ows\admin\ows\zip\httpd_spice\zip\cobol\cobdrctr.dll	This is the location of the COBOL Cartridge dynamic link library. This must be mapped with the virtual directory of the listener as shown in Figure 1.3.
Entry Point	COBOL_Entry	The entry point to the cartridge. Enter exactly as is.
Min	1	The minimum number of resident processes desired. The server will add more as necessary, but this is the minimum. You can change it later in order to tune the cartridge.
Max	100	This value depends on how much memory your server has. If you can't support 100 memory resident cartridges and applications at one time, reduce the number to an acceptable value. This may require some experimentation.
Services	ICX	The COBOL Cartridge uses Inter-Cartridge Exchange protocol.
Cartridge Icon	/ows-img/80flag.jpg	You may use any graphic for the icon. Put it into the /ows-img/ directory because this is the directory from where the Web Application Server administration listener reads images.
Virtual Path	c:\fsc\pcobol32	This is the physical path to the COBOL run-time binary files. This is the path for a default installation of the Fujitsu COBOL 4.0 products.

Once the new entries are made, click on the Register New Cartridge button to finalize entries.

3. The COBOL entry point needs to be registered with the Web Request Broker (WRB). These values should automatically be entered when you registered the new cartridge in the previous step. To check this, navigate to the Web Request Broker Administration page, and select the Applications and Objects section. From the Cartridge Administration page (Figure 1.4), you may navigate to this location by selecting the *Cartridge Summary (Web Request Broker)* link. Confirm that there is an entry for the COBOL cartridge that reflects your directory path to the cartridge. This is shown under the heading "Object Path". The entry point must be named "COBOL_Entry". Set the minimum and maximum number of COBOL cartridges as you did in step 2. You may want to adjust these later as you tune your system. The screen that configures this example is shown below.

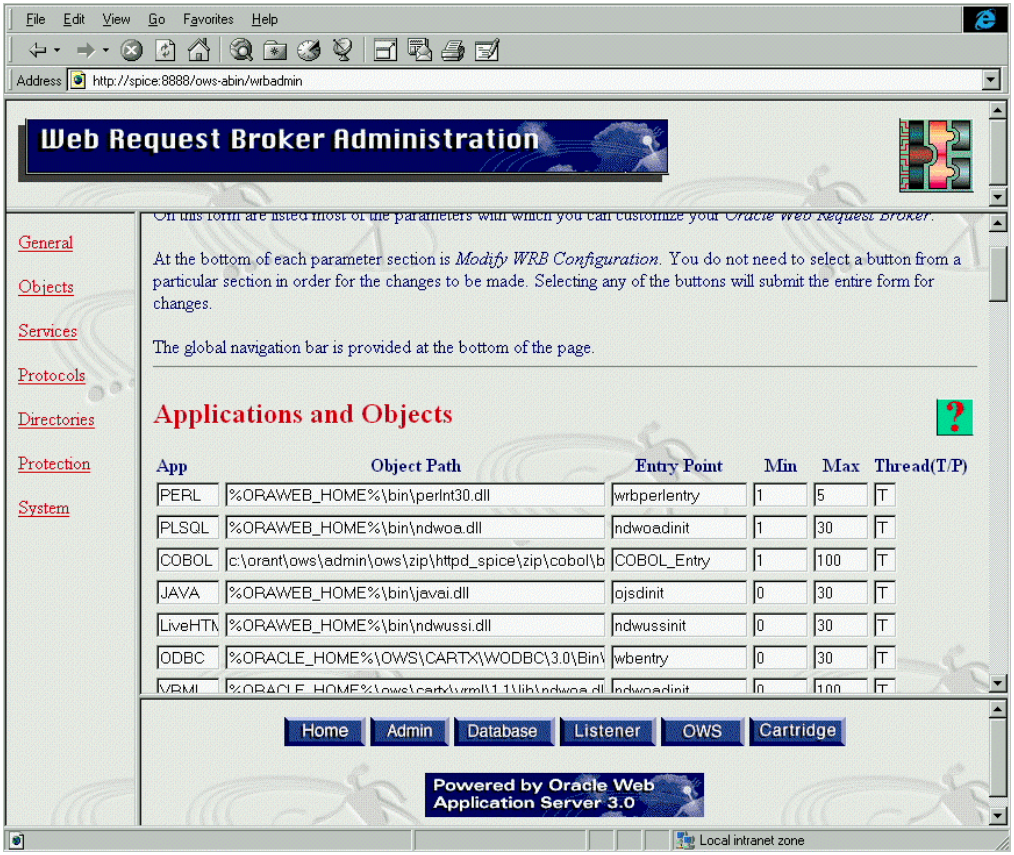

Figure 1.7. Registering the COBOL Cartridge with the Web Request Broker

Setting the COBOL Cartridge Specific Parameters

Navigate back to the Cartridge Administration page. (This page is shown in Figure 1.4.) Select the *COBOL Cartridge* link, and then the *COBOL Cartridge specific parameters* hypertext link. You will see the Cartridge Configuration form as show in the following figure. **Note**: Your form will be blank.

Figure 1.8. The COBOL Cartridge configuration form

You will need to enter parameters and values to complete this form. The following table can be used to fill in the blanks as it shows the proper values for both UNIX and Windows NT operating systems.

Note that the COBOL_PATH parameter has two paths in this example:

```
C:\ORANT\OWS\admin\OWS\zip\HTTPD_SPICE\zip\cobol\bin,
c:\fsc\pcobol32
```

This is because we plan to have executable binary files in both of these locations to support our COBOL Cartridge.

The example shown in Figure 1.4 uses a minimal configuration thus the security parameters have been omitted

Table 1.2. COBOL Cartridge Specific Parameters for UNIX and Windows NT

Parameter Name	UNIX Value	Windows NT Value
System_InitPGM	Libcobol.so/JMPCINT2	F3BIPRCT.DLL/JMPCINT2
System_termPGM JMPCINT3	Libcobol.so/JMPCINT3	F3BIPRCT.DLL/JMPCINT3
User_InitPGM DBCONECT	libcobdrctr.so/DBCONECT	COBDRCTR.dll/DBCONECT
User_ExecPGM DBACCESS	libcobdrctr.so/DBACCESS	COBDRCTR.dll/DBACCESS
User_TermPGM DBDSCNCT	libcobdrctr.so/DBDSCNCT	COBDRCTR.dll/ DBDSCNCT
COBOL_PATH	Path to COBOL program you have “Web-enabled”	
AuthorizeSPEC	Authorization level to be applied to User_AuthorizePGM. Values and descriptions are shown in Table 1.3. Using this parameter requires specification of a realm.	
AuthorizeREALM	Realm name must be defined previously in the listener security setup. See Oracle documentation on Web Application security for more information.	
User_AuthorizePGM	The name of the user program for processing authorizations.	

The following table shows the possible values of AuthorizeSPEC. This information is necessary to choose the correct value for AuthorizeSPEC in the COBOL Cartridge specific parameters.

Table 1.3. AuthorizeSPEC Values Descriptions

Value	Description
EXIST	This parameter defaults the user authorization to the current setting of the Web Listener
BASIC_NEW	This parameter flags basic authorization that will be made by the user program.
DIGEST_NEW	This parameter flags digest authorization that will be made by the user program. Oracle recommends not using this because all browsers do not support it, and what little security it has is weak.

Your final task in configuring the COBOL Cartridge is to shut down and restart the cartridge to re-initialize the Web Request Broker and make your changes effective. When bringing down the WRB, you must first shut down any listeners that are running and then the WRB itself. For example, if you have a listener named “zip”, and the admin listener running, you would enter the following commands to shut down the Web Application server and bring it back up:

```
owsctl stop zip
owsctl stop admin
owsctl stop wrb
owsctl start wrb
owsctl start admin
owsctl start zip
```

Turning off the Run-time Environment Setup

You will most likely want to copy the file COBOL85.CBR to the C:\ORANT\OWS\3.0\BIN\ directory to prevent the Run-time Environment Setup window from popping up when users run your Web application. The Run-time Environment Setup window could be annoying and possibly bewildering to your users.

Chapter 2. COBOL Cartridge API Reference

This chapter describes the COBOL Cartridge Application Program Interface subroutines that are provided with your COBOL Cartridge so that you can web-enable your COBOL programs. The subroutines are discussed in alphabetical order rather than by functional groupings since many users find it easier to study the example programs in Chapter 3, and look up the APIs in this chapter. The functional grouping of these subroutines can be found in the table below.

COBOL Cartridge WRB API Subroutines

The COBOL Cartridge comprises of a set of standard COBOL APIs that allows the integration of the Oracle Web Application Server and COBOL programs. This integration enables the deployment of COBOL programs onto the Web for Intranet or Internet use. You may also use the APIs to web-enable legacy COBOL code. The table below lists the subroutines according to function.

Table 2.1. COBOL API Subroutines by Function

Function	Subroutine Name	Description
Initialization	COBW3_INIT	Free up resources reserved at initialization.
Search and Fetch	COBW3_NAME	Retrieve CGI environmental name.
	COBW3_VALUE	Retrieve CGI environmental value.
Output Process Results	COBW3_PUT_HTML	Send HTML to the browser.
	COBW3_PUT_TEXT	Send plain text to the browser.
	COBW3_CNV_INIT	Initialize values for HTML file edit
	COBW3_CNV_SET	Set a value in an HTML file.
	COBW3_CNV_DEL	Delete a previously set value from an HTML file
Execute System Command	COBW3_SYSTEM	Run a system command.
Acquire Authorization	COBW3_AUTHORIZE	Process program authorization using user id, password and IP address
Release Environment Resources	COBW3_FREE	Free up resources reserved by the COBW3_INIT subroutine.

In addition to the following topics on the COBW3 subroutines, you may profit from examining the COBW3.CBL file that defines all of the constants used in the COBW3 environment block. This file is well documented and is useful as a quick-reference to understand the variables and constants that are used by the COBW3 subroutines. You will find the COBW3.CBL file in Appendix A, “The COBW3.CBL File. ”

The use of the constants in the COBW3 include file will make your COBOL program code more readable and easier to maintain. For example, the COBW3-CNV-MODE-ADD constant may assume a different value in the future. If you assigned a hard-coded value of "2" (the current value of COBW3-CNV-MODE-ADD) you would need to update your code to the new value. While it's unlikely this value will ever assume a different constant, you should be aware of the benefits of using the COBW3 constants, and the risks of hard-coded values.

Using Subroutine COBW3_CNV_DEL

This procedure will delete a value that was previously set using the COBW3_CNV_SET subroutine from an HTML file that is specified in the COBW3-HTML-FILENAME location of the COBW3 API routine work area.

Setting Parameters

COBW3-CNV-NAME: The conversion name that is targeted in the HTML document.

COBW3-CNV-NAME-LENGTH: The character string length in bytes of conversion name targeted in the HTML document. The range of this value is 0 to 30, with a default value of zero.

Return Values

None.

Notes

The purpose of the conversion subroutines is to simplify the process of making dynamic HTML pages. Rather than build an entirely new page each time a subroutine is called, the "CNV"

subroutines update an existing HTML file. Special tags that are unique to the COBOL Cartridge are embedded into the HTML file to specify where, and what parameters are to be acted upon. The format of these tags is:

```
//COBOL// SOME-VALUE //COBOL//
```

In this case, “SOME-VALUE” is the item that is changed by the CNV routines.

Example

```
ENVIRONMENT DEVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
     SYMBOLIC CONSTANT
     SC-HTML-FILE-NAME IS
"c:\orant\ows\admin\ows\zip\spice\zip\cobol\calculator.htm"
     SC-CNV-HEXNUMBER   IS "CNV-HEXNUMBER"
.
.
.
MOVE SC-CNV-HEXNUMBER TO COBW3-CNV_NAME
CALL "COB3_CNV_DEL" USING COBW3
.
.
.
MOVE SC-HTML-FILE-NAME TO COBW3-HTML-FILENAME
CALL "COBW3_PUT_HTML" USING COBW3.
```

This example uses a file named “SPS.HTM” and updates a value called SC-CNV-HEXNUMBER. This effectively clears the HTML input text field. Below is a partial listing of CALCULATOR.HTM that shows the CNV notation used to update an HTML file.

```
.
.
.
<P>
<STRONG><FONT FACE="Courier New">Hex Number : </FONT></STRONG>
<INPUT TYPE="text" NAME="HEXNUMBER"
SIZE="16" //COBOL//CNV-HEXNUMBER//COBOL//>
</P>
.
```

```
:
```

Using Subroutine COBW3_CNV_INIT

This procedure initializes resources for updating an HTML file that is specified in the COBW3-HTML-FILENAME location of the COBW3 API routine work area. Making this call will clear out any values previously set by COBW3_CNV_SET calls.

Setting Parameters

None.

Return Values

None.

Example

```
CALL "COBW3_CNV_INIT" USING COBW3
```

Using Subroutine COBW3_CNV_SET

This procedure sets a value in an HTML file that is specified in the COBW3-HTML-FILENAME location of the COBW3 API routine work area.

Setting Parameters

COBW3-CNV-NAME: The name of the string that is to be converted.

COBW3-CNV-NAME-LENGTH: The byte length of COBW3-CNV-NAME.

COBW3-CNV-VALUE: The value that the string COBW3-CNV-NAME is to be converted to as prescribed by the COBW3-CNV-MODE.

COBW3-CNV-MODE: Mode for the conversion to take place for the targeted COBW3-CNV-NAME to be acted and processed. The default value for COBW3-CNV-MODE is COBW3-CNV-MODE-ADDREP. The following table lists the available modes.

Table 2.2. Modes for COBW3-CNV-MODE

Mode	Set Value	Description
COBW3-CNV-MODE-ADDREP	LOW-VALUE	COBW3-CNV-NAME data is to be inserted if it doesn't exist, or replaced if it does exist.
COBW3-CNV-MODE-REPLACE	1	COBW3-CNV-NAME data is to be replaced.
COBW3-CNV-MODE-ADD	2	COBW3-CNV-NAME data is to be inserted.

Return Values

None.

Notes

See notes for COBW3_CNV_DEL.

Example

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
    SYMBOLIC CONSTANT
    SC-INVALID-MONTH       IS "Invalid month. Please try again."
    SC-CNV-MESSAGE         IS "CNV-MESSAGE"
```

```
    SC-HTML-FILE-NAME      IS
"c:\oracle\ows\admin\ows\zip\spice\zip\cobol\birthday.htm".
DATA DIVISION.
WORKING-STORAGE SECTION.
01  HTML-MESSAGE           PIC X(200) VALUE SPACES..
.
.
.
MOVE SC-INVALID-MONTH TO HTML-MESSAGE
MOVE HTML-MESSAGE     TO COBW3-CNV-VALUE
MOVE SC-CNV-MESSAGE   TO COBW3-CNV-NAME
CALL "COBW3_CNV_SET"  USING COBW3

MOVE SC-HTML-FILE-NAME TO COBW3-HTML-FILENAME
CALL "COBW3_PUT_HTML"  USING COBW3.
.
.
.
```

Using Subroutine COBW3_FREE

This procedure frees the resources that were reserved during the initialization of the COBOL application and the COBOL Cartridge. This should be the last call in all web-enabled COBOL programs, just before the EXIT PROGRAM statement.

Setting Parameters

None.

Return Values

None.

Notes

Failure to use this call, followed by an unrelated session of COBW3 subroutine calls will give unpredictable results.

Example

```
.
.
.
*-----------------------------------------
TERMINATE-ROUTINE.
   CALL "COBW3_FREE" USING COBW3
```

Using Subroutine COBW3_GET_AUTHORIZE

This procedure is necessary to process authorization by the COBOL Program. The user ID, password and IP address are parameters that are captured by this call.

Setting Parameters

None.

Return Values

The following table lists values that are returned within the COBW3 environment block when the COBW3_GET_AUTHORIZE subroutine is called.

Table 2.3. Values Returned Within the COBW3 Environment Block

Value	Description
COBW3-USERID	A string containing the user ID of the client.
COBW3-USERID-LENGTH	The length of the user ID string in bytes.
COBW3-PASSWORD	A string containing the password of the client.
COBW3-PASSWORD-LENGTH	The length of the password string in bytes.
COBW3-IP-ADDRESS	A string containing the IP address of the client.
COBW3-IP-ADDRESS-LENGTH	The length of the IP address string in bytes.

Notes

You will need to configure the COBOL Cartridge Specific Parameters to enable the use of this procedure. See “Setting the COBOL Cartridge Specific Parameters” in Chapter 1 for a discussion on configuring the COBOL Cartridge for authorization. You also may want to read discussions on configuring the Web listener, and the WRB for security authorization schemes.

Example

```
CALL "COBW3_AUTHORIZE"                    USING COBW3
```

Using Subroutine COBW3_INIT

This procedure sets up the environment variables that serve as a communications area between the Cartridge and the program. This call is mandatory for all web-enabled COBOL programs.

Setting Parameters

The following parameters may be set before calling this procedure:

COBW3-CONTEXT: A pointer to synchronize data areas in the Cartridge and the COBOL application.

COBW3-DMODE: Specifies if debugging is to be done. This parameter has two settings as shown in the following table.

Table 2.4. Modes for COBW3-DMODE

Mode	Set Value	Description
COBW3-DMODE-NODBG	LOW-VALUE	No debugging will be done. This is the default value.
COBW3-DMODE-DBG	1	Debugging will be done using the COBW3_PUT_TEXT call.

COBW3-CONTENT-TYPE: Used only when "debug mode" is set. This parameter has two settings as shown in the following table.

Table 2.5. Modes for COBW3-CONTENT-TYPE

Mode	Set Value	Description
COBW3-CONTENT-TYPE-HTML	"text/html"	This sets up HTTP output to be formatted HTML. This is the default value.
COBW3-CONTENT-TYPE-TEXT	"text/plain".	This sets up HTTP output to be plain text with native formatting.

Return Values

None.

Notes

When debugging by calling COBW3_PUT_TEXT, the debug mode may be set arbitrarily as the need arises, in locations of your COBOL program that are in question. Data that is not supported by the COBW3_PUT_TEXT procedure (for example, non-text data) cannot be displayed.

Error messages that are detected in the COBOL Cartridge APIs will be displayed on the browser when the COBW3_PUT_TEXT procedure is called.

Example

This example sets the debug mode on.

```
MOVE WRBCTX                          TO COBW3-CONTEXT.
SET COBW3-DMODE-DBG TO TRUE.
CALL "COBW3_INIT"                    USING COBW3.
```

Using Subroutine COBW3_NAME

This procedure is used to retrieve CGI “name/value” pairs from the environment data area of the COBW3.

Setting Parameters

The following parameters are used when retrieving name/value pairs:

COBW3-SEARCH-DATA: Set to the CGI “name” of the value that being passed.

COBW3-SEARCH-LENGTH: The length of the “name” character string. COBW3-SEARCH-LENGTH ranges from 0 - 1024. The default value is zero.

COBW3-NUMBER: If multiple names exist, this parameter will be set to the number of names. The first value is COBW3-NUMBER-INIT, and is set to a value of numeric 1. COBW3-NUMBER ranges from 1-9999. The default value is 1.

Return Values

The following values are set when COB_NAME is called:

COBW3-SEARCH-FLAG: The possible settings are shown in the following table.

Table 2.6. Values for COBW3-SEARCH-FLAG

Mode	Set Value	Description
COBW3-SEARCH-FLAG-NON	0	The “name” does not exist.
COBW3-SEARCH-FLAG-EXIST	“1	The “name” does exist.

COBW3-GET-DATA: Indicates that the “value” that corresponds to “name” is set.

COBW3-GET-LENGTH: The length of the character string in bytes of the CGI value that corresponds to the CGI name.

Example

```
.
.
.
CALL "COBW3_NAME"                  USING COBW3.
IF COBW3-SEARCH-FLAG = SC-COBW3-SEARCH-NOT-FOUND OR
    COBW3-GET-DATA = SPACE
    GO TO OUTPUT-HTML
END-IF.

MOVE COBW3-GET-DATA                TO    DECNUMBER-SAVE.
.
.
.
```

Using Subroutine COBW3_PUT_HEAD

This procedure sends a line of text as an HTTP command.

Setting Parameters

The following are parameters that may be set before calling this subroutine:

COBW3-PUT-HEAD: The value that contains the string to be output to the browser.

COBW3-PUT-HEAD-LENGTH: The byte length of the COBW3-PUT-HEAD string. The default is zero, and the range is from 1 to 512.

COBW3-CONTENT-TYPE: Sets the HTTP content header to plain text or HTML. Possible values are shown in the following table.

Table 2.7. Values for COBW3-CONTENT-TYPE

Mode	Set Value	Description
COBW3-CONTENT-TYPE-HTML	text/html	This is the default value. Used for HTML text
COBW3-CONTENT-TYPE-TEXT COBW3-CONTENT-TYPE-NON	text/plain HIGH-VALUE	Used for plain text. Content-type header will not be output

COBW3-STATUS-CODE: The return status that results from the HTTP Content-type command. Possible values are shown in the following table.

Table 2.8. Values for COBW3-STATUS-CODE

Mode	Set Value	Description
COBW3-STATUS-CODE-200	200	This is the default value. It indicates normal completion.
COBW3-STATUS-CODE-NON	HIGH-VALUE	This parameter specifies that the status-code header is not to be output, otherwise this value receives the numeric value of the status code as a three-digit integer.

Return Values

None.

Example:

```
EXECUTE-PLSQL SECTION.
 MOVE DOB-DAY  OF DATE-OF-BIRTH               TO DOB-DAY   OF DATE-OF-BIRTH-
FORMAT
 MOVE MONTH-NAME (DOB-MONTH OF DATE-OF-BIRTH) TO DOB-MONTH OF DATE-OF-BIRTH-
FORMAT
 MOVE DOB-YEAR OF DATE-OF-BIRTH               TO DOB-YEAR  OF DATE-OF-BIRTH-
FORMAT
 MOVE SPACES TO COBW3-PUT-HEAD
```

```
STRING PLSQL-ADDRESS-STRING DELIMITED BY SIZE
          FIRST-NAME        DELIMITED BY SPACE
          LAST-NAME-STRING  DELIMITED BY SIZE
          LAST-NAME         DELIMITED BY SPACE
          DATE-OF-BIRTH-STRING DELIMITED BY SIZE
          DATE-OF-BIRTH-FORMAT DELIMITED BY SPACE
          INTO COBW3-PUT-HEAD
END-STRING
SET  COBW3-CONTENT-TYPE-NON TO TRUE
SET  COBW3-STATUS-CODE-NON  TO TRUE
CALL "COBW3_PUT_HEAD"       USING COBW3.
```

Using Subroutine COBW3_PUT_TEXT

This procedure sends a line of text back to the browser.

Setting Parameters

The following are parameters that may be set before calling this subroutine:

COBW3-PUT-STRING: The value that contains the string to be output to the browser.

COBW3-PUT-STRING-LENGTH: The length of the string COBW3-PUT-STRING in bytes. Multiple spaces between words are considered as a single space, and a string of all spaces is considered to be zero length. The default value for COBW3-PUT-STRING-LENGTH is zero. Range of values is 0 to 1024.

COBW3-CONTENT-TYPE: Sets the HTTP content header to plain text or HTML. Possible values are shown in the following table.

Table 2.9. Values for COBW3-CONTENT-TYPE

Mode	Set Value	Description
COBW3-CONTENT-TYPE-HTML	text/html	This is the default value. Used for HTML text
COBW3-CONTENT-TYPE-TEXT	text/plain	Used for plain text.

Return Values

None.

Example

```
MOVE FUNCTION LENG(HTML-IMAGE)          TO    COBW3-PUT-STRING-LENGTH.
MOVE HTML-IMAGE                         TO    COBW3-PUT-STRING.
CALL "COBW3_PUT_TEXT"                   USING COBW3.
```

Chapter 3. Using the COBOL Cartridge

This chapter explains the mechanics of how to write a COBOL program that uses the COBOL Cartridge. You will learn how to write some very simple, but useful programs that demonstrate how to use the subroutines that are described in Chapter 2.

Setting Up the Development Environment

Before you begin to write your application, you will need to set up your development environment. Chapter 1 explains how to install the COBOL Cartridge, and configure the listener. The examples in this chapter will use the "zip" listener that runs on the domain "spice", and uses port 7777. The zip listener has a virtual directory "zip-doc". This is where the HTML files that are associated with the examples for COBOL development will go. This chapter will rely on one HTML file called COBOL.HTM to run all the examples.

You should be familiar with the process of creating a *.dll* (Dynamic-link library) for Windows NT, or a *.so* (Shared Object) file. These examples were created on the Windows NT operating system, so the examples will cover .dll (DLL) files. The creation and placement of .so files is the same on a UNIX operating system.

You will see in the Zip Listener Virtual Directory Configuration page that the virtual directory /zip-bin/ is associated with the physical directory: `c:\ORANT\ows\admin\ows\zip\httpd_spice\zip\cobol\bin\ /zip-bin/`. This is where the DLL files are kept. You will likely want to organize your files in a similar manner so the COBOL Cartridge will run your DLL files from a common area.

Looking at Development Techniques

Programmers have their own software development habits. Fujitsu has worked hard to create an integrated environment for software development that frees the programmer from the menial tasks of working with complicated scripts, and keeping track of the files that make up a project. However, when integrating your COBOL application with the Oracle Web

Application Server, you will be required to manually move your DLL files from your project area to the directory that the COBOL Cartridge will use to link in your program at run-time.

New users will need to become familiar with the behavior of the Oracle Web Application Server. During the development process, the listener or the Web Request Broker will sometimes hang due to minor indiscretions within your development code. Examples of this include the creation of an endless loop in your DLL or exceeding the bounds of an array. In some cases you will not know if the WRB or Listener is hung until you attempt to overwrite your development DLL with a new DLL. In this case the operating system will prevent this from happening because the file is in use. This has the potential to be confusing if you are unprepared for such eventualities. The remedy is to bounce the Web Application Server to free up the hung system. (See Chapter 1 for details on how to bounce the Web Application Server.)

Using the Examples HTML Page

All of the examples in this chapter may be run through the COBOL.HTM file. For the configuration used in this book, you can display the Examples page by entering the following URL:

```
http://spice:7777/zip-doc/cobol.htm
```

This will display the following Web page:

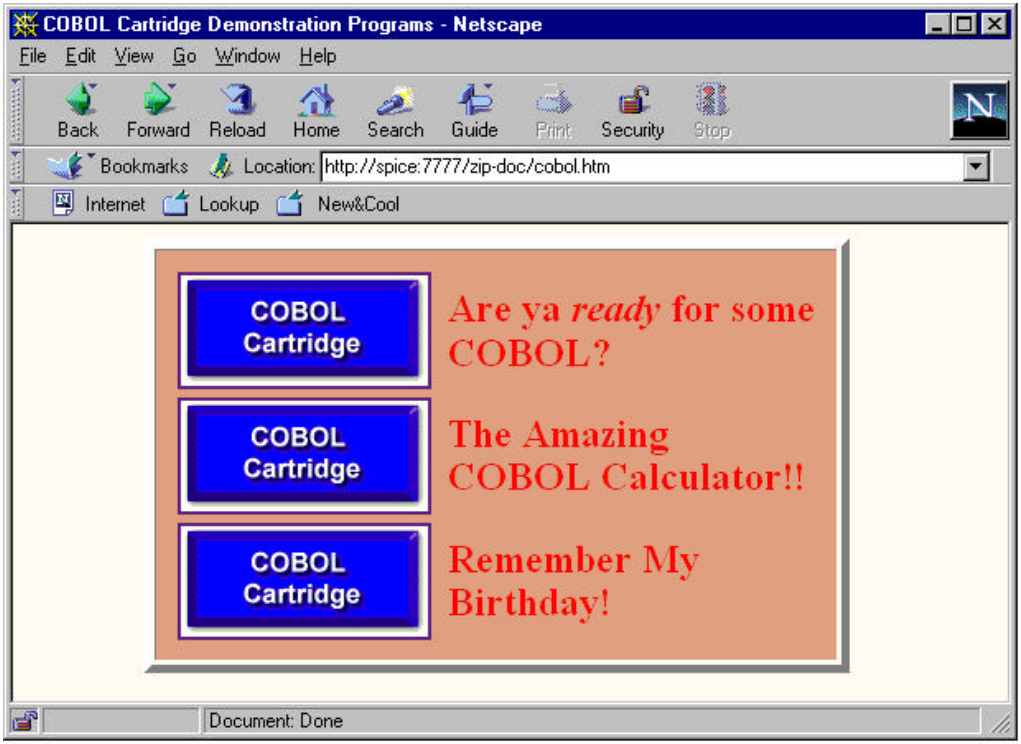

Figure 3.1. The Examples page to run the examples produced by file COBOL.HTM

The HTML source listing that created this Web page shows how the COBOL Cartridge is called, and how the DLL files are linked. The listing below shows the file COBOL.HTM. Notice the structure of the URL that is associated with each of the buttons in the menu.

```
<HTML>
<HEAD>
<TITLE>COBOL Cartridge Demonstration Programs</TITLE>
</HEAD>
<BODY   BGCOLOR="#EEEEEE" >
<CENTER>
<TABLE BORDER=5 WIDTH=75% CELLPADDING=8 BGCOLOR=#FFAF79
  BORDERCOLORDARK=#00A000 BORDERCOLORLIGHT=#00FFFF><TR><TD>
  <TABLE>
  <TR><TD>
    <A HREF="HTTP://spice:7777/zip-bin/cobol/WEBSTER.DLL/WEBSTER">
    <IMG SRC="/zip-img/Cobol/cobol_cart.gif" ALIGN=CENTER></A>
  </TD><TD>
    <A NAME="POST"><FONT SIZE=5 COLOR="RED">
    <B>Are ya <I>ready</I> for some COBOL!!</FONT></B></A>
  </TD></TR>
  <TR><TD>
   <A HREF="HTTP://spice:7777/zipbin/cobol/calculator.DLL/CALCULATOR">
   <IMG SRC="/zip-img/Cobol/cobol_cart.gif" ALIGN=CENTER></A>
  </TD><TD>
    <A NAME="POST"><FONT SIZE=5 COLOR="RED">
    <B>The Amazing COBOL Calculator!!</FONT></B></A>
  </TD></TR>
  <TR><TD>
    <A HREF="HTTP://spice:7777/zip-bin/cobol/birthday.DLL/BIRTHDAY">
    <IMG SRC="/zip-img/Cobol/cobol_cart.gif" ALIGN=CENTER></A>
  </TD><TD>
    <A NAME="POST"><FONT SIZE=5 COLOR="RED">
    <B>Remember My Birthday!</FONT></B></A>
  </TD></TR>
  </TABLE>
  </TD></TR>
</TABLE>
</CENTER>
</BODY>
</HTML>
```

How a URL Invokes a COBOL Application

The previous example demonstrates how the URL calls a COBOL program. The following URL requests the WEBSTER program:

```
HTTP://spice:7777/zip-bin/cobol/WEBSTER.DLL/WEBSTER">
```

The sub-string HTTP://spice:7777 in the URL signals the Web browser to connect to the domain called spice on port 7777 using

the HTTP protocol. . When the Oracle Web Listener that is running on the spice server and assigned to port 7777 receives the request, the sub-string `/zip-bin/cobol/WEBSTER.DLL/WEBSTER` signals the Web Request Broker to connect to the COBOL cartridge. The /zip-bin/ portion of the URL specifies the virtual directory containing the DLL file that the COBOL Cartridge will process. The /cobol/ entry is the name of your COBOL Cartridge. (This is the name you assigned it during the installation in Chapter 1.) WEBSTER.DLL is the file name of the DLL to be run, and WEBSTER is the entry point that was declared in that program.

How to Display an HTML Page with COBOL

This example makes use of the COBOL WRB API subroutine COBW3_PUT_TEXT to format an HTML page. The HTML page is static, and is best created by scripting an HTML text file. This example will show you the basics of HTML page construction although efficiency is compromised for purposes of demonstration.

When you press the COBOL Cartridge button for the “Are ya ready for some COBOL?” selection, the following Web page will be displayed in your browser.

Figure 3.2. The web page created by the WEBSTER program

The code that created this page is shown on the following pages. As you examine the code, notice the steps that are necessary to write a COBOL Cartridge DLL.

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. WEBSTER.
000030*********************************************
000040*  Crafted by SHY
000050*********************************************
000060 DATA DIVISION.
000070 WORKING-STORAGE SECTION.
```

```
000080 01  HTML-HEADER              PIC X(15)  VALUE "<HTML><HEAD>".
000090 01  HTML-TITLE               PIC X(50)  VALUE "<TITLE>Hello, I'm
Webster!</TITLE></HEAD>".
000100 01  HTML-BODY                PIC X(50)  VALUE "<BODY BGCOLOR=#EEEEEE >".
000110 01  HTML-CENTER-OPEN         PIC X(20)  VALUE "<CENTER>".
000120 01  HTML-TEXT1               PIC X(80)  VALUE "<FONT COLOR=#0000FF
SIZE=8>" &
000130                                          "<B>Hello, I'm
Webster!</B></FONT>".
000140 01  HTML-IMAGE               PIC X(50)  VALUE "<BR><IMG SRC=/zip-
img/angel.gif ALIGN=top>".
000150 01  HTML-TEXT2               PIC X(80)  VALUE "<BR><FONT COLOR=#0000FF
SIZE=7>" &
000160                                          "<B>I program Fujitsu
COBOL!</B></FONT>".
000170 01  HTML-CENTER-CLOSE        PIC X(15)  VALUE "</CENTER>".
000180 01  HTML-END-OF-HEADER       PIC X(20)  VALUE "</BODY></HTML>".
000190*============================================================*
000200*  WRB API ROUTINE WORK AREA                                 *
000210*============================================================*
000220     COPY  COBW3.
000230
000240 LINKAGE SECTION.
000250 01  WRBCTX                   POINTER.
000260
000270*------------------------------------------------------------
000280 PROCEDURE DIVISION USING WRBCTX.
000281     MOVE WRBCTX TO COBW3-CONTEXT
000300     CALL "COBW3_INIT" USING COBW3
000310
000320     MOVE FUNCTION LENG(HTML-HEADER) TO COBW3-PUT-STRING-LENGTH
000330     MOVE HTML-HEADER                TO COBW3-PUT-STRING
000340     CALL "COBW3_PUT_TEXT" USING COBW3
000350
000360     MOVE FUNCTION LENG(HTML-TITLE) TO COBW3-PUT-STRING-LENGTH
000370     MOVE HTML-TITLE                TO COBW3-PUT-STRING
000380     CALL "COBW3_PUT_TEXT" USING COBW3
000390
000400     MOVE FUNCTION LENG(HTML-BODY) TO COBW3-PUT-STRING-LENGTH
000410     MOVE HTML-BODY                TO COBW3-PUT-STRING
000420     CALL "COBW3_PUT_TEXT" USING COBW3
000430
000440     MOVE FUNCTION LENG(HTML-CENTER-OPEN) TO COBW3-PUT-STRING-LENGTH
000450     MOVE HTML-CENTER-OPEN                TO COBW3-PUT-STRING
000460     CALL "COBW3_PUT_TEXT" USING COBW3
000470
000480     MOVE FUNCTION LENG(HTML-TEXT1) TO COBW3-PUT-STRING-LENGTH
000490     MOVE HTML-TEXT1                TO COBW3-PUT-STRING
000500     CALL "COBW3_PUT_TEXT" USING COBW3
000510
000520     MOVE FUNCTION LENG(HTML-IMAGE) TO COBW3-PUT-STRING-LENGTH
000530     MOVE HTML-IMAGE                TO COBW3-PUT-STRING
000540     CALL "COBW3_PUT_TEXT" USING COBW3
000550
000560     MOVE FUNCTION LENG(HTML-TEXT2) TO COBW3-PUT-STRING-LENGTH
000570     MOVE HTML-TEXT2                TO COBW3-PUT-STRING
000580     CALL "COBW3_PUT_TEXT" USING COBW3
```

```
000600     MOVE FUNCTION LENG(HTML-CENTER-CLOSE) TO COBW3-PUT-STRING-LENGTH
000610     MOVE HTML-CENTER-CLOSE                TO COBW3-PUT-STRING
000620     CALL "COBW3_PUT_TEXT" USING COBW3
000630
000640     MOVE FUNCTION LENG(HTML-END-OF-HEADER) TO COBW3-PUT-STRING-LENGTH
000650     MOVE HTML-END-OF-HEADER                TO COBW3-PUT-STRING
000660     CALL "COBW3_PUT_TEXT" USING COBW3
000670
000680     CALL "COBW3_FREE" USING COBW3
000690
000700     EXIT PROGRAM.
```

Identifying the Components of the WEBSTER Program

The WEBSTER program demonstrates rudimentary steps in creating any COBOL Cartridge DLL. As you examine the listing, you can identify the following sections:

1. The WORKING-STORAGE section of the program builds the HTML strings that will be sent back to the COBOL Cartridge.
2. The following lines set up the COBW3 environment area. This serves as a communication area that is common to the DLL and the COBOL Cartridge, and must appear in each DLL.

```
000220     COPY  COBW3.
000230
000240 LINKAGE SECTION.
000250 01  WRBCTX                POINTER.
```

3. The following lines show how the COBW3 data area is initialized.

```
000280 PROCEDURE DIVISION USING WRBCTX.
000281     MOVE WRBCTX TO COBW3-CONTEXT
000300     CALL "COBW3_INIT" USING COBW3
```

4. The body of the program is composed of sets of “code triplets” that repeat a process of stuffing pre-defined strings of HTML into the COBW3 environment area. The following code is an example of a “code triplet”.

```
000480     MOVE FUNCTION LENG(HTML-TEXT1) TO COBW3-PUT-STRING-LENGTH
```

```
000400     MOVE HTML-TEXT1 TO COBW3-PUT-STRING
000500     CALL "COBW3_PUT_TEXT" USING COBW3
```

5. The final step in the program is to release the COBW3 environment area that was reserved in step 1. The following line does this.

```
000680     CALL "COBW3_FREE" USING COBW3
```

This is the simplest program to write, and may be suitable for tasks that are short and need to be completed quickly, as it requires little technical knowledge to write. However, in the following examples, you may want to choose another approach that may be more suitable for your programming requirements.

How to Code an HTML Form Using COBOL

The next example demonstrates “The Amazing COBOL Calculator”. This program produces an HTML form and will process a user’s input from that form in the COBOL program. Pressing the second button in the Examples page displays the following Web page.

This program will accept a decimal number that a user enters as input, and return the hexadecimal, octal, and binary values of that number.

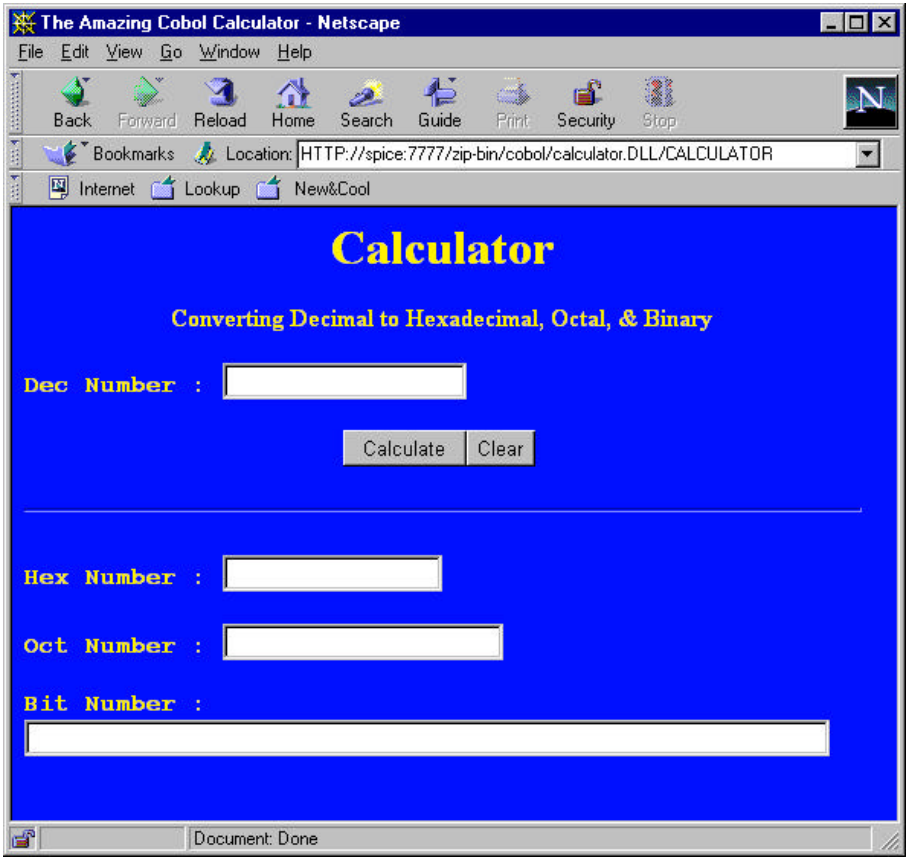

Figure 3.3. The web page created by the CALCULATOR program

The COBOL source that created this form, and processes the program input is shown in the following listing.

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. CALCULATOR.
000030*******************************************
000040* Crafted by SHY
000050*******************************************
```

```
000060 ENVIRONMENT DIVISION.
000070 CONFIGURATION SECTION.
000080 SPECIAL-NAMES.
000090     SYMBOLIC CONSTANT
000100     SC-HTML-FILE-NAME         IS
"c:\orant\ows\admin\ows\zip\httpd_spice\zip\cobol\calculator.htm"
000110     SC-CNV-DECNUMBER          IS "CNV-DECNUMBER"
000120     SC-CNV-HEXNUMBER          IS "CNV-HEXNUMBER"
000130     SC-CNV-OCTNUMBER          IS "CNV-OCTNUMBER"
000140     SC-CNV-BITNUMBER          IS "CNV-BITNUMBER"
000150     SC-CNV-MESSAGE            IS "CNV-MESSAGE"
000160     SC-COBW3-SEARCH-NOT-FOUND IS "O"
000170     SC-COBW3-SEARCH-FOUND     IS "1"
000180     SC-NUMBER-OF-HEX          IS 16
000190     SC-NUMBER-OF-OCT          IS 21
000200     SC-NUMBER-OF-BIT          IS 64
000210     SC-WRB-INVALID-INPUT      IS "Invalid data. Please reenter the
data.".
000220 DATA DIVISION.
000230 WORKING-STORAGE SECTION.
000240 01  INPUT-NUMBER           PIC 9(18) VALUE ZERO.
000250
000260 01  BINARY-NUMBER          PIC 9(18) BINARY VALUE ZERO.
000270 01  REDEFINES BINARY-NUMBER.
000280     02  BINARY-HEX OCCURS SC-NUMBER-OF-HEX TIMES    PIC 1(4) BIT.
000290 01    REDEFINES BINARY-NUMBER.
000300     02  FILLER                                      PIC 1 BIT.
000310     02  BINARY-OCT OCCURS SC-NUMBER-OF-OCT TIMES    PIC 1(3) BIT.
000320 01    REDEFINES BINARY-NUMBER.
000330     02  BINARY-BIT OCCURS SC-NUMBER-OF-BIT TIMES    PIC 1 BIT.
000340
000350 01  HEX-EXPESSION                 PIC X(16) VALUE "0123456789ABCDEF".
000360 01  REDEFINES HEX-EXPRESSION.
000370     02  HEX-DATA OCCURS 16 TIMES   PIC X.
000380
000390 01  OCT-EXPRESSION                 PIC X(8) VALUE "01234567".
000400 01  REDEFINES OCT-EXPRESSION.
000410     02  OCT-DATA OCCURS 8 TIMES    PIC X.
000420 01  I                              PIC 9(4) COMP-5 VALUE ZERO.
000430
000440 01  HEX-TO-BINARY.
000450     02  FILLER             PIC 1(12)  BIT VALUE ALL B"0".
000460     02  HEX-LOWER-PART     PIC 1(4)   BIT.
000470 01  BINARY-FROM-HEX REDEFINES HEX-TO-BINARY PIC 9(4) BINARY.
000480
000490 01  OCT-TO-BINARY.
000500     02  FILLER             PIC 1(13)  BIT VALUE ALL B"0".
000510     02  OCT-LOWER-PART     PIC 1(3)   BIT.
000520 01  BINARY-FROM-OCT REDEFINES OCT-TO-BINARY PIC 9(4) BINARY.
000530
000540 01  FILLER                 PIC X VALUE "N".
000550     88  Z-DEC-NODATA       VALUE "N".
000560     88  Z-DEC-CORRECT      VALUE "C".
000570     88  Z-DEC-INVALID      VALUE "I".
000580
000590 01  DECNUMBER-SAVE         PIC X(18) VALUE SPACES.
000600
```

```
000610 01  HEX-ANSWER              PIC X(16) VALUE SPACES.
000620 01  OCT-ANSWER              PIC X(21) VALUE SPACES.
000630 01  BIT-ANSWER              PIC X(64) VALUE SPACES.
000640
000650 01  HTML-MESSAGE            PIC X(100) VALUE SPACES
000660*============================================================*
000670*  WRB API ROUTINE WORK AREA                                 *
000680*============================================================*
000690     COPY  COBW3.
000700
000710 CONSTANT SECTION.
000720 01  C-DECNUMBER             PIC X(9) VALUE "DECNUMBER".
000730
000740 LINKAGE SECTION.
000750 01  WRBCTX                  POINTER.
000760
000770*-------------------------------------------------------------
000780 PROCEDURE DIVISION USING WRBCTX.
000790     PERFORM INITIALZE-ROUTINE
000800     PERFORM DECIMAL-CHECK
000810     IF Z-DEC-CORRECT
000820        PERFORM CONVERT-DATA
000830     END-IF
000840     PERFORM OUTPUT-HTML
000850     PERFORM TERMINATE-ROUTINE.
000860     EXIT PROGRAM.
000870*-------------------------------------------------------------
000880 INITIALZE-ROUTINE.
000890     MOVE LOW-VALUE  TO COBW3
000900     MOVE WRBCTX     TO COBW3-CONTEXT
000910     CALL "COBW3_INIT" USING COBW3
000920     SET  Z-DEC-NODATA TO TRUE
000930     MOVE SPACES TO HTML-MESSAGE.
000940
000950*-------------------------------------------------------------
000960 DECIMAL-CHECK.
000970     MOVE C-DECNUMBER  TO COBW3-SEARCH-DATA
000980     MOVE 1            TO COBW3-NUMBER
000990     CALL "COBW3_NAME" USING COBW3
001000     IF COBW3-SEARCH-FLAG = SC-COBW3-SEARCH-NOT-FOUND OR
001010        COBW3-GET-DATA = SPACE
001020     THEN
001030        SET Z-DEC-INVALID TO TRUE
001040     ELSE
001050        MOVE COBW3-GET-DATA TO DECNUMBER-SAVE
001060        IF COBW3-GET-DATA (1 : COBW3-GET-LENGTH) IS NOT NUMERIC
001070           MOVE SC-WRB-INVALID-INPUT TO HTML-MESSAGE
001080           SET Z-DEC-INVALID TO TRUE
001090        ELSE
001100           SET Z-DEC-CORECT TO TRUE
001110        END-IF
001120     END-IF
001130
001140     IF Z-DEC-CORRECT
001150        MOVE ALL ZERO TO INPUT-NUMBER
001160       PERFORM TEST BEFORE VARYING I FROM COBW3-GET-LENGTH BY -1 UNTIL
I < 1
```

```
001170          MOVE COB3-GET-DATA (I : 1) TO INPUT-NUMBER (18 - COBW3-GET-
LENGTH + I : 1)
001180        END-PERFORM
001190        MOVE INPUT-NUMBER TO BINARY-NUMBER
001200     END-IF.
001210
001220*-------------------------------------------------------------
001230 CONVERT-DATA.
001240     PRFORM TEST BEFORE VARYING I FROM 1 BY 1 UNTIL I > SC-NUMBER-OF-
HEX
001250       MOVE  BINARY-HEX (I)                 TO HEX-LOWER-PART
001260       MOVE  HEX-DATA (BINARY-FROM-HEX + 1) TO HEX-ANSWER (I : 1)
001270     END-PERFORM
001280
001290     PRFORM TEST BEFORE VARYING I FROM 1 BY 1 UNTIL I > SC-NUMBER-OF-
OCT
001300       MOVE  BINARY-OCT (I)                 TO OCT-LOWER-PART
001310       MOVE  OCT-DATA (BINARY-FROM-OCT + 1) TO OCT-ANSWER (I : 1)
001320     END-PERFORM
001330
001340     PRFORM TEST BEFORE VARYING I FROM 1 BY 1 UNTIL I > SC-NUMBER-OF-
BIT
001350       MOVE  BINARY-BIT (I)                 TO BIT-ANSWER (I : 1)
001360     END-PERFORM.
001370
001380*-------------------------------------------------------------
001390 OUTPUT-HTML.
001400
001410     MOVE SPACES TO COBW3-CNV-VALUE
001420     IF NOT Z-DEC-NODATA
001430        STRING  'VALUE="'         DELIMITED BY SIZE
001440                DECNUMBER-SAVE    DELIMITED BY SPACE
001450                '"'               DELIMITED BY SIZE
001460           INTO COBW3-CNV-VALUE
001470        END-STRING
001480     END-IF
001490     MOVE SC-CNV-DECNUMBER TO COBW3-CNV-NAME
001500     CALL "COBW3_SET_CNV"  USING COBW3
001510
001520     MOVE SPACES TO COBW3-CNV-VALUE
001530     IF Z-DEC-CORRECT
001540        STRING  'VALUE="'         DELIMITED BY SIZE
001550                HEX-ANSWER        DELIMITED BY SPACE
001560                '"'               DELIMITED BY SIZE
001570           INTO COBW3-CNV-VALUE
001580        END-STRING
001590     END-IF
001600     MOVE SC-CNV-HEXNUMBER TO COBW3-CNV-NAME
001610     CALL "COBW3_SET_CNV"  USING COBW3
001620
001630     MOVE SPACES TO COBW3-CNV-VALUE
001640     IF Z-DEC-CORRECT
001650        STRING  'VALUE="'         DELIMITED BY SIZE
001660                OCT-ANSWER        DELIMITED BY SPACE
001670                '"'               DELIMITED BY SIZE
001680           INTO COBW3-CNV-VALUE
001690        END-STRING
```

```
001700     END-IF
001710     MOVE SC-CNV-OCTNUMBER TO COBW3-CNV-NAME
001720     CALL "COBW3_SET_CNV"  USING COBW3
001730
001740     MOVE SPACES TO COBW3-CNV-VALUE
001750     IF Z-DEC-CORRECT
001760        STRING  'VALUE="'         DELIMITED BY SIZE
001770                BIT-ANSWER        DELIMITED BY SPACE
001780                '"'               DELIMITED BY SIZE
001790           INTO COBW3-CNV-VALUE
001800        END-STRING
001810     END-IF
001820     MOVE SC-CNV-BITNUMBER TO COBW3-CNV-NAME
001830     CALL "COBW3_SET_CNV"  USING COBW3
001840
001850     MOVE HTML-MESSAGE     TO COBW3-CNV-VALUE
001860     MOVE SC-CNV-MESSAGE   TO COBW3-CNV-NAME
001870     CALL "COBW3_SET_CNV"  USING COBW3
001880
001890     MOVE SC-HTML-FILE-NAME TO COBW3-HTML-FILENAME
001900     CALL "COBW3_PUT_HTML"  USING COBW3.
001910
001920*-------------------------------------------------------
001930 TERMINATE-ROUTINE.
001940     CALL "COBW3_FREE" USING COBW3
```

Identifying the Components of the CALCULATOR Program

The CALCULATOR program introduces error checking, reading data from an HTML form, and writing data back to the form. You will notice that this program is similar to the WEBSTER program as it initializes the COBW3 environment and frees it at the end. The CALCULATOR program is rather unique in how it was designed. Rather than publishing an HTML form each time, it relies on the presence of an HTML file, and merely updates this HTML file (using the COBW3_CNV subroutines) to reflect changes made by the user. This innovative program design results in a 20% reduction in the lines of source code (about 50 lines) for this example. A calculator example that doesn't use the COBW3_CNV subroutines is in Appendix B, "Listing of CALCULATE_BIG.COB."

You will also note that this program is modularized. Lines 780 to 860 clearly state what is happening:

- Initialize the data
- Check input data for errors
- If the input data is valid, perform the conversion
- Output the HTML file
- Terminate the routine

The HTML file is self-explanatory, but when you study the listing of the CALCULATOR program it is useful to look at the special tags that are updated by the COBW3_CNV subroutines. The HTML file CALCULATOR.HTM is stored in the virtual directory /zip-cobol/ when the program runs. This file is shown below, and the special tags are denoted by bold type.

```
<HTML>
<HEAD>
<TITLE>The Amazing Cobol Calculator</TITLE>
</HEAD>
<BODY BGCOLOR="#000FFF" TEXT="#FFF000">
<FORM METHOD="GET"
ACTION="http://spice:7777/zip-bin/cobol/calculator.DLL/CALCULATOR">
<P>
<CENTER>
<H1>Calculator</H1>
<H4>Converting Decimal to Hexadecimal, Octal, & Binary</H4>
</CENTER>
</P>
<P>
<STRONG><FONT FACE="Courier New">Dec Number : </FONT></STRONG>
<INPUT TYPE="text" NAME="DECNUMBER" SIZE="18"
//COBOL//CNV-DECNUMBER//COBOL//>
</P>
<P>
<CENTER>
<INPUT TYPE="SUBMIT" VALUE="Calculate"><INPUT TYPE="reset" VALUE="Clear">
</CENTER>
</P>
<HR>
<P>
<STRONG><FONT FACE="Courier New">Hex Number : </FONT></STRONG>
<INPUT TYPE="text" NAME="HEXNUMBER" SIZE="16"
//COBOL//CNV-HEXNUMBER//COBOL//>
</P>
<P>
<STRONG><FONT FACE="Courier New">Oct Number : </FONT></STRONG>
<INPUT TYPE="text" NAME="OCTNUMBER" SIZE="21"
//COBOL//CNV-OCTNUMBER//COBOL//>
</P>
```

```
<P>
<STRONG><FONT FACE="Courier New">Bit Number : </FONT></STRONG>
<INPUT TYPE="text" NAME="BITNUMBER" SIZE="64"
//COBOL//CNV-BITNUMBER//COBOL//>
</P>
<P><BLINK>//COBOL//CNV-MESSAGE//COBOL//</BLINK></P>
</FORM>
</BODY>
</HTML>
```

Looking at the CALCULATOR.COB listing, you can see that symbolic constants are specified with the prefix of “SC” to be used locally. Note that on line 100, we included a *full* pathname to the HTML source file that is listed above. Had we left off the path to the file, the COBOL Cartridge would have defaulted to the default directory that is set in the HTTP environmental variable. (This would be the c:\WINNT\system32 directory for the NT environment.) The cartridge parameter CurrentDIR (set on the Cartridge Configuration page of the WRB administrative tool) will set this default for the COBOL Cartridge. However, at the time this book is being published, the feature using the CurrentDIR is not working, so the full pathname was used in the CALCULATOR program.

Lines 890 to 910 initialize the environment for the COBOL Cartridge. The defaults are used for COBW3_INIT, and this sets the content type to text/HTML, and the debugging mode is turned off.

Line 990 retrieves the first value from the COBW3 environment area. The “IF” logic beginning on line 1000 checks to see if anything was found by checking the COBW3-SEARCH-FLAG. This flag is set whenever a call is made to the COBW3-NAME procedures are called.

The “IF” construct in lines 1000 to 1020 check the input number to be sure a value was input. Lines 1050 to 1120 then check to see if the value is numeric. If it is, then lines 1140 to 1200 prepare the input number for conversion.

Lines 1330 to 1360 account for the mathematical calculations necessary to convert the data from decimal to hexadecimal (lines 1240-1270), octal (lines 1290-1320) and binary (lines 1340 to 1360).

Lines 1410 to 1900 output the conversion directives to the HTML file. Either the conversion answers are sent, or error messages are displayed indicating the user entered something invalid. This section is a good example of how the COBW3_CNV routines work. This section of code produces the dynamic HTML portion of the program.

How to Call the PL/SQL Cartridge from the COBOL Cartridge

Calling another cartridge from your COBOL program is one of the more powerful features of using the Oracle Web Application Server. It allows you to easily exploit the strong features that are inherited in that cartridge. Most users will find it more convenient to use the PL/SQL Cartridge to communicate to the database server rather than using Pro*COBOL. You may use the techniques in this example to use the other cartridges, such as Perl (useful for system calls and text processing), Java, LiveHTML, ODBC, and others.

This example creates a form to enter a birthday, using a COBOL program, and then sends that data to a PL/SQL procedure (insertBirthday) to insert the data into the database. When you select the third button on the Examples page you will see the following Web page.

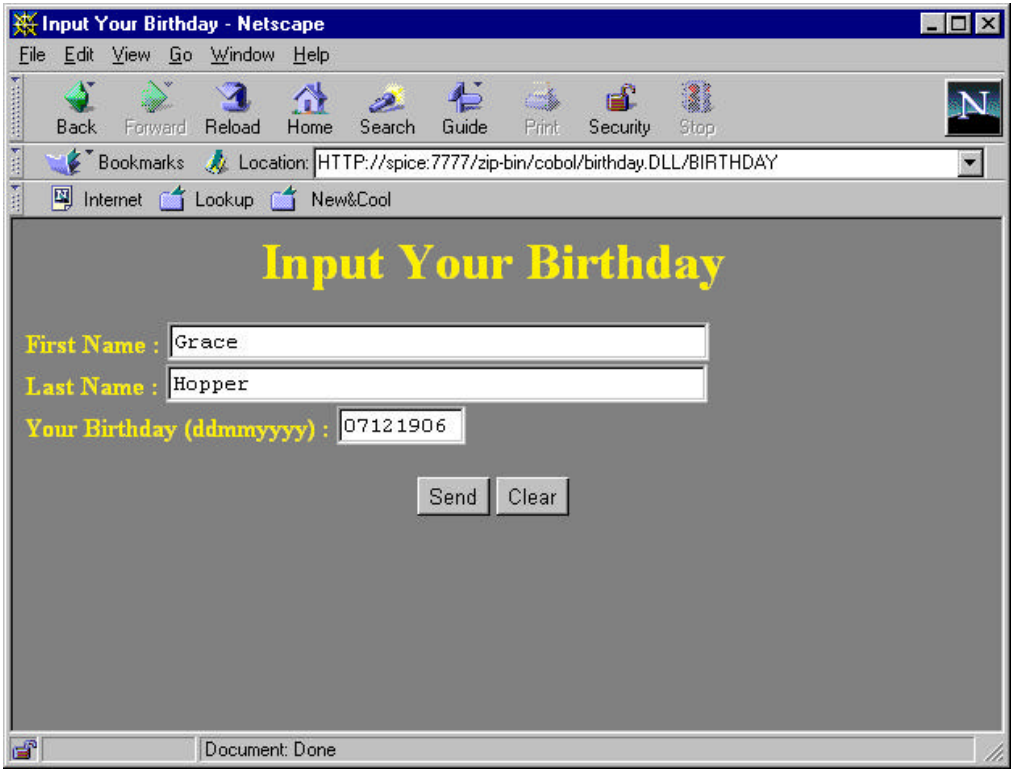

Figure 3.4. The web page created by the BIRTHDAY program

The COBOL source code that produced this page is shown in the following listing.

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. BIRTHDAY.
000030*********************************************
000040* Crafted by SHY
000050*********************************************
000060 ENVIRONMENT DIVISION.
000070 CONFIGURATION SECTION.
000080 SPECIAL-NAMES.
000090     SYMBOLIC CONSTANT
000100     SC-INPUT-FIRSTNAME     IS "Please input first name."
000110     SC-INPUT-LASTNAME      IS "Please input last name."
000120     SC-INPUT-DOB           IS "Please input date of birth."
000130     SC-INVALID-DAY         IS "Invalid day. Please try again."
000140     SC-INVALID-MONTH       IS "Invalid month. Please try again."
000150     SC-INVALID-YEAR        IS "Invalid year. Please try  again."
000160     SC-CNV-FIRSTNAME       IS "CNV-FIRSTNAME"
```

```
000170     SC-CNV-LASTNAME          IS "CNV-LASTNAME"
000180     SC-CNV-BIRTHDAY          IS "CNV-BIRTHDAY"
000190     SC-CNV-MESSAGE           IS "CNV-MESSAGE"
000200     SC-HTML-FILE-NAME        IS
"c:\orant\ows\admin\ows\zip\httpd_spice\zip\cobol\birthday.htm".
000210 DATA DIVISION.
000220 WORKING-STORAGE SECTION.
000230 01  SENDING-STRING          PIC X(200) VALUE SPACES.
000240
000250 01  HTML-MESSAGE            PIC X(200) VALUE SPACES.
000260
000270 01  FIRST-NAME              PIC X(40) VALUE SPACES.
000280 01  LAST-NAME               PIC X(40) VALUE SPACES.
000290 01  DATE-OF-BIRTH.
000300     02  DOB-DAY             PIC 9(2) VALUE ZERO.
000310     02  DOB-MONTH           PIC 9(2) VALUE ZERO.
000320     02  DOB-YEAR            PIC 9(4) VALUE ZERO.
000330
000340 01  DATE-OF-BIRTH-FORMAT.
000350     02  DOB-DAY             PIC 9(2) VALUE ZERO.
000360     02  FILLER              PIC X VALUE '-'.
000370     02  DOB-MONTH           PIC X(3) VALUE ZERO.
000380     02  FILLER              PIC X VALUE '-'.
000390     02  DOB-YEAR            PIC 9(2) VALUE ZERO.
000400
000410 01  PLSQL-ADDRESS-STRING    PIC X(67) VALUE 'Location:
http://spice:7777/agent_zip/plsql/' &
000420                                        'insertBirthday?p_first='.
000430 01  LAST-NAME-STRING        PIC X(8)  VALUE '&p_last='.
000440 01  DATE-OF-BIRTH-STRING    PIC X(7)  VALUE '&p_dob='.
000450
000460 01  FILLER.
000470     02  MONTH-NAME-TABLE    PIC X(36) VALUE
'JANFEBMARAPRMAYJUNJULAUGSEPOCTNOVDEC'.
000480     02  REDEFINES MONTH-NAME-TABLE.
000490         03  MONTH-NAME  OCCURS 12 TIMES PIC X(3).
000500
000510 01  FILLER.
000520     02  NUMBER-OF-DAYS-FOR-EACH-MONTH.
000530         03  MONTH-JANUARY       PIC 9(2) VALUE 31.
000540         03  MONTH-FEBRUARY      PIC 9(2) VALUE 28.
000550         03  MONTH-MARCH         PIC 9(2) VALUE 31.
000560         03  MONTH-APRIL         PIC 9(2) VALUE 30.
000570         03  MONTH-MAY           PIC 9(2) VALUE 31.
000580         03  MONTH-JUNE          PIC 9(2) VALUE 30.
000590         03  MONTH-JULY          PIC 9(2) VALUE 31.
000600         03  MONTH-AUGUST        PIC 9(2) VALUE 31.
000610         03  MONTH-SEPTEMBER     PIC 9(2) VALUE 30.
000620         03  MONTH-OCTOBER       PIC 9(2) VALUE 31.
000630         03  MONTH-NOVEMBER      PIC 9(2) VALUE 30.
000640         03  MONTH-DECEMBER      PIC 9(2) VALUE 31.
000650     02  REDEFINES NUMBER-OF-DAYS-FOR-EACH-MONTH.
000660         03  NUMBER-OF-DAYS OCCURS 12 TIMES PIC 9(2).
000670
000680 01  MAX-DAYS-OF-THE-MONTH       PIC 9(2) VALUE 31.
000690
000700 01  FILLER                      PIC X VALUE "N".
```

```
000710     88  CN-FIRSTNAME-SET        VALUE 'S'.
000720     88  CN-FIRSTNAME-NO-SET     VALUE 'N'.
000730 01  FILLER                      PIC X VALUE "N".
000740     88  CN-LASTNAME-SET         VALUE 'S'.
000750     88  CN-LASTNAME-NO-SET      VALUE 'N'.
000760 01  FILLER                      PIC X VALUE "N".
000770     88  CN-DOB-SET              VALUE 'S'.
000780     88  CN-DOB-NO-SET           VALUE 'N'.
000790 01  FILLER                      PIC X VALUE "I".
000800     88  CN-INVALID-DAY          VALUE 'I'.
000810     88  CN-VALID-DAY            VALUE 'V'.
000820 01  FILLER                      PIC X VALUE "I".
000830     88  CN-INVALID-MONTH        VALUE 'I'.
000840     88  CN-VALID-MONTH          VALUE 'V'.
000850 01  FILLER                      PIC X VALUE "I".
000860     88  CN-INVALID-YEAR         VALUE 'I'.
000870     88  CN-VALID-YEAR           VALUE 'V'.
000880
000890*============================================================*
000900*  WRB API ROUTINE WORK AREA                                 *
000910*============================================================*
000920     COPY  COBW3.
000930
000940 CONSTANT SECTION.
000950 01  C-FIRSTNAME            PIC X(9) VALUE 'FIRSTNAME'.
000960 01  C-LASTNAME             PIC X(8) VALUE 'LASTNAME'.
000970 01  C-DATE-OF-BIRTH        PIC X(8) VALUE 'BIRTHDAY'.
000980
000990 LINKAGE SECTION.
001000 01  WRBCTX                 POINTER.
001010*------------------------------------------------------------
001020 PROCEDURE DIVISION USING WRBCTX.
001030
001040     PERFORM INITIALZE-ROUTINE
001050     PERFORM CHECK-INPUT
001060     IF CN-FIRSTNAME-NO-SET OR
001070        CN-LASTNAME-NO-SET OR
001080        CN-DOB-NO-SET OR
001090        CN-INVALID-DAY OR
001100        CN-INVALID-MONTH OR
001110        CN-INVALID-YEAR
001120     THEN
001130        PERFORM OUTPUT-HTML
001140     ELSE
001150        PERFORM EXECUTE-PLSQL
001160     END-IF
001170     PERFORM TERMINATE-ROUTINE
001180     EXIT PROGRAM.
001190*------------------------------------------------------------
001200 INITIALZE-ROUTINE SECTION.
001210     MOVE LOW-VALUE TO COBW3
001220     MOVE WRBCTX    TO COBW3-CONTEXT
001230     CALL "COBW3_INIT" USING COBW3
001240
001250     MOVE SPACE TO FIRST-NAME LAST-NAME DATE-OF-BIRTH
001260     MOVE SPACE TO HTML-MESSAGE.
001270
```

```
001280*-----------------------------------------------------------------
001290 CHECK-INPUT-SECTION.
001300     MOVE C-FIRSTNAME  TO COBW3-SEARCH-DATA
001310     MOVE 1            TO COBW3-NUMBER
001320     CALL "COBW3_NAME" USING COBW3
001330     IF COBW3-SEARCH-FLAG-EXIST AND
001340        COBW3-GET-LENGTH NOT = 0 AND
001350        COBW3-GET-DATA (1 : COBW3-GET-LENGTH) NOT = SPACE
001360     THEN
001370        MOVE COBW3-GET-DATA (1 : COBW3-GET-LENGTH) TO FIRST-NAME
001380        SET  CN-FIRSTNAME-SET TO TRUE
001390     ELSE
001400        SET  CN-FIRSTNAME-NO-SET TO TRUE
001410     END-IF
001420
001430     MOVE C-LASTNAME  TO COBW3-SEARCH-DATA
001440     MOVE 1           TO COBW3-NUMBER
001450     CALL "COBW3_NAME" USING COBW3
001460     IF COBW3-SEARCH-FLAG-EXIST AND
001470        COBW3-GET-LENGTH NOT = 0 AND
001480        COBW3-GET-DATA (1 : COBW3-GET-LENGTH) NOT = SPACE
001490     THEN
001500        MOVE COBW3-GET-DATA (1 : COBW3-GET-LENGTH) TO LAST-NAME
001510        SET  CN-LASTNAME-SET TO TRUE
001520     ELSE
001530        SET  CN-LASTNAME-NO-SET TO TRUE
001540     END-IF
001550
001560     MOVE C-DATE-OF-BIRTH  TO COBW3-SEARCH-DATA
001570     MOVE 1                TO COBW3-NUMBER
001580     CALL "COBW3_NAME" USING COBW3
001590     IF COBW3-SEARCH-FLAG-EXIST AND
001600        COBW3-GET-LENGTH NOT = 0 AND
001610        COBW3-GET-DATA (1 : COBW3-GET-LENGTH) NOT = SPACE
001620     THEN
001630        MOVE COBW3-GET-DATA (1 : COBW3-GET-LENGTH) TO DATE-OF-BIRTH
001640        SET  CN-DOB-SET TO TRUE
001650     ELSE
001660        SET  CN-DOB-NO-SET TO TRUE
001670     END-IF
001680
001690     IF CN-FIRSTNAME-NO-SET AND
001700        CN-LASTNAME-NO-SET AND
001710        CN-DOB-NO-SET
001720        MOVE SPACE TO HTML-MESSAGE
001730        GO TO CHECK-INPUT-EXIT
001740     END-IF
001750
001760     IF CN-FIRSTNAME-NO-SET
001770        MOVE SC-INPUT-FIRSTNAME TO HTML-MESSAGE
001780        GO TO CHECK-INPUT-EXIT
001790     END-IF
001800
001810     IF CN-LASTNAME-NO-SET
001820        MOVE SC-INPUT-LASTNAME TO HTML-MESSAGE
001830        GO TO CHECK-INPUT-EXIT
001840     END-IF
```

```
001850
001860     IF CN-DOB-NO-SET
001870        MOVE SC-INPUT-DOB TO HTML-MESSAGE
001880        GO TO CHECK-INPUT-EXIT
001890     END-IF
001900
001910     IF DOB-DAY OF DATE-OF-BIRTH IS NOT NUMERIC
001920        MOVE SC-INVALID-DAY TO HTML-MESSAGE
001930        SET  CN-INVALID-DAY TO TRUE
001940        GO TO CHECK-INPUT-EXIT
001950     END-IF
001960
001970     IF DOB-MONTH OF DATE-OF-BIRTH IS NOT NUMERIC
001980        MOVE SC-INVALID-MONTH TO HTML-MESSAGE
001990        SET  CN-INVALID-MONTH TO TRUE
002000        GO TO CHECK-INPUT-EXIT
002010     END-IF
002020
002030     IF DOB-YEAR OF DATE-OF-BIRTH IS NOT NUMERIC
002040        MOVE SC-INVALID-YEAR TO HTML-MESSAGE
002050        SET  CN-INVALID-YEAR TO TRUE
002060        GO TO CHECK-INPUT-EXIT
002070     ELSE
002080        SET  CN-VALID-YEAR   TO TRUE
002090     END-IF
002100
002110     IF DOB-MONTH OF DATE-OF-BIRTH = 2
002120        IF (DOB-YEAR OF DATE-OF-BIRTH / 4 * 4 NOT = DOB-YEAR OF DATE-
OF-BIRTH) OR
002130           ((DOB-YEAR OF DATE-OF-BIRTH / 100  * 100 = DOB-YEAR OF DATE-
OF-BIRTH) AND
002140           (DOB-YEAR OF DATE-OF-BIRTH / 400  * 400 NOT = DOB-YEAR OF
DATE-OF-BIRTH))
002150           MOVE NUMBER-OF-DAYS (DOB-MONTH OF DATE-OF-BIRTH) TO MAX-
DAYS-OF-THE-MONTH
002160        ELSE
002170           COMPUTE MAX-DAYS-OF-THE-MONTH = NUMBER-OF-DAYS (DOB-MONTH OF
DATE-OF-BIRTH) + 1
002180        END-IF
002190     ELSE
002200        MOVE NUMBER-OF-DAYS (DOB-MONTH OF DATE-OF-BIRTH) TO MAX-DAYS-
OF-THE-MONTH
002210     END-IF
002220
002230     IF DOB-DAY OF DATE-OF-BIRTH < 1 OR
002240        DOB-DAY OF DATE-OF-BIRTH > MAX-DAYS-OF-THE-MONTH
002250        MOVE SC-INVALID-DAY TO HTML-MESSAGE
002260        SET  CN-INVALID-DAY TO TRUE
002270        GO TO CHECK-INPUT-EXIT
002280     ELSE
002290        SET  CN-VALID-DAY   TO TRUE
002300     END-IF
002310
002320     IF DOB-MONTH OF DATE-OF-BIRTH < 1 OR
002330        DOB-MONTH OF DATE-OF-BIRTH > 12
002340        MOVE SC-INVALID-MONTH TO HTML-MESSAGE
002350        SET  CN-INVALID-MONTH TO TRUE
```

```
002360     ELSE
002370        SET  CN-VALID-MONTH   TO TRUE
002380     END-IF.
002390
002400 CHECK-INPUT-EXIT.
002410     EXIT.
002420*-------------------------------------------------------------
002430 EXECUTE-PLSQL SECTION.
002440
002450     MOVE DOB-DAY  OF DATE-OF-BIRTH                TO DOB-DAY   OF
DATE-OF-BIRTH-FORMAT
002460     MOVE MONTH-NAME (DOB-MONTH OF DATE-OF-BIRTH)  TO DOB-MONTH OF
DATE-OF-BIRTH-FORMAT
002470     MOVE DOB-YEAR OF DATE-OF-BIRTH                TO DOB-YEAR  OF
DATE-OF-BIRTH-FORMAT
002480
002490     MOVE SPACES TO COBW3-PUT-HEAD
002500     STRING PLSQL-ADDRESS-STRING DELIMITED BY SIZE
002510            FIRST-NAME           DELIMITED BY SPACE
002520            LAST-NAME-STRING     DELIMITED BY SIZE
002530            LAST-NAME            DELIMITED BY SPACE
002540            DATE-OF-BIRTH-STRING DELIMITED BY SIZE
002550            DATE-OF-BIRTH-FORMAT DELIMITED BY SPACE
002560            INTO COBW3-PUT-HEAD
002570     END-STRING
002580     SET  COBW3-CONTENT-TYPE-NON TO TRUE
002590     SET  COBW3-STATUS-CODE-NON  TO TRUE
002600     CALL "COBW3_PUT_HEAD"       USING COBW3.
002610
002620*-------------------------------------------------------------
002630 OUTPUT-HTML SECTION.
002640     MOVE SPACES TO COBW3-CNV-VALUE
002650     IF FIRST-NAME NOT = SPACE
002660        STRING  'VALUE="'         DELIMITED BY SIZE
002670                FIRST-NAME        DELIMITED BY SPACE
002680                '"'               DELIMITED BY SIZE
002690           INTO COBW3-CNV-VALUE
002700        END-STRING
002710     END-IF
002720     MOVE SC-CNV-FIRSTNAME TO COBW3-CNV-NAME
002730     CALL "COBW3_SET_CNV"  USING COBW3
002740
002750     MOVE SPACES TO COBW3-CNV-VALUE
002760     IF LAST-NAME NOT = SPACE
002770        STRING  'VALUE="'         DELIMITED BY SIZE
002780                LAST-NAME         DELIMITED BY SPACE
002790                '"'               DELIMITED BY SIZE
002800           INTO COBW3-CNV-VALUE
002810        END-STRING
002820     END-IF
002830     MOVE SC-CNV-LASTNAME TO COBW3-CNV-NAME
002840     CALL "COBW3_SET_CNV"  USING COBW3
002850
002860     MOVE SPACES TO COBW3-CNV-VALU
002870     IF DATE-OF-BIRTH NOT = SPACE
002880        STRING  'VALUE="'         DELIMITED BY SIZE
002890                DATE-OF-BIRTH     DELIMITED BY SPACE
```

```
002900                                                  DELIMITED BY SIZE
002910           INTO COBW3-CNV-VALUE
002920       END-STRING
002930     END-IF
002940     MOVE SC-CNV-BIRTHDAY  TO COBW3-CNV-NAME
002950     CALL "COBW3_SET_CNV"  USING COBW3
002960
002970     MOVE HTML-MESSAGE     TO COBW3-CNV-VALUE
002980     MOVE SC-CNV-MESSAGE   TO COBW3-CNV-NAME
002990     CALL "COBW3_SET_CNV"  USING COBW3
003000
003010     MOVE SC-HTML-FILE-NAME TO COBW3-HTML-FILENAME
003020     CALL "COBW3_PUT_HTML"  USING COBW3.
003030
003040*-------------------------------------------------------------
003050 TERMINATE-ROUTINE SECTION.
003060     CALL "COBW3_FREE" USING COBW3
```

The design of this program is quite like the CALCULATOR program; however, there is considerably more error processing. Lines 1040 to 1080 document the processing in the BIRTHDAY program. The following is an explanation of the processing:

1. Initialize the COBW3 environment. This is the same as the other two programs.
2. Check the input. This is similar to the CALCULATOR program except that there are date routines being checked.
3. If there are errors, PL/SQL is not called. Instead, COBW3_CNV subroutines are called to edit the BIRTHDAY.HTM file (line 200). A listing of this file is shown below. You can see from the HTML file that error messages are packed into the CNV-MESSAGE parameter. This item will blink an error message.

```
<HTML>
<HEAD>
<TITLE>Input Your Birthday</TITLE>
</HEAD>

<BODY BGCOLOR=gray TEXT="#FFF000">
<FORM METHOD="POST"
ACTION="http://spice:7777/zip-bin/cobol/birthday.DLL/BIRTHDAY">
<P><CENTER><H1>Input Your Birthday</H1></CENTER></P>

<P><STRONG>First Name : </STRONG>
<INPUT TYPE="text" NAME="FIRSTNAME" SIZE="40"
//COBOL//CNV-FIRSTNAME//COBOL//><BR>
```

```
<STRONG>Last Name : </STRONG>
<INPUT TYPE="text" NAME="LASTNAME" SIZE="40"
//COBOL//CNV-LASTNAME//COBOL//><BR>

<STRONG>Your Birthday (ddmmyyyy) : </STRONG>
<INPUT TYPE="text" NAME="BIRTHDAY" SIZE="8"
//COBOL//CNV-BIRTHDAY//COBOL//></P>

<P><CENTER><INPUT TYPE="SUBMIT" VALUE="Send">
<INPUT TYPE="reset" VALUE="Clear"></CENTER></P>

<P><BLINK>//COBOL//CNV-MESSAGE//COBOL//</P>

</FORM>
</BODY>
</HTML>
```

4. If the input has a valid date, and the first and last name has been input, then the PL/SQL procedure INSERTBIRTHDAY is called. This procedure will insert the new record into a table in the Oracle database, and then call another PL/SQL procedure called GETBIRTHDAYS. GETBIRTHDAYS will display an HTML page of all the birthdays in the database table. The PL/SQL that is included with the BIRTHDAY program is discussed in Appendix C, “Using the PL/SQL Web Toolkit.”

   When a user enters the data as shown in the previous figure, and clicks on the Send button, the following Web page appears. This page was created entirely by PL/SQL, using the PL/SQL Web Toolkit.

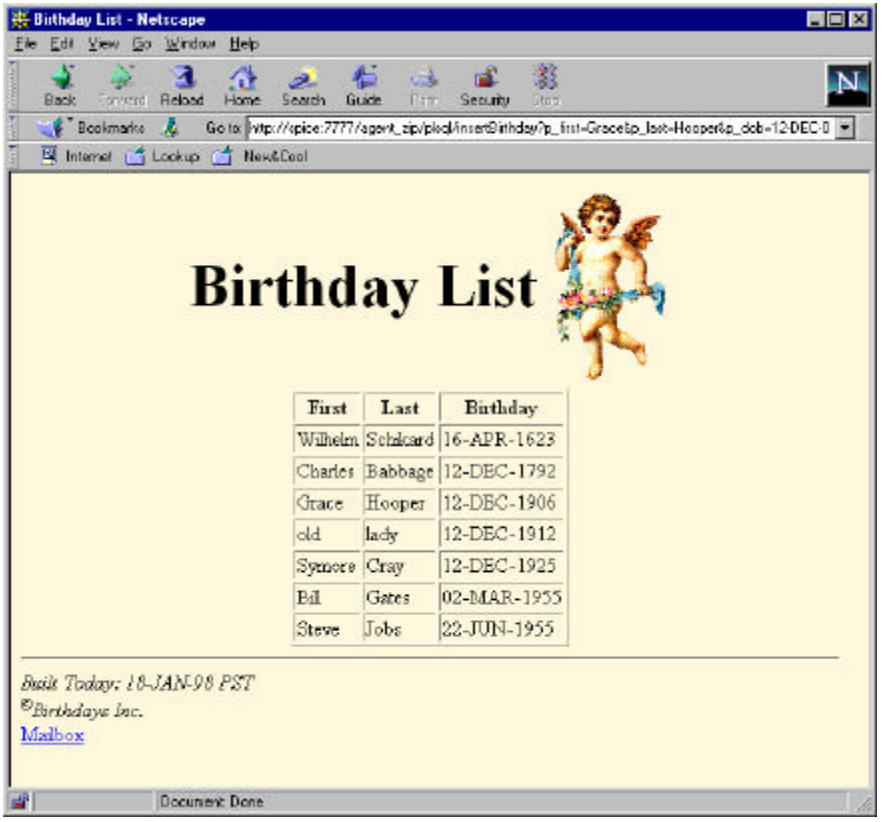

Figure 3.5. The web page created by the PL/SQL procedure GETBIRTHDAY

You will note from the above code that the key to redirecting the processing from the COBOL Cartridge to the PL/SQL Cartridge was the use of the COBW3-PUT-HEAD subroutine. This allowed the creation of the HTTP "Location" command that set up the re-direction.

Chapter 4. Debugging COBOL Cartridge Applications

This chapter explains different methods of application development. Depending on your level of comfort and the complexity of your COBOL Cartridge application you may want to employ one more of these techniques for the purposes of understanding the exchange of data, HTTP messages and building HTML documents.

The following topics will be discussed:

- Using DMODE for debugging
- Using COBW3-STATUS for error processing and debugging
- Using the Fujitsu COBOL Debugger for run-time debugging on the server
- Using Display for debugging and demonstration

Using COBW3-DMODE

Setting the COBW3-DMODE parameter allows you to examine the output of COBW3_PUT_TEXT calls. This is useful for looking at HTML output. The following lines are from the BIRTHDAY program described in the previous chapter.

```
001190*-------------------------------------------------------------
001200 INITIALZE-ROUTINE SECTION.
001210     MOVE LOW-VALUE TO COBW3
001220     MOVE WRBCTX    TO COBW3-CONTEXT
001221*---- Turn on Debug Mode ---
001222     SET COBW3-DMODE-DBG TO TRUE.
1230        CALL "COBW3_INIT" USING COBW3
```

Line 001222 sets the debug mode on. You will need to rebuild the .DLL file to make the debugging line effective. You then transfer the .DLL to the Object Path directory that was configured in the COBOL Cartridge installation. The following figure shows the resulting display in the browser when this debug option is turned on.

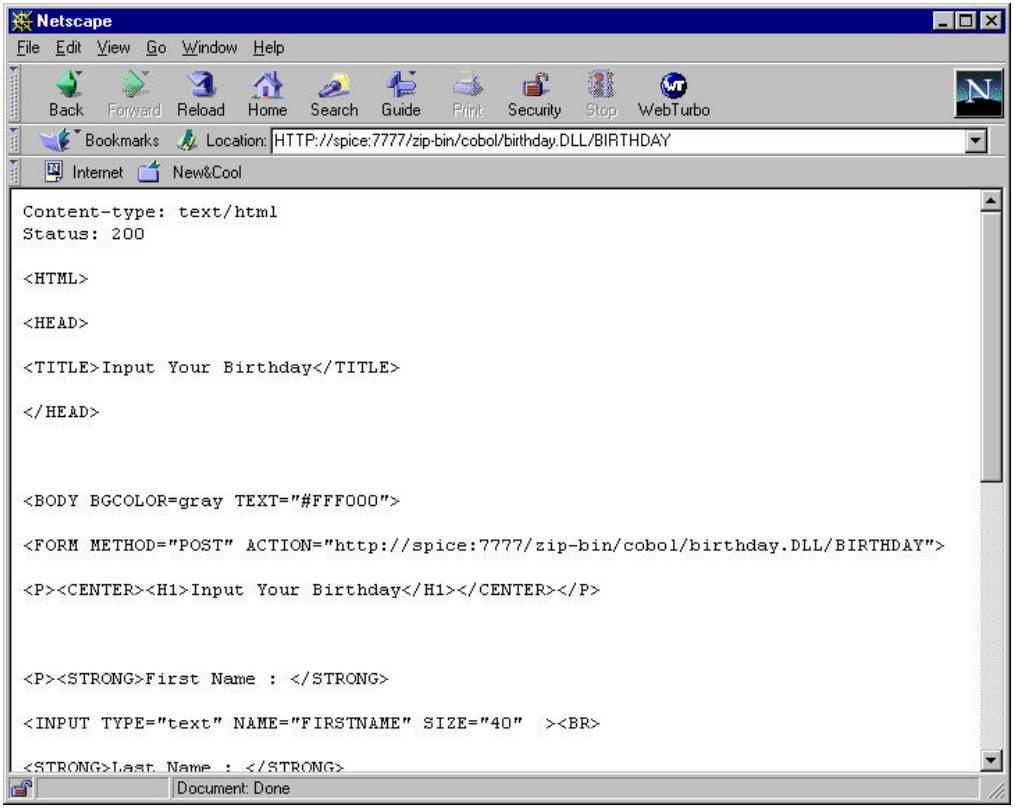

Figure 4.1. Page created using COBW3-DMODE option set to TRUE

You will note that the HTTP Content-type header, and the HTTP Status is displayed with the HTML source in the above figure.

Using COBW3-STATUS

The COBW3-STATUS variable is set with a return status each time a COBW3 API routine is called. A return status of zero indicates a successful completion. A non-zero status indicates an error. For simplicity, none of the examples in the previous chapter exploit the advantages of using the return status from the

COBOL Cartridge. The following table lists the possible error codes that the COBOL Cartridge will return. “U” error codes are for unrecoverable errors, “E“ error codes allow processing to continue.

Table 4.1 Processing errors from COBW3 subroutines

Code	Error Number	Message Text
U	0100	Processing halted because the COB-WRB-CONTEXT was not set.
U	1100	Processing halted because the COBW3 data space was corrupted. Probable cause is the mis-use of COBW3 subroutines.
U	2000	Processing halted because of illegal declaration of Content-type.
E	2050	A negative retrieval character string length was specified. Processing continues with length set to zero.
E	2051	The specified value for character string length exceeds the length of the character string. Processing continues with length set to the maximum length of the string.
E	2052	NAME to be retrieved was not found.
E	2053	VALUE to be retrieved was not found.
E	3001	COB3_INIT was called without first releasing the previously called COBW3 space. Call COB3_FREE before issuing the second COB3_INIT call.
U	3002	Could not access the subroutine workspace.
E	3003	Could not access the POST or GET method data.
E	3050	COBW3_IN_CODE_SET was not set. No conversion done.
E	3051	Illegal value for COBW3_IN_CODE_SET. No conversion done.
E	3052	Exceeded maximum size for COBW3_IN_CODE_SET. No more than 10 pairs allowed.
E	3060	COBW3_OUT_CODE_SET was empty.
E	3061	Exceeded maximum size for COBW3_OUT_CODE_SET. No more than 10 pairs allowed.
E	3700	Unrecognized character code encountered.
E	3702	Illegal specification for COBW3_IN_CODE_SET.
E	3900	Processing error in the conversion of Japanese character code.
E	4006	A negative value or zero was specified for COB3- NUMBER. Processing continues with COB3-NUMBER set to 1.
E	4008	The length of VALUE exceeds the maximum allowed. VALUE set to the maximum.
E	4400	The conversion name was not specified.
E	4401	A negative value was specified for character string length. Processing continues with length set to zero.

E	4402	The specified value for character string length exceeds the length of the character string. Processing continues with length set to the maximum length of the string.
E	4006	COBW3_NUMBER specified as negative or zero. Processing continues with COBW3_NUMBER set to 1.
E	4008	VALUE exceeds string length. Line truncated at maximum length.
E	4009	Passed data with GET/POST was empty.
E	4400	No conversion name specified.
E	4401	Negative value specified for string length of conversion name. Processing continues with string length set to zero.
E	4402	The string length value exceeds the maximum. Value reset to the maximum.
E	4450	Deletion failed because all necessary conversion parameters have not been initialized.
E	4470	Change failed because all necessary conversion parameters have not been initialized.
E	5000	The system command execution failed.
E	6100	An I/O error occurred from the HTML OPEN file.
E	6105	Conversion name was not set. Use COBW3_SET_CNV to set HTML file name.
E	6106	File write failed. Check the file name length, or the file name.
E	6107	File write failed because proper conversion name in the HTML document was not found.
E	6300	No conversion name was found.
E	6301	A negative value was specified for conversion name character string length. Processing continues with length set to zero.
E	6302	The specified value for conversion character string length exceeds the length of the character string. Processing continues with length set to the maximum length of the string.
E	6303	No conversion name was specified.
E	6304	A negative value was specified for character string length. Processing continues with length set to zero.
E	6305	The specified value for string length exceeds the length of the character string. Processing continues with length set to the maximum length of the string.
E	6310	The conversion name specified already exists.
E	6320	Insufficient work space. Call COBW3-CNV-INIT and initialize conversion information.
E	6502	Unexpected error occurred while writing to standard output.
E	6503	Negative value specified for COBW3-PUT-STRING-LENGTH. Processing continued with length set to zero.
E	6504	The specified value for the character string length exceeds the length

		of the character string. Processing continues with length set to the maximum length of the string.
E	6700	Invalid operation. Either COBW3_PUT_HTML or COBW3_PUT_TEXT has been previously called without resetting the header.
E	6704	A negative value has been specified for COBW3-PUT-HEAD-LENGTH. The processing continues with length set to zero.
E	6706	The value of COBW3-PUT-HEAD-LENGTH exceeds the maximum. Processing continues with COBW3-PUT-HEAD-LENGTH set to maximum.
E	6708	The object output header was not specified.
E	6710	Invalid value specified for COBW3-CONTENT-TYPE.
E	6714	Invalid value specified for COBW3-STATUS-CODE.
E	6730	Illegal value specified for Content-type.
E	6734	Error in the declaration of Status code.
E	6742	Output header failure.
E	6746	Incomplete output of the header.
E	7010	The user name could not be found.
E	7011	The specified value for the user name string length exceeds the length of the character string. Processing continues with length set to the maximum length of the string.
E	7020	The password was not found.
E	7021	The specified value for the password string length exceeds the length of the character string. Processing continues with length set to the maximum length of the string.
E	7030	Internet Protocol address was not found.
E	7031	The specified value for the IP address string length exceeds the length of the character string. Processing continues with length set to the maximum length of the string.

You may use COBW3-STATUS for two purposes: debugging, and error processing. The following two topics explain how to debug and process errors using COBW3-STATUS.

Using COBW3-STATUS for Debugging

The following is a partial listing of the CALCULATOR program that was discussed in the previous chapter. It has been modified to include debugging statements that use the COBW3-STATUS

variable. There are a variety of ways to employ COBW3-STATUS for debugging. The following code example demonstrates the mechanics of how it is done. You can customize the example to suite your needs.

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. CALCULATOR.
.
.
.
000670 01  STATUS-LINE.
000680     05  HTML-BREAK              PIC X(4)   VALUE "<BR>".
000690     05  STATUS-VALUE            PIC 9(4)   VALUE 8888.
000700     05  STATUS-PRETTY           PIC X(3)   VALUE " = ".
000710     05  STATUS-LABEL            PIC X(20)  VALUE SPACES.
.
.
.
000830*-----------------------------------------------------------------
000840 PROCEDURE DIVISION USING WRBCTX.
000850     PERFORM INITIALZE-ROUTINE
000860     PERFORM DECIMAL-CHECK
000870     IF Z-DEC-CORRECT
000880        PERFORM CONVERT-DATA
000890     END-IF
000900     PERFORM OUTPUT-HTML
000910     PERFORM TERMINATE-ROUTINE.
000920     EXIT PROGRAM.
000930*-----------------------------------------------------------------
000940 INITIALZE-ROUTINE.
000950     MOVE LOW-VALUE  TO COBW3
000960     MOVE WRBCTX     TO COBW3-CONTEXT
000970     CALL "COBW3_INIT" USING COBW3
000980     MOVE "COBW3_INIT" TO STATUS-LABEL.
000990     PERFORM CHECK-STATUS.
001000     SET  Z-DEC-NODATA TO TRUE
001010     MOVE SPACES TO HTML-MESSAGE.
001020
001030*-----------------------------------------------------------------
.
.
.
001480*-----------------------------------------------------------------
001490 OUTPUT-HTML.
001500
.
.
.
001600     CALL "COBW3_SET_CNV"  USING COBW3
001610     MOVE "COBW3_SET_CNV 1" TO STATUS-LABEL.
001620     PERFORM CHECK-STATUS.
001640     MOVE SPACES TO COBW3-CNV-VALUE
001650     IF Z-DEC-CORRECT
```

```
001660         STRING  'VALUE="'              DELIMITED BY SIZE
001670                 HEX-ANSWER             DELIMITED BY SPACE
001680                 '"'                    DELIMITED BY SIZE
001690            INTO COBW3-CNV-VALUE
001700         END-STRING
001710     END-IF
001720     MOVE SC-CNV-HEXNUMBER TO COBW3-CNV-NAME
001730     CALL "COBW3_SET_CNV"  USING COBW3
001740     MOVE "COBW3_SET_CNV 2" TO STATUS-LABEL.
001750     PERFORM CHECK-STATUS.
001760
.
.
.
002090     MOVE SC-HTML-FILE-NAME TO COBW3-HTML-FILENAME
002100     CALL "COBW3_PUT_HTML"  USING COBW3.
002110     MOVE "COBW3_PUT_HTML" TO STATUS-LABEL.
002120     PERFORM CHECK-STATUS.
002130
002140*-----------------------------------------------------------------
002150 CHECK-STATUS.
002160     MOVE COBW3-STATUS TO STATUS-VALUE.
002170     MOVE STATUS-LINE TO COBW3-PUT-STRING.
002180     MOVE FUNCTION LENG(STATUS-LINE) TO COBW3-PUT-STRING-LENGTH.
002190     CALL "COBW3_PUT_TEXT" USING COBW3.
002200
002210*-----------------------------------------------------------------
002220 TERMINATE-ROUTINE.
002230     CALL "COBW3_FREE" USING COBW3
```

There are three parts to this process: building a data structure, building a 'perform' paragraph, and inserting the debug statements.

Building a Data Structure

Lines 670 – 710 build a data structure that will contain the status messages. This one allows for the error status, the name of the part of the program that you want to use to identify that location with, and an HTML line break so that the each status messages appears on a single line.

Building a 'Perform' Paragraph

Lines 2150 to 2190 account for the 'perform' paragraph. The output string is built, and the following line writes the debug information to the browser window.

```
CALL "COBW3_PUT_TEXT" USING COBW3
```

Exercise caution in building elaborate status messages. You need to keep in mind that the call that prints the status message creates a status message itself, overwriting the previous status data.

Inserting Debug Statements

Throughout the program, couplets of code first set the debug location, and then call the 'perform' paragraph that gets the status and constructs the debug line. Lines 2110 and 2120 are examples of how you set a tag to identify your position in the program, and how the section of code is called to check the status and output the debug information.

The following figure shows the output from this debugging session. You can see that all of the COBW3 subroutines returned successful status codes. You can also see that because we gave each value of STATUS-LABEL a different name, it is quite easy to see where the status message came from.

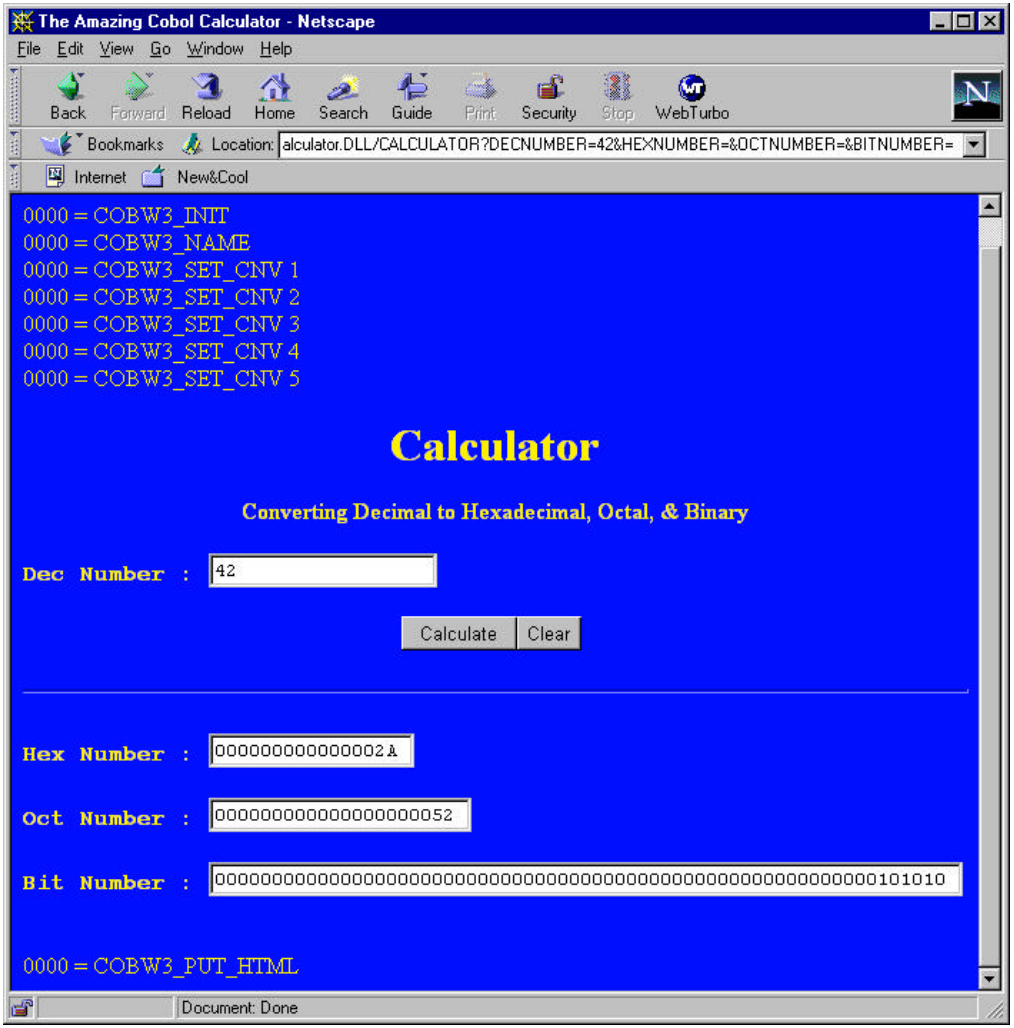

Figure 4.2. The web page created by the CALCULATOR program with debugging code

This technique of debugging will work fine for errors that are not unrecoverable (See Table 4.1). However, because the .DLL runs as a background process it will likely hang the WRB if a fatal error is encountered. (The remedy for a hung WRB is to shut down the listener, and the WRB, then bring the WRB back up,

followed by the listener.) To root out program errors that are of the 'unrecoverable' nature, you should use the COBOL Debugger. The techniques for doing this are discussed in the following sections.

Using COBW3-STATUS for Error Processing

You should always include error processing in your applications. When using the COBOL Cartridge, an indiscretion in your program that is not gracefully handled can have sometimes embarrassing consequences. It is not uncommon for a poorly written application to lock up the COBOL Cartridge, or the WRB, so that the listeners and the WRB must be bounced before the site can be used again. The following lines of code introduce an error into the CALCULATOR program.

```
002030     MOVE HTML-MESSAGE     TO COBW3-CNV-VALUE
002040*     MOVE SC-CNV-MESSAGE   TO COBW3-CNV-NAME
002044     MOVE SC-WRB-INVALID-INPUT TO COBW3-CNV-NAME
```

You will note that line 2040 is the incorrect line. In this example, the user improperly set the COBW3-CNV-NAME parameter.

If the above 'perform' paragraph is slightly modified, it will service any errors that happen. The following section of code shows this modification.

```
002160 CHECK-STATUS.
002170   IF COBW3-STATUS  > 0 AND COBW3-STATUS NOT = 4009
002180      MOVE COBW3-STATUS TO STATUS-VALUE
002190      MOVE STATUS-LINE TO COBW3-PUT-STRING
002200      MOVE FUNCTION LENG(STATUS-LINE) TO COBW3-PUT-STRING-LENGTH
002210      CALL "COBW3_PUT_TEXT" USING COBW3
002220      PERFORM TERMINATE-ROUTINE
002230   END-IF.
```

This section of code will terminate if there is any return status that is greater than zero, and print out a message. Notice that we allow error 4009. This is because the name/value pairs returned by our HTML form will be empty. (Refer to error 4009 in Table 4.1.)

The status data structure was also changed to inform the user that the session has been terminated. The following lines of code show the change from the above debugging example.

```
000670 01  STATUS-LINE.
000680     05  HTML-BREAK          PIC X(4)   VALUE "<BR>".
000690     05  STATUS-VALUE        PIC 9(4)   VALUE 8888.
000700     05  STATUS-PRETTY       PIC X(15)  VALUE " Terminated at ".
000710     05  STATUS-LABEL        PIC X(20)  VALUE SPACES.
```

The figure below shows the output from running the CALCULATOR program after it has been modified to crash.

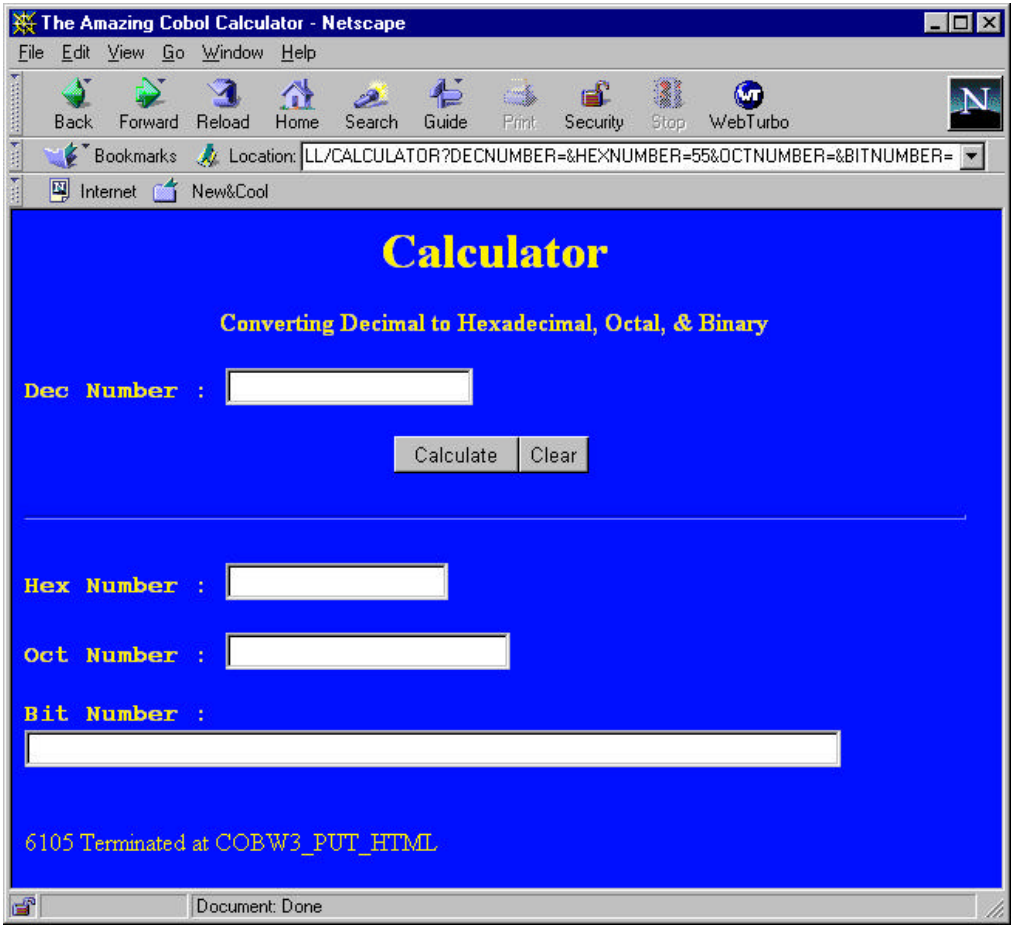

Figure 4.3. The CALCULATOR program with error processing

You can see from the above figure that the program terminated with a 6105 error. From the above table of errors, you can easily discover the error. However, without the error processing, it would not be obvious what the problem was, since the program would continue to compile properly.

Using the COBOL Run-time Debugger

Using the COBOL Run-time Debugger is the ultimate way to examine the behavior of your program. It offers all the features that one expects to find in a first class debugger: break points, examination of variables, step-into, step-over, and other features; all in an intuitive window. The disadvantages of the Run-time Debugger is that you will be required to do your debugging on the computer that runs the Web Application Server to do run-time debugging. You will also want to do your testing on a system that is dedicated to that task (at least for the time you reserve to do testing) because the configuration of the Web Server will effect the behavior of other programs.

Configuring Your Environment for Run-time Debugging

Using the Run-time Debugger requires the following steps be performed:

1. Configure the Oracle Web Server to allow for desktop interaction so that you can interact with your program outside of the browser window.
2. Compile and link your .DLL with debug options set.
3. Move the .DLL and associated files to the directory that you configured when you set the Object Path parameter of the COBOL Cartridge Configuration page. (See Chapter 1 for more information on the Object Path.)
4. Set the Run-time Environment Setup to: @CBR_ATTACH_TOOL=TEST

The following section explains how the configuration is done.

Configuring the Oracle Web Application Server

You need to configure the Oracle Web Application Server so that you will be able to interact with the WINSVD debugging program that pops up with the program you want to debug. The consequences of not doing this step is for the WRB to hang, and the browser screen to go blank as an 'invisible' process in the background awaits your interaction.

Set the Web Application Server for desktop interaction as follows:

1. Open the Services windows. Select Start, Settings, Control Panel, Services. Scroll down the list of services until you see the **OracleWRBPrimaryService** as shown in the following figure.

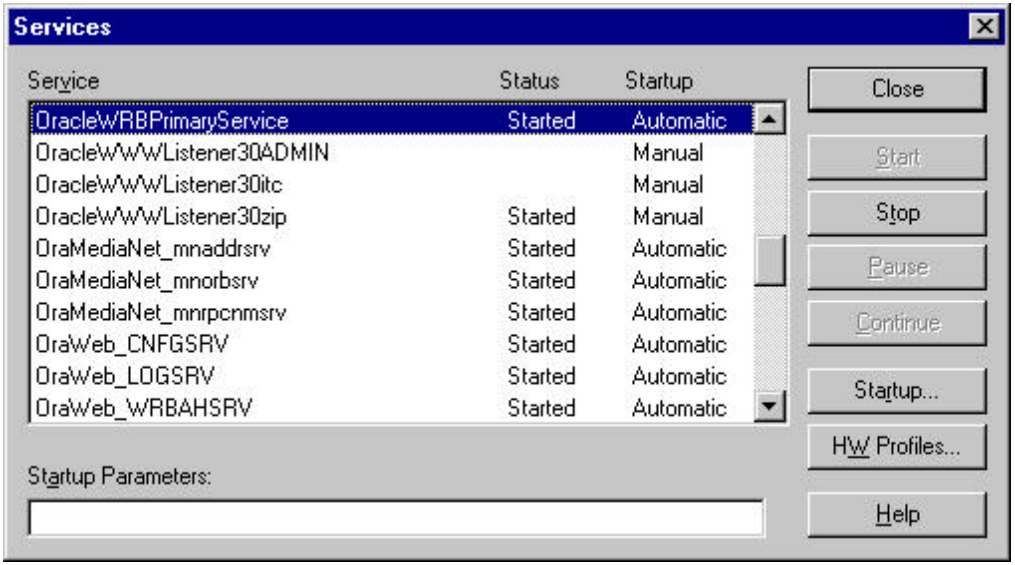

Figure 4.4. The Services dialog box

2. Next, double-click the mouse on the **OracleWRBPrimaryService** to display the Service (behavior)

dialog box. Select the check box titled “Allow Service to Interact with Desktop”. Your display should resemble the following figure:

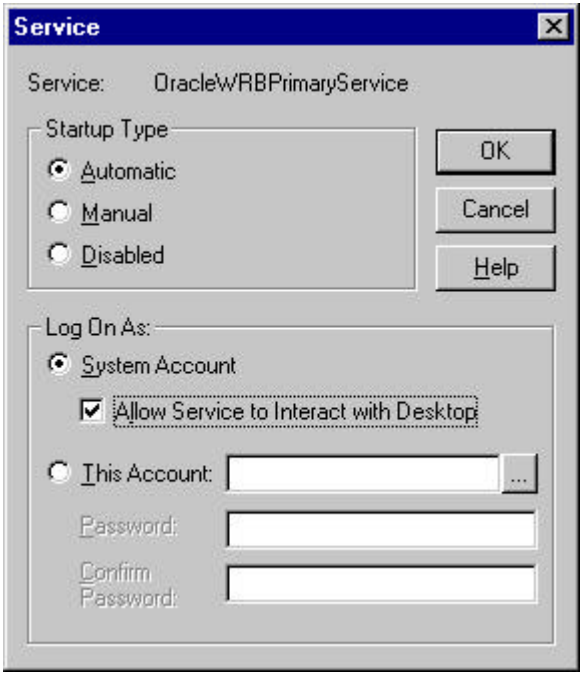

Figure 4.5. Setting the OracleWRBPrimaryService for desktop interaction

You will need to repeat this process for each of the Oracle Web Application server components. These include:

OraMediaNet_mnaddrsrv

OraMediaNet_mnorbsrv

OraMediaNet_mnrpcnmsrv

OraWeb_CNFGSRV

OraWeb_LOGSRV

OraWeb_WRBAHSRV

OraWeb_WRBASRV

OraWeb_WRBFAC

OraWeb_WRBROKER

OraWeb_WRBVPM

You will also want to configure your listener for desktop interaction.

3. The last step is to shut down the listeners, the Web Application Server, and then start them up again. This will initialize the changes that you made in the previous two steps.

Building the Application for Run-time Debugging

Building a project for run-time debugging is easy and convenient when using the Fujitsu COBOL Project Manager. The example below explains how to create a project file for building the CALCULATOR program for debugging. This example will take you through the basic steps of building a project for debugging. For a detailed explanation of the COBOL Project Manager, refer to the "Fujitsu COBOL User's Guide for Windows. "

1. Initialize the Fujitsu COBOL 4.0 Project Manager. Select Project Open from the File menu. The Open dialog box will appear for you to select which file to open. Type in the name of the project file you want to create. This example uses the file name CALCULATOR_DEBUG.

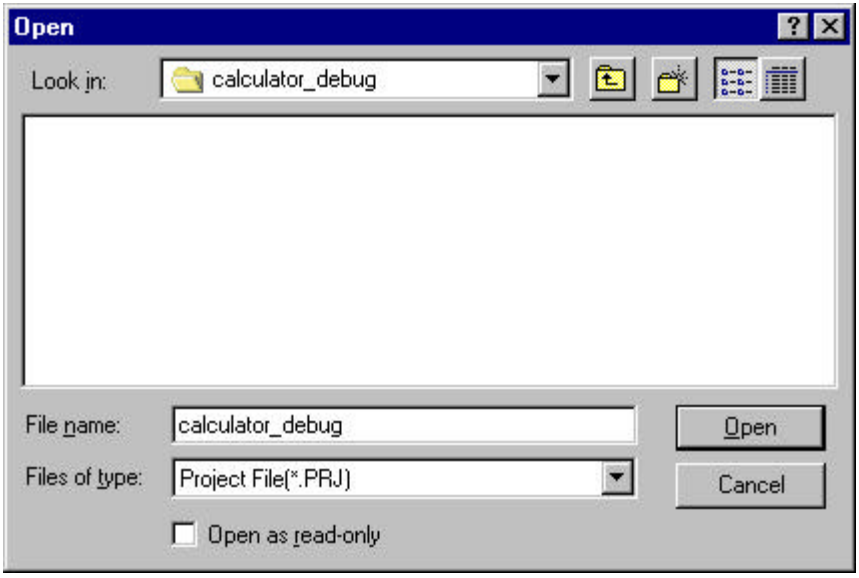

Figure 4.6. Creating a new project file

A window will appear to tell you that the file does not exist, and ask if you want to create one. Select “Yes”. The project folder will now appear in the Project Manager window.

2. Add the target .DLL file name to be built by right clicking on the project folder to display a pop-up menu, selecting Add, and entering the name CALCULATOR.DLL. The following figure shows the resulting Project Manager window.

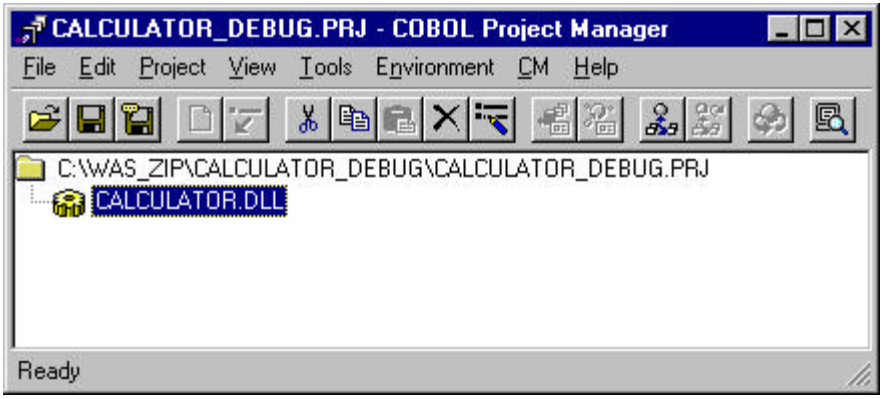

Figure 4.7. The Project Manager after inserting the new target .DLL name

3. Add the COBOL source file by right clicking on CALCULATOR.DLL, and selecting New Folder – COBOL Source File from the pop-up menus. A file folder will appear titled Cobol Source File. Right-click on that folder, and select Add File from the pop-up menu. A window will appear from which you may browse your way to the CALCULATOR source if it isn't already displayed in the window. The following figure demonstrates the results from selecting the CALCULATOR source file.

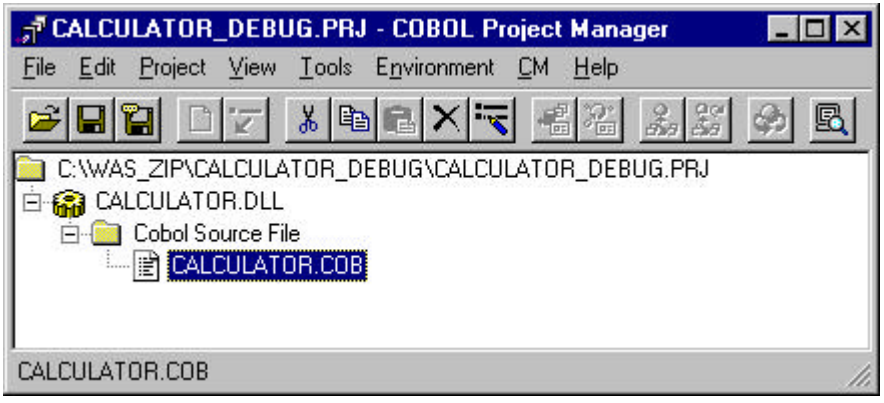

Figure 4.8. The Project Manager with the source file added

4. Next, add the library file by right clicking the CALCULATOR.DLL gear icon and selecting New Folder – Library File from the pop-up menus. A folder will appear called Library File. Next, right-click the mouse on Library File, and select Add File from the pop-up menu. A window will appear that will allow you to select or navigate to the required library file: Cobwapi.lib. Selecting this file will produce the following display.

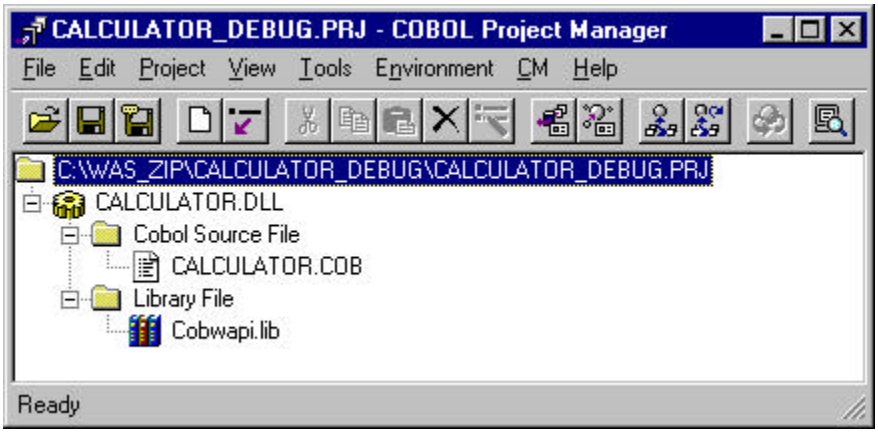

Figure 4.9. The Project Manager with all files needed for the build

5. The next step is to specify the compile options. Select Options, then Compiler Options from the Project menu. The Compiler Options dialog box will be displayed. Click on the Add button. The Add Compiler Options dialog box will be displayed. Scroll down the list, select TEST, and click on the Add button. The (Details) Compiler Option dialog box will be displayed. Select the Generate WINSVD DEGUG information radio button. The following figure shows the resulting dialog boxes from completing this step.

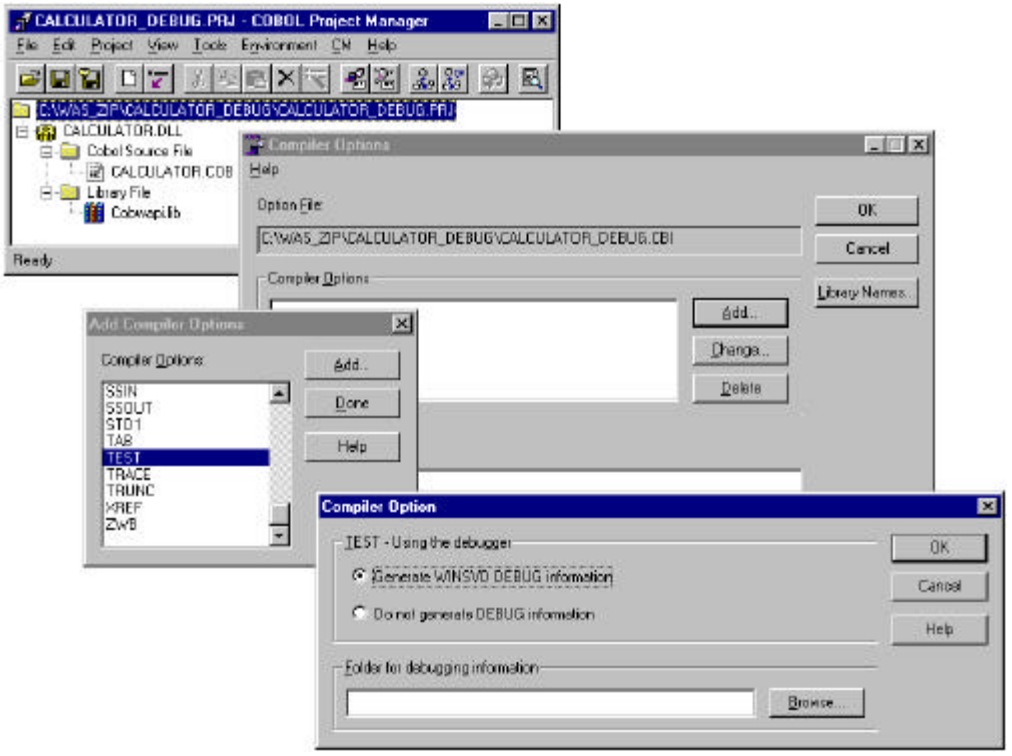

Figure 4.10. Selecting compile options

Finish this step by selecting the OK, Done, and OK buttons to back out of these dialog boxes. The resulting COBOL Project Manager window will appear unchanged.

6. The next step is to specify the link options. Select Options, then Link Options from the Project menu. The Linker Options dialog box will be displayed. Click on the Add button. A secondary Linker Options dialog box will be displayed. Type in the following line:
/DEBUG /DEBUGTYPE:COFF.

The following figure shows the results of this step.

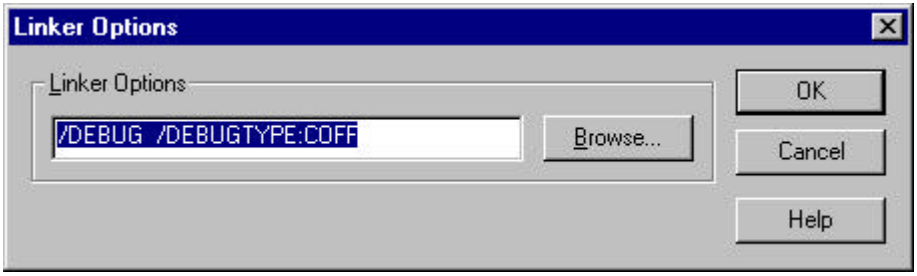

Figure 4.11. Setting linker options to debug

Click on the OK button to close this dialog box.

7. Build the application. Select the Rebuild icon. (If you are unsure which icon is the Rebuild icon, select the Project Manager window, and rest the cursor on each button, and read the yellow help tag that appears.) The following screen indicates a successful build.

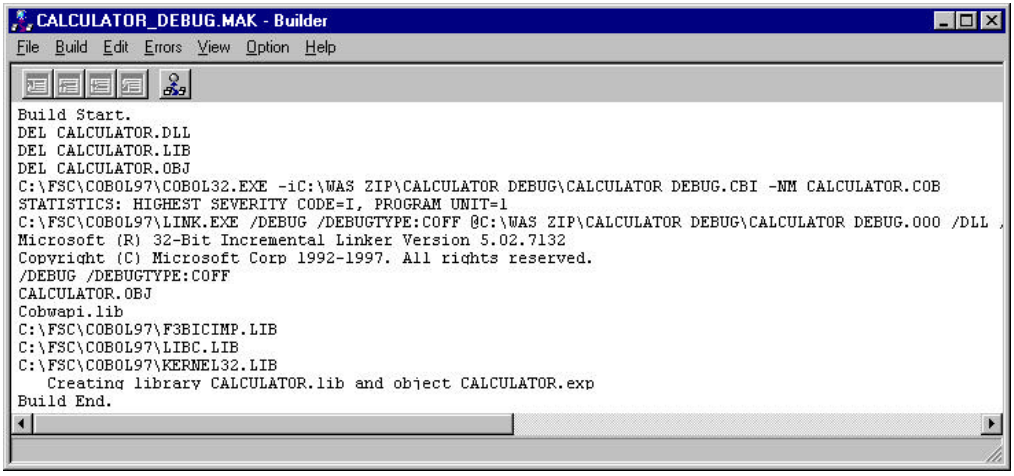

Figure 4.12. Build results with debug options set

8. The final step is to move the .DLL and associated files to the directory that you configured when you set the Object Path parameter on the COBOL Cartridge Configuration page. An

alternative is to select this directory as the directory where you build your project to be debugged.

Using the Run-time Debugger

After you have configured the Web Application Server to allow desktop interaction, and you have built the application using debug options, you are ready to exploit the powerful features of the Fujitsu COBOL Debugger.

For purposes of discussion, the CALCULATOR program has had a vital line of code commented out that will produce an unrecoverable error. This code snippet is shown below.

```
000930*-------------------------------------------------------------
000940 INITIALZE-ROUTINE.
000950     MOVE LOW-VALUE  TO COBW3
000960*     MOVE WRBCTX     TO COBW3-CONTEXT
000970     CALL "COBW3_INIT" USING COBW3
000980     MOVE "COBW3_INIT" TO STATUS-LABEL.
000990     PERFORM CHECK-STATUS.
001000     SET  Z-DEC-NODATA TO TRUE
001010     MOVE SPACES TO HTML-MESSAGE.
001020
001030*-------------------------------------------------------------
```

You will notice that line 960 has been commented out. The code should produce an unrecoverable error 100: Processing halted because the COB-WRB-CONTEXT was not set. Without specifically checking each status code for that particular error number, such an error would cause the Web Request Broker to freeze up and you would not have a clue from the browser window what happened.

The following steps run the Debugger for the CALCULATOR example that has been configured above, and demonstrate how to catch that error.

1. Start the CALCULATOR program from your browser running on the Web Application Server. When you select “The Amazing COBOL Calculator!”, rather than a new page

on the browser, you will see the Run-time Environment Setup window as shown in the following figure.

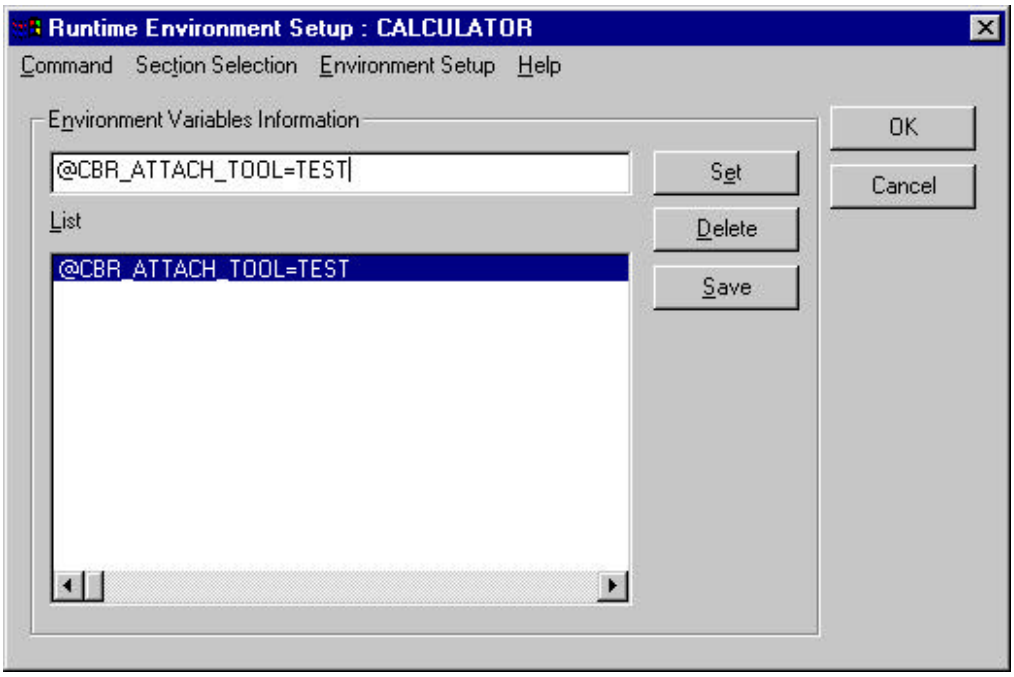

Figure 4.13. The Run-time Environment Setup window

You will need to type in the line **@CBR_ATTACH_TOOL=TEST**, then click on the Save button. Selecting the Save button will create a COBOL85.CBR file that will keep your run-time settings for the next time you run the program. The @CBR_ATTACH_TOOL=TEST specifies that the WINSVD debugger is to be run.

When you click on the OK button, the following dialog box will appear.

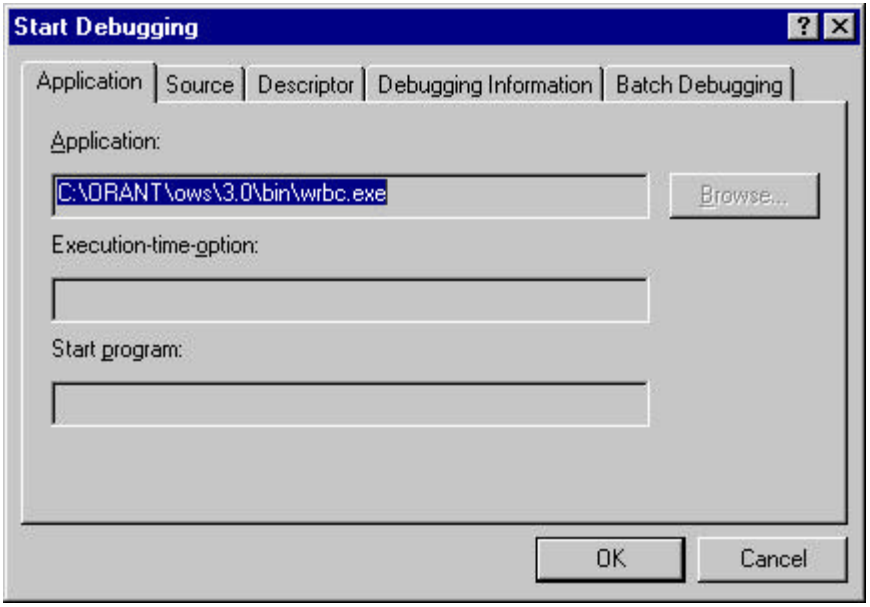

Figure 4.14. The Start Debugging dialog box

The Application field will already be filled in. This is the application that the COBOL .DLL links into.

2. Next, you will need to declare the path name to the COBOL source file, and the debugging information, by selecting those tabs, and either using the Browse button to navigate to that directory, or type in the path name. The following figure shows the Source tab filled with the correct information.

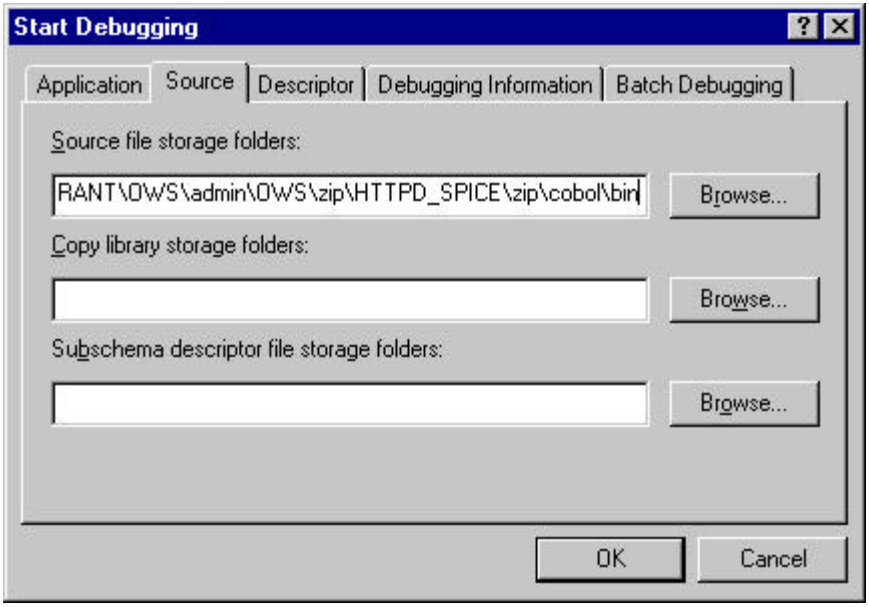

Figure 4.15. The Source tab in the Start Debugging dialog box

The path name is too long for the window in the figure. For this example, the path is:

```
C:\ORANT\OWS\admin\OWS\zip\HTTPD_SPICE\zip\cobol\bin
```

After the Source and Debugging Information tabs have been filled out, and the OK button is selected the main COBOL Debugger window appears. If you have run full screen debuggers before, this one will be very easy to learn. This example will present the steps to solve the problem of the induced error without a full explanation of all the features of the Debugger. For a complete explanation of the COBOL Debugger, refer to the “Fujitsu COBOL Debugging Guide”.

Note: As you fill out the Start Debugging Dialog-box, the Web Request Broker will be paused until the OK button is selected. This will lock out all other users that are trying to access the WRB. A timeout will occur after about 5 minutes if

you have not completed the filling out of the dialog-box. The debugging session will be canceled, and an informational pop-up window will explain that a time-out has occurred. You may select OK from the pop-up, and start your process from the browser if you like. The following figure shows the resulting pop-up window that will appear.

Figure 4.16. The Timeout Informational Pop-up

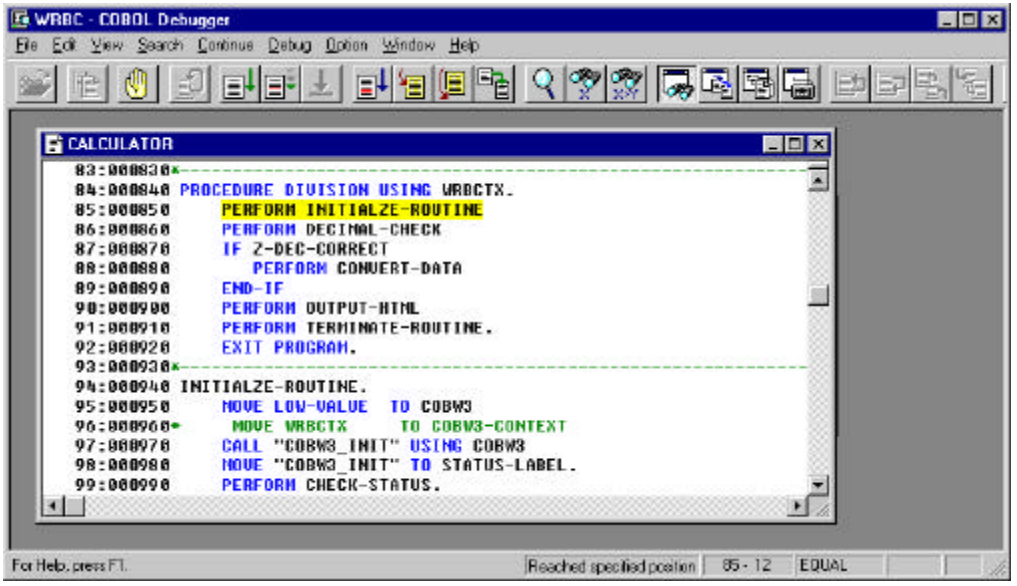

Figure 4.17. The COBOL Debugger window for the WRBC program

3. In the previous figure, line 660 has been commented out to produce the error. To set a window to watch the return value of the COBOL WRB API calls, select the icon for Watch Data or select Add Watch Data from the Debug menu. Alternatively you can use CTRL+W to display the Add Watch Data dialog box. Next, enter the return status value (COBW3-STATUS) for the COBOL WRB API subroutines. The following figure shows the Add Watch Data dialog box filled with the correct information.

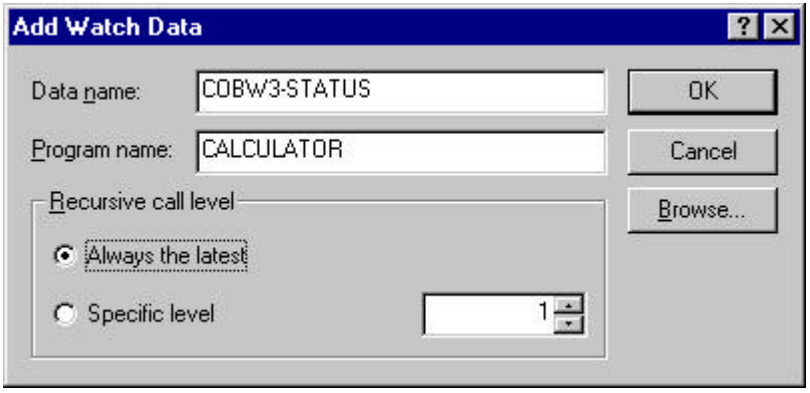

Figure 4.18. The Add Watch Data dialog box

You will see a new window with COBW3-STATUS in it. The initial data value of COBW3-STATUS will be zero.

4. Now select the CALCULATOR source screen, and select the Step Into button to step through the lines of code. The first statement is the PERFORM INITIALZE-ROUTINE. The yellow bar immediately jumps to line 950 of the INITIALZE-ROUTINE.

   Step another line to jump over the comment statement, and then step again to line 980. You will see the value for COBW3-STATUS change from zero to "+0100". Our error has been discovered! The following figure shows the results of the resulting steps.

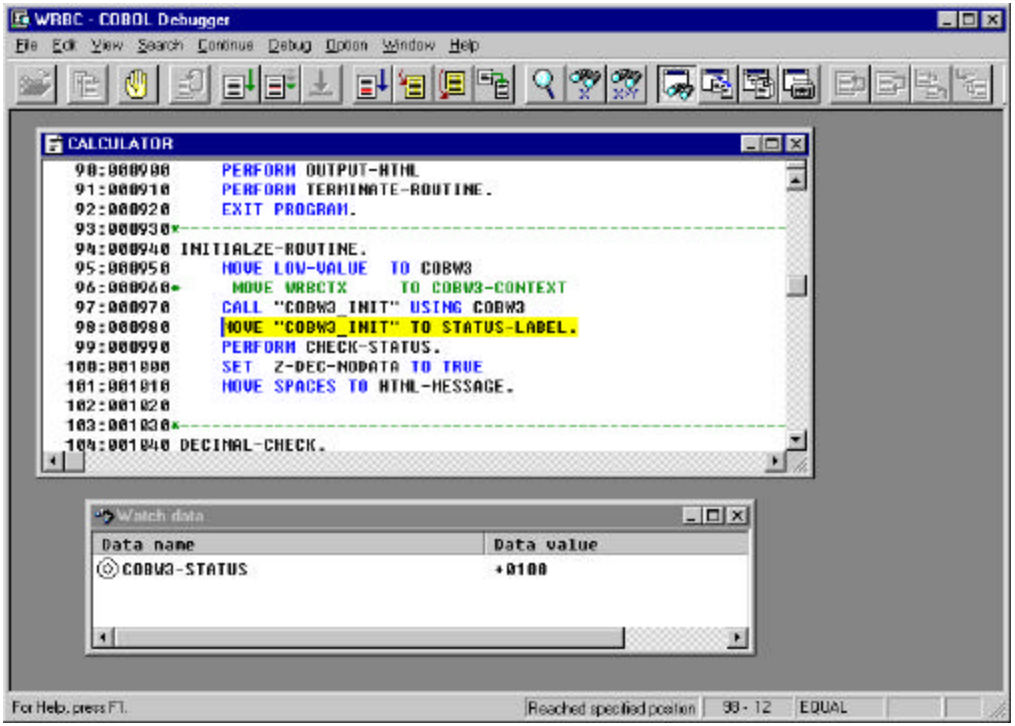

Figure 4.19. The COBOL Debugger window showing the return status of COBW3_INIT

You would want to exit the Debugger at this point, fix your source program, rebuild the project, and step through it again to avoid any other unrecoverable errors.

Using Display for Debugging and Demonstration.

Using the COBOL DISPLAY verb to output data is an easy way to see the progress of the execution of your Web application, and it doesn't require any special compilation or debug options as the Debugger does. However, you must configure your Web Server to interact with the desktop. You may run the browser client on any machine, and the resulting display messages will pop up on the computer that hosts the Web Application Server. This is a

useful technique for demonstration purposes, as well as debugging.

The code below shows the CALCULATOR program with selected parts of code where the DISPLAY statements have been added.

```
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. CALCULATOR.
.
.
000940 INITIALZE-ROUTINE.
000000      DISPLAY "Entered INITIALZE-ROUTINE Paragraph.".
000950     MOVE LOW-VALUE  TO COBW3
.
.
001040 DECIMAL-CHECK.
000000      DISPLAY "Entered DECIMAL-CHECK Paragraph.".
.
.
001330 CONVERT-DATA.
000000      DISPLAY "Entered CONVERT-DATA Paragraph.".
.
.
001490 OUTPUT-HTML.
001500
000000      DISPLAY "Entered OUTPUT-HTML Paragraph.".
.
.
.
002180 CHECK-STATUS.
000000      DISPLAY "Entered OUTPUT-HTML Paragraph.".
000000      DISPLAY COBW3-STATUS.
.
.
002290 TERMINATE-ROUTINE.
000000      DISPLAY "Entered TERMINATE-ROUTINE Paragraph.".
002300     CALL "COBW3_FREE" USING COBW3
```

The following figure shows the output that is displayed from running the CALCULATOR program using these display statements.

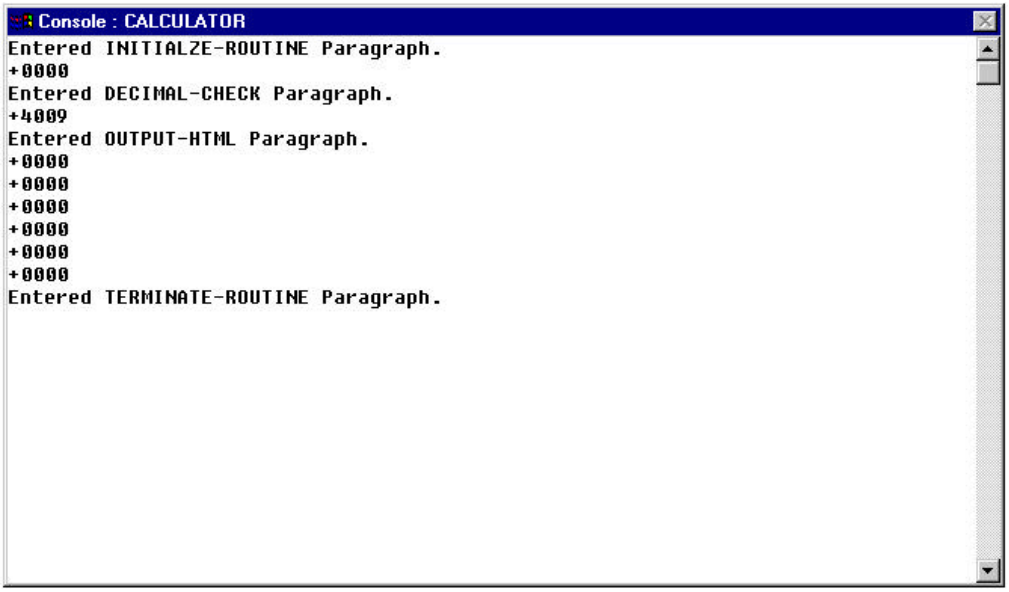

Figure 4.20. Output from COBOL DISPLAY statements

You can see from the above display the paragraphs that were entered, and the status codes that were returned from each of the COBW3 WRB API subroutines.

Appendix A. The COBW3.CBL File

This COBW3.CBL file contains the data definitions that are used by the COBOL Web Request Broker API subroutines. While this file was not included in the listings that demonstrated various techniques in the previous chapters of this manual, you will find that this listing is a good source of documentation to the COBOL WRB API subroutines. It is also a handy file to use in the coding process to copy the names of constants and variables and paste into your program so that the spelling of the names is assured to be correct.

```
000000*****************************************************************
000000* The COBW3 data definition (for V1.0)
000000*  All Rights Reserved, Copyright(C) FUJITSU LIMITED 1997
000000*****************************************************************
000000 01  COBW3.
000000*****************************************************************
000000* STATUS: Error number of subroutine
000000*****************************************************************
000000     03 COBW3-STATUS                       PIC  S9(4) COMP-5.
000000
000000*****************************************************************
000000* DBGINFO: The information data definition related to debugging
000000*****************************************************************
000000     03 COBW3-DBGINFO.
000000*----------------------------------------------------------------
000000* DMODE: Execution instruction in debugging mode
000000*----------------------------------------------------------------
000000       05 COBW3-DMODE                      PIC  X(1).
000000         88 COBW3-DMODE-NODBG              VALUE LOW-VALUE.
000000         88 COBW3-DMODE-DBG                VALUE "1".
000000     03                                     PIC  X(1).
000000     03                                     PIC  X(12).
000000
000000*****************************************************************
000000* The information data definition related to NAME/VALUE
000000*****************************************************************
000000     03 COBW3-NVINFO.
000000*----------------------------------------------------------------
000000* NUMBER: Appearance frequency to be retrieved
000000*----------------------------------------------------------------
000000       05 COBW3-NUMBER                     PIC  S9(4) COMP-5.
000000         88 COBW3-NUMBER-INIT              VALUE 1.
000000*----------------------------------------------------------------
000000* SEARCH-LENGTH: Character string (number of bytes) for retrieval
000000*----------------------------------------------------------------
000000       05 COBW3-SEARCH-LENGTH              PIC  S9(4) COMP-5.
000000*----------------------------------------------------------------
000000* SEARCH-DATA: Character string for retrieval
000000*----------------------------------------------------------------
000000       05 COBW3-SEARCH-DATA                PIC  X(1024).
000000*----------------------------------------------------------------
000000* GET-DATA: Character string of VALUE of NAME
000000*----------------------------------------------------------------
000000       05 COBW3-GET-DATA                   PIC  X(1024).
000000*----------------------------------------------------------------
000000* GET-LENGTH: Length of VALUE character string (number of bytes) to
NAME
000000*----------------------------------------------------------------
000000       05 COBW3-GET-LENGTH                 PIC  S9(4) COMP-5.
000000*----------------------------------------------------------------
000000* SEARCH-FLAG: Retrieval result
000000*----------------------------------------------------------------
000000       05 COBW3-SEARCH-FLAG                PIC  X(1).
```

```
000000         88 COBW3-SEARCH-FLAG-NON          VALUE "0".
000000         88 COBW3-SEARCH-FLAG-EXIST         VALUE "1".
000000       05                                   PIC  X(1).
000000     03                                     PIC  X(12).
000000
000000*****************************************************************
000000* The information data definition related to HTML
000000*****************************************************************
000000     03 COBW3-HTMLINFO.
000000*---------------------------------------------------------------
000000* HTML-FILENAME: HTML document file name
000000*---------------------------------------------------------------
000000       05 COBW3-HTML-FILENAME               PIC  X(256).
000000*---------------------------------------------------------------
000000* HTML-FSTATUS: State of file I/O of HTML document file
000000*---------------------------------------------------------------
000000       05 COBW3-HTML-FS                     PIC  X(2).
000000         88 COBW3-HTML-FS-NORMALEND         VALUE "00".
000000       05                                   PIC  X(1).
000000       05                                   PIC  X(1).
000000*---------------------------------------------------------------
000000* CNV-NAME: Conversion name defined in HTML document
000000*---------------------------------------------------------------
000000       05 COBW3-CNV-NAME                    PIC  X(30).
000000       05                                   PIC  X(2).
000000*---------------------------------------------------------------
000000* CNV-NAME-LENGTH: Character string length of conversion name
000000*  defined in HTML document (byte length)
000000*---------------------------------------------------------------
000000       05 COBW3-CNV-NAME-LENGTH             PIC  S9(4) COMP-5.
000000*---------------------------------------------------------------
000000* CNV-VALUE-LENGTH: Character string length of conversion value
000000*---------------------------------------------------------------
000000       05 COBW3-CNV-VALUE-LENGTH            PIC  S9(4) COMP-5.
000000*---------------------------------------------------------------
000000* CNV-VALUE: Conversion character string defined in HTML document
000000*---------------------------------------------------------------
000000       05 COBW3-CNV-VALUE                   PIC  X(1024).
000000*---------------------------------------------------------------
000000* CNV-MODE: Maintenance of instruction of change and additional
000000*  processing of conversion name defined in HTML document flag
000000*---------------------------------------------------------------
000000       05 COBW3-CNV-MODE                    PIC  X(1).
000000         88 COBW3-CNV-MODE-ADDREP    VALUE LOW-VALUE.
000000         88 COBW3-CNV-MODE-REPLACE   VALUE       "1".
000000         88 COBW3-CNV-MODE-ADD       VALUE       "2".
000000*---------------------------------------------------------------
000000*---------------------------------------------------------------
000000       05                                   PIC  X(3).
000000     03                                     PIC  X(12).
000000
000000*****************************************************************
000000* The information data definition related to PUT_HEAD
000000*****************************************************************
000000     03 COBW3-HEADINFO.
000000*---------------------------------------------------------------
000000* PUT-HEAD-LENGTH: The Header-data length of the data output to Browser
```

```
000000*------05 COBW3-PUT-HEAD-LENGTH------------PIC  S9(8) COMP-5.---
000000*-----------------------------------------------------------------
000000* PUT-HEAD: The Header-data output to Browser(option header)
000000*-----------------------------------------------------------------
000000       05 COBW3-PUT-HEAD                   PIC  X(512).
000000*-----------------------------------------------------------------
000000* CONTENT-TYPE: The Content-Type Header output to Browser
000000*-----------------------------------------------------------------
000000       05 COBW3-CONTENT-TYPE               PIC  X(32).
000000         88 COBW3-CONTENT-TYPE-HTML        VALUE "text/html".
000000         88 COBW3-CONTENT-TYPE-TEXT        VALUE "text/plain".
000000         88 COBW3-CONTENT-TYPE-NON         VALUE HIGH-VALUE.
000000*-----------------------------------------------------------------
000000* STATUS-CODE: The Status-code Header output to Browser
000000*-----------------------------------------------------------------
000000       05 COBW3-STATUS-CODE                PIC  X(3).
000000         88 COBW3-STATUS-CODE-200          VALUE "200".
000000         88 COBW3-STATUS-CODE-NON          VALUE HIGH-VALUE.
000000       05                                   PIC  X(1).
000000     03                                     PIC  X(12).
000000
000000*****************************************************************
000000* The information data definition related to PUT_TEXT
000000*****************************************************************
000000     03 COBW3-TEXTINFO.
000000*-----------------------------------------------------------------
000000* PUT-STRING-LENGTH: The data length of the data output to Browser
000000*-----------------------------------------------------------------
000000       05 COBW3-PUT-STRING-LENGTH          PIC  S9(8) COMP-5.
000000*-----------------------------------------------------------------
000000* PUT-STRING: The data output to Browser
000000*-----------------------------------------------------------------
000000       05 COBW3-PUT-STRING                 PIC  X(1024).
000000     03                                     PIC  X(12).
000000
000000*****************************************************************
000000* The information data definition related to AUTHORIZE
000000*****************************************************************
000000     03 COBW3-AUTHORIZEINFO.
000000*-----------------------------------------------------------------
000000* USER-ID: User ID information on client
000000*-----------------------------------------------------------------
000000       05 COBW3-USERID-LENGTH              PIC  S9(4)  COMP-5.
000000       05 COBW3-USERID                     PIC  X(90).
000000       05 COBW3-PASSWORD-LENGTH            PIC  S9(4)  COMP-5.
000000       05 COBW3-PASSWORD                   PIC  X(90).
000000*-----------------------------------------------------------------
000000* IP-ADDRESS: Internet Protocol address information on client
000000*-----------------------------------------------------------------
000000       05 COBW3-IP-ADDRESS-LENGTH          PIC  S9(4)  COMP-5.
000000       05 COBW3-IP-ADDRESS                 PIC  X(90).
000000     03                                     PIC  X(12).
000000
000000*****************************************************************
000000* The information data definition related to SYSTEM
000000*****************************************************************
```

```
000000*----03 COBW3-SYSTEMINFO.-------------------------------------
000000* SYSTEM-COMMAND: System command and parameter character string
000000*-------------------------------------------------------------
000000       05 COBW3-SYSTEM-COMMAND             PIC  X(512).
000000     03                                    PIC  X(12).
000000
000000*****************************************************************
000000* WRBINFO: The information data definition related to WRB
000000*****************************************************************
000000     03 COBW3-WRBINFO.
000000*-------------------------------------------------------------
000000* CONTEXT WRB context
000000*-------------------------------------------------------------
000000       05 COBW3-CONTEXT                    POINTER.
000000       05                                 PIC  X(216).
000000
000000*****************************************************************
000000* WORKINFO: The information data definition related to subroutine
000000*  work area
000000*****************************************************************
000000     03 COBW3-WORKINFO                    PIC  X(600).
```

Appendix B. Listing of CALCULATE_BIG.COB

This listing demonstrates a different technique in building dynamic HTML Web pages using the COBOL WRB API subroutines. This technique avoids the use of the conversion subroutines in the COBOL WRB API. When you compare this listing with the listing of CALCULATOR.COB in Chapter 3, you can see that using the conversion routines require fewer lines of code.

```
000000       05 COBW3-DMODE                       PIC  X(1).
000000         88 COBW3-DMODE-NODBG                VALUE LOW-VALUE.
000000         88 COBW3-DMODE-DBG                  VALUE "1".
000010 IDENTIFICATION DIVISION.
000020 PROGRAM-ID. CALCULATOR.
000030*****************************************
000040* Crafted by SHY
000050*****************************************
000060 ENVIRONMENT DIVISION.
000070 CONFIGURATION SECTION.
000080 SPECIAL-NAMES.
000090     SYMBOLIC CONSTANT
000100     SC-COBW3-SEARCH-NOT-FOUND IS "O"
000110     SC-COBW3-SEARCH-FOUND     IS "1"
000120     SC-NUMBER-OF-HEX            IS 16
000130     SC-NUMBER-OF-OCT            IS 21
000140     SC-NUMBER-OF-BIT            IS 64
000150     SC-WRB-INVALID-INPUT        IS "Invalid data. Please reenter the
data.".
000160 DATA DIVISION.
000170 WORKING-STORAGE SECTION.
000180 01  HTML-HEAD           PIC X(15)  VALUE '<HTML><HEAD>'.
000190 01  HTML-TITLE          PIC X(50)  VALUE '<TITLE>The Amazing Cobol
Calculator</TITLE></HEAD>'.
000200 01  HTML-BODY           PIC X(50)  VALUE '<BODY BGCOLOR="#000FFF"
TEXT="#FFF000">'.
000210 01  HTML-FORM           PIC X(80)  VALUE '<FORM METHOD="GET"
ACTION="/que-bin/cobol/calculator.DLL/CALCULATOR">'.
000220 01  HTML-HEADER1        PIC X(120)  VALUE
'<P><CENTER><H1>Calculator</H1><H4>Converting Decimal to Hexadecimal, Octal,
& Binary</H4></CENTER></P>'.
000230 01  HTML-DECNUMBER      PIC X(120) VALUE '<P><STRONG><FONT
FACE="Courier New">Dec Number : </FONT></STRONG><INPUT TYPE="text"
NAME="DECNUMBER" SIZE="18"></P>'.
000240 01  HTML-DECNUMBER-REPLY-HEAD.
000250     02                PIC X(36) VALUE '<P><STRONG><FONT
FACE="Courier New">'.
000260     02                PIC X(29) VALUE 'Dec Number :
</FONT></STRONG>'.
000270     02                PIC X(53) VALUE '<INPUT TYPE="text"
NAME="DECNUMBER" SIZE="18" VALUE="'.
000280 01  HTML-DECNUMBER-REPLY-FOOT  PIC X(6) VALUE '"></P>'.
000290 01  HTML-SUBMIT-BUTTON  PIC X(80)  VALUE '<P><CENTER><INPUT
TYPE="SUBMIT" VALUE="Calculate">'.
000300 01  HTML-CLEAR-BUTTON   PIC X(90)  VALUE '<INPUT TYPE="reset"
VALUE="Clear"></CENTER></P>'.
000310 01  HTML-HEXNUMBER      PIC X(120) VALUE '<HR><P><STRONG><FONT
FACE="Courier New">Hex Number : </FONT></STRONG><INPUT TYPE="text"
NAME="HEXNUMBER" SIZE="16"><BR>'.
000320 01  HTML-HEXNUMBER-ANS.
000330     02                PIC X(40)  VALUE '<HR><P><STRONG><FONT
FACE="Courier New">'.
000340     02                PIC X(29)  VALUE 'Hex Number :
</FONT></STRONG>'.
```

```
000350     02                   PIC X(53)  VALUE '<INPUT TYPE="text"
NAME="HEXNUMBER" SIZE="16" VALUE="'.
000360     02  HTML-HEX-ANS     PIC X(16).
000370     02                   PIC X(6)   VALUE '"><BR>'.
000380 01  HTML-OCTNUMBER       PIC X(130) VALUE '<STRONG><FONT FACE="Courier
New">Oct Number : </FONT></STRONG><INPUT TYPE="text" NAME="OCTNUMBER"
SIZE="21"><BR>'.
000390 01  HTML-OCTNUMBER-ANS.
000400     02                   PIC X(33)  VALUE '<STRONG><FONT FACE="Courier
New">'.
000410     02                   PIC X(29)  VALUE 'Oct Number :
</FONT></STRONG>'.
000420     02                   PIC X(53)  VALUE '<INPUT TYPE="text"
NAME="OCTNUMBER" SIZE="21" VALUE="'.
000430     02  HTML-OCT-ANS     PIC X(21).
000440     02                   PIC X(6)   VALUE '"><BR>'.
000450 01  HTML-BITNUMBER       PIC X(130) VALUE '<STRONG><FONT FACE="Courier
New">Bit Number : </FONT></STRONG><INPUT TYPE="text" NAME="BITNUMBER"
SIZE="64"></P>'.
000460 01  HTML-BITNUMBER-ANS.
000470     02                   PIC X(33)  VALUE '<STRONG><FONT FACE="Courier
New">'.
000480     02                   PIC X(29)  VALUE 'Bit Number :
</FONT></STRONG>'.
000490     02                   PIC X(53)  VALUE '<INPUT TYPE="text"
NAME="BITNUMBER" SIZE="64" VALUE="'.
000500     02  HTML-BIT-ANS     PIC X(64).
000510     02                   PIC X(6)   VALUE '"></P>'.
000520 01  HTML-MESSAGE.
000530     02                       PIC X(10)  VALUE '<P><BLINK>'.
000540     02  HTML-MSG-CONTENTS    PIC X(50).
000550     02                       PIC X(4)   VALUE '</P>'.
000560 01  HTML-END-OF-HEAD     PIC X(30)  VALUE '</FORM></BODY></HTML>'.
000570 01  HTML-TEXT            PIC X(200).
000580
000590 01  INPUT-NUMBER         PIC 9(18).
000600
000610 01  BINARY-NUMBER        PIC 9(18) BINARY.
000620 01    REDEFINES BINARY-NUMBER.
000630     02  BINARY-HEX OCCURS SC-NUMBER-OF-HEX TIMES    PIC 1(4) BIT.
000640 01    REDEFINES BINARY-NUMBER.
000650     02                                             PIC 1 BIT.
000660     02  BINARY-OCT OCCURS SC-NUMBER-OF-OCT TIMES    PIC 1(3) BIT.
000670 01    REDEFINES BINARY-NUMBER.
000680     02  BINARY-BIT OCCURS SC-NUMBER-OF-BIT TIMES    PIC 1 BIT.
000690
000700 01  HEX-EXPRESSION       PIC X(16) VALUE "0123456789ABCDEF".
000710 01    REDEFINES HEX-EXPRESSION.
000720     02  HEX-DATA OCCURS 16 TIMES   PIC X.
000730
000740 01  OCT-EXPRESSION       PIC X(8) VALUE "01234567".
000750 01    REDEFINES OCT-EXPRESSION.
000760     02  OCT-DATA OCCURS 8 TIMES    PIC X.
000770
000780 01  I                    PIC 9(4) COMP-5.
000790
000800 01  HEX-TO-BINARY.
```

```
000810     02                       PIC 1(13) BIT VALUE ALL B"0".
000820     02  HEX-LOWER-PART       PIC 1(4) BIT.
000830 01  BINARY-FROM-HEX REDEFINES HEX-TO-BINARY PIC 9(4) BINARY.
000840
000850 01  OCT-TO-BINARY.
000860     02                       PIC 1(13) BIT VALUE ALL B"0".
000870     02  OCT-LOWER-PART       PIC 1(3) BIT.
000880 01  BINARY-FROM-OCT REDEFINES OCT-TO-BINARY PIC 9(4) BINARY.
000890
000900 01                     PIC X.
000910     88  Z-DEC-NODATA   VALUE "N".
000920     88  Z-DEC-CORRECT  VALUE "C".
000930     88  Z-DEC-INVALID  VALUE "I".
000940
000950 01  DECNUMBER-SAVE     PIC X(18).
000960
000970 01  HEX-ANSWER         PIC X(16).
000980 01  OCT-ANSWER         PIC X(21).
000990 01  BIT-ANSWER         PIC X(64).
001000
001010*=========================================================*
001020*  WRB API ROUTINE WORK AREA                               *
001030*=========================================================*
001040     COPY  COBW3.
001050
001060 CONSTANT SECTION.
001070 01  C-DECNUMBER        PIC X(9) VALUE "DECNUMBER".
001080
001090 LINKAGE         SECTION.
001100 01  WRBCTX             POINTER.
001110
001120 PROCEDURE       DIVISION USING WRBCTX.
001130     MOVE LOW-VALUE                      TO    COBW3.
001140     MOVE WRBCTX                         TO    COBW3-CONTEXT.
001150     CALL "COBW3_INIT"              USING COBW3.
001160     SET  Z-DEC-NODATA                   TO    TRUE.
001170     MOVE SPACE                          TO    HTML-MSG-CONTENTS.
001180 DECIMAL-CHECK.
001190     MOVE FUNCTION LENG (C-DECNUMBER)    TO    COBW3-SEARCH-LENGTH.
001200     MOVE C-DECNUMBER                    TO    COBW3-SEARCH-DATA.
001210     MOVE 1                              TO    COBW3-NUMBER.
001220     CALL "COBW3_NAME"              USING COBW3.
001230     IF COBW3-SEARCH-FLAG = SC-COBW3-SEARCH-NOT-FOUND OR
001240        COBW3-GET-DATA = SPACE
001250        GO TO OUTPUT-HTML
001260     END-IF.
001270
001280     MOVE COBW3-GET-DATA              TO    DECNUMBER-SAVE.
001290
001300     IF COBW3-GET-DATA (1 : COBW3-GET-LENGTH) IS NOT NUMERIC
001310        MOVE SC-WRB-INVALID-INPUT        TO    HTML-MSG-CONTENTS
001320        SET  Z-DEC-INVALID               TO    TRUE
001330        GO TO OUTPUT-HTML
001340     END-IF.
001350
001360     MOVE ALL ZERO                       TO    INPUT-NUMBER.
```

```
001370     PERFORM TEST BEFORE VARYING I FROM COBW3-GET-LENGTH BY -1 UNTIL I
001380       MOVE COBW3-GET-DATA (I : 1)    TO    INPUT-NUMBER (18 - COBW3-
GET-LENGTH + I : 1)
001390     END-PERFORM.
001400     MOVE INPUT-NUMBER                  TO    BINARY-NUMBER.
001410
001420 CONVERT-DATA.
001430     SET  Z-DEC-CORRECT                        TO TRUE.
001440     PERFORM TEST BEFORE VARYING I FROM 1 BY 1 UNTIL I > SC-NUMBER-OF-
HEX
001450       MOVE  BINARY-HEX (I)                    TO  HEX-LOWER-PART
001460       MOVE  HEX-DATA (BINARY-FROM-HEX + 1)  TO  HEX-ANSWER (I : 1)
001470     END-PERFORM.
001480
001490     PERFORM TEST BEFORE VARYING I FROM 1 BY 1 UNTIL I > SC-NUMBER-OF-
OCT
001500       MOVE  BINARY-OCT (I)                     TO  OCT-LOWER-PART
001510       MOVE  OCT-DATA (BINARY-FROM-OCT + 1)   TO  OCT-ANSWER (I : 1)
001520     END-PERFORM.
001530
001540     PERFORM TEST BEFORE VARYING I FROM 1 BY 1 UNTIL I > SC-NUMBER-OF-
BIT
001550       MOVE  BINARY-BIT (I)                     TO  BIT-ANSWER (I : 1)
001560     END-PERFORM.
001570
001580 OUTPUT-HTML.
001590     MOVE FUNCTION LENG(HTML-HEAD)           TO    COBW3-PUT-STRING-
LENGTH.
001600     MOVE HTML-HEAD                          TO    COBW3-PUT-STRING.
001610     CALL "COBW3_PUT_TEXT"                USING COBW3.
001620
001630     MOVE FUNCTION LENG(HTML-TITLE)          TO    COBW3-PUT-STRING-
LENGTH.
001640     MOVE HTML-TITLE                         TO    COBW3-PUT-STRING.
001650     CALL "COBW3_PUT_TEXT"                USING COBW3.
001660
001670     MOVE FUNCTION LENG(HTML-BODY)           TO    COBW3-PUT-STRING-
LENGTH.
001680     MOVE HTML-BODY                          TO    COBW3-PUT-STRING.
001690     CALL "COBW3_PUT_TEXT"                USING COBW3.
001700
001710     MOVE FUNCTION LENG(HTML-FORM)           TO    COBW3-PUT-STRING-
LENGTH.
001720     MOVE HTML-FORM                          TO    COBW3-PUT-STRING.
001730     CALL "COBW3_PUT_TEXT"                USING COBW3.
001740
001750     MOVE FUNCTION LENG(HTML-HEADER1)        TO    COBW3-PUT-STRING-
LENGTH.
001760     MOVE HTML-HEADER1                       TO    COBW3-PUT-STRING.
001770     CALL "COBW3_PUT_TEXT"                USING COBW3.
001780
001790     IF Z-DEC-NODATA
001800        MOVE FUNCTION LENG(HTML-DECNUMBER)   TO    COBW3-PUT-STRING-
LENGTH
001810        MOVE HTML-DECNUMBER                  TO    COBW3-PUT-STRING
001820     ELSE
```

```
001830        MOVE 1                                  TO    COBW3-PUT-STRING-
LENGTH
001840        STRING  HTML-DECNUMBER-REPLY-HEAD    DELIMITED BY SIZE
001850                DECNUMBER-SAVE               DELIMITED BY SPACE
001860                HTML-DECNUMBER-REPLY-FOOT    DELIMITED BY SIZE
001870           INTO COBW3-PUT-STRING  WITH POINTER COBW3-PUT-STRING-LENGTH
001880        END-STRING
001890     END-IF.
001900     CALL "COBW3_PUT_TEXT"               USING COBW3.
001910
001920     MOVE FUNCTION LENG(HTML-SUBMIT-BUTTON)  TO    COBW3-PUT-STRING-
LENGTH.
001930     MOVE HTML-SUBMIT-BUTTON                 TO    COBW3-PUT-STRING.
001940     CALL "COBW3_PUT_TEXT"               USING COBW3.
001950
001960     MOVE FUNCTION LENG(HTML-CLEAR-BUTTON)   TO    COBW3-PUT-STRING-
LENGTH.
001970     MOVE HTML-CLEAR-BUTTON                  TO    COBW3-PUT-STRING.
001980     CALL "COBW3_PUT_TEXT"               USING COBW3.
001990
002000     IF Z-DEC-CORRECT
002010        MOVE HEX-ANSWER                          TO    HTML-HEX-ANS
002020        MOVE FUNCTION LENG(HTML-HEXNUMBER-ANS)   TO    COBW3-PUT-
STRING-LENGTH
002030        MOVE HTML-HEXNUMBER-ANS                  TO    COBW3-PUT-
STRING
002040     ELSE
002050        MOVE FUNCTION LENG(HTML-HEXNUMBER)       TO    COBW3-PUT-
STRING-LENGTH
002060        MOVE HTML-HEXNUMBER                      TO    COBW3-PUT-
STRING
002070     END-IF.
002080     CALL "COBW3_PUT_TEXT"               USING COBW3.
002090
002100     IF Z-DEC-CORRECT
002110        MOVE OCT-ANSWER                          TO    HTML-OCT-ANS
002120        MOVE FUNCTION LENG(HTML-OCTNUMBER-ANS)   TO    COBW3-PUT-
STRING-LENGTH
002130        MOVE HTML-OCTNUMBER-ANS                  TO    COBW3-PUT-
STRING
002140     ELSE
002150        MOVE FUNCTION LENG(HTML-OCTNUMBER)       TO    COBW3-PUT-
STRING-LENGTH
002160        MOVE HTML-OCTNUMBER                      TO    COBW3-PUT-
STRING
002170     END-IF.
002180     CALL "COBW3_PUT_TEXT"               USING COBW3.
002190
002200     IF Z-DEC-CORRECT
002210        MOVE BIT-ANSWER                          TO    HTML-BIT-ANS
002220        MOVE FUNCTION LENG(HTML-BITNUMBER-ANS)   TO    COBW3-PUT-
STRING-LENGTH
002230        MOVE HTML-BITNUMBER-ANS                  TO    COBW3-PUT-
STRING
002240     ELSE
002250        MOVE FUNCTION LENG(HTML-BITNUMBER)       TO    COBW3-PUT-
STRING-LENGTH
```

```
002260        MOVE HTML-BITNUMBER                          TO    COBW3-PUT-
STRING
002270     END-IF.
002280     CALL "COBW3_PUT_TEXT"               USING COBW3.
002290
002300     MOVE FUNCTION LENG(HTML-END-OF-HEAD)    TO    COBW3-PUT-STRING-
LENGTH.
002310     MOVE HTML-END-OF-HEAD                   TO    COBW3-PUT-STRING.
002320     CALL "COBW3_PUT_TEXT"               USING COBW3.
002330
002340     IF HTML-MESSAGE NOT = SPACE
002350        MOVE FUNCTION LENG(HTML-MESSAGE)     TO    COBW3-PUT-STRING-
LENGTH
002360        MOVE HTML-MESSAGE                    TO    COBW3-PUT-STRING
002370        CALL "COBW3_PUT_TEXT"            USING COBW3
002380     END-IF.
002390
002400 END-PROC.
002410     CALL "COBW3_FREE"                   USING COBW3.
002420
2430       EXIT PROGRAM.
```

Appendix C. Using the PL/SQL Web Toolkit

The following three listings are used with the program discussed in Chapter 3 called BIRTHDAY.COB. Oracle PL/SQL offers a means to have native connections to your Oracle database. When using Web Application Server PL/SQL Web Toolkit, dynamic Web pages are very easily created. Using different Cartridges with your COBOL Cartridge can leverage your productivity. For more information on using PL/SQL, and the PL/SQL Web Toolkit, see your Oracle documentation.

```
000000       05 COBW3-DMODE                      PIC  X(1).



PROCEDURE insertBirthday( p_first IN VARCHAR2,
                          p_last  IN VARCHAR2,
                          p_dob   IN DATE )
IS
BEGIN

              INSERT INTO birthday ( firstName, lastName, dob ) VALUES
                  ( p_first, p_last, p_dob );
              getBirthdays;
    EXCEPTION
        WHEN OTHERS THEN
            ROLLBACK;
            htp.br;
            htp.p('Oracle error:' || SQLERRM);
END;
```

Figure C.1. PL/SQL Procedure called to insert a record into the birthday table

```
PROCEDURE getBrithdays
 IS
    l_sql_statement   VARCHAR2(2000);
BEGIN
    l_sql_statement  := 'SELECT firstName, lastName, dob FROM birthday ORDER
BY dob';
    htp.htmlOpen;
    htp.bodyOpen(  cattributes => ' BGCOLOR=#FFF8DC');
    htp.centerOpen;
    htp.tableOpen( 'border' );
       htp.tableCaption( htf.fontOpen(ccolor=>'#0000FF',csize=>'5') ||
                                    htf.bold('Birthday List'),'center');

        htp.tableHeader( htf.bold('First'),'center');
        htp.tableHeader( htf.bold('Last'),'center');
        htp.tableHeader( htf.bold('Birthday'),'center');
        htp.tableHeader( htf.bold('Play'),'center');
        owa_util.cellsprint( l_sql_statement  );
    htp.tableClose;
    htp.centerClose;
    htp.bodyClose;
   htp.htmlClose;
  EXCEPTION
    WHEN OTHERS THEN
            htp.br;
            htp.p('Oracle error:' || SQLERRM);
END;
```

Figure C.2. PL/SQL Procedure dynamically creates a table of birthday names

```
CREATE TABLE  birthday(
              firstName   VARCHAR2(40),
              lastName    VARCHAR2(40),
              dob        DATE);
```

Figure C. 3. Data definition for the birthday table

Index