

FUJITSU Software

NetCOBOL V11.0

A horizontal band with a red abstract graphic featuring glowing, overlapping lines and curves, creating a sense of motion and energy.

ISAPI Subroutines User's Guide

Windows

B1WD-3362-01ENZ0(00)
August 2015

Preface

"NetCOBOL ISAPI Subroutines 3.1" operate in 32-bit mode of the following operating systems and provide the development environment and execution environment for 32-bit applications that execute under Microsoft® Internet Information Server (called "IIS" from here on).

- Microsoft® Windows Server® 2012 R2
- Microsoft® Windows Server® 2012
- Microsoft® Windows Server® 2008 R2

Purpose of This Manual

This manual explains how to create COBOL programs using NetCOBOL ISAPI Subroutines 3.1 and how to execute and debug these types of programs.

For the creation, compilation, execution and debugging of COBOL programs, please refer to the "*NetCOBOL User's Guide*" manual.

For the debugging of COBOL Web applications, refer to the "*NetCOBOL Debugging Guide*".

For the COBOL syntax rules, please refer to the "*NetCOBOL Language Reference*" manual.

To create a Web application that runs under Unicode, please refer to the "*Use of Unicode*" in "NetCOBOL Web Guide" manual.

Product Names

Product names described in this manual are abbreviated as follows:

Product Name	Abbreviation
Microsoft® Windows Server® 2012 R2 Datacenter Microsoft® Windows Server® 2012 R2 Standard Microsoft® Windows Server® 2012 R2 Essentials Microsoft® Windows Server® 2012 R2 Foundation	Windows Server 2012 R2
Microsoft® Windows Server® 2012 Datacenter Microsoft® Windows Server® 2012 Standard Microsoft® Windows Server® 2012 Essentials Microsoft® Windows Server® 2012 Foundation	Windows Server 2012
Microsoft® Windows Server® 2008 R2 Foundation Microsoft® Windows Server® 2008 R2 Standard Microsoft® Windows Server® 2008 R2 Enterprise Microsoft® Windows Server® 2008 R2 Datacenter	Windows Server 2008 R2
Microsoft® Internet Information Server	IIS
Microsoft® Internet Explorer	IE

Trademarks

- NetCOBOL is a trademark or registered trademark of Fujitsu Limited or its subsidiaries in the United States or other countries or in both.
- Microsoft, Windows, Windows Server, Visual Basic and Visual C++ are trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.
- Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners. Oracle Solaris might be described as Solaris, Solaris Operating System, or Solaris OS.
- Other product names are trademarks or registered trademarks of each company. Trademark indications are omitted for some system and product names described in this manual.

- The permission of the Microsoft Corporation has been obtained for the use of screen images.

Export Regulation

Exportation/release of this document may require necessary procedures in accordance with the regulations of your resident country and/or US export control laws.

The contents of this manual may be revised without prior notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Fujitsu Limited.

August 2015

Copyright 1996-2015 FUJITSU LIMITED

Contents

Chapter 1 ISAPI Subroutines.....	1
1.1 Overview of ISAPI.....	1
1.2 ISAPI Application Configuration.....	1
1.3 COBOL ISAPI Subroutines.....	2
Chapter 2 Creating a Web Application Using ISAPI Subroutines.....	3
2.1 What is Required.....	3
2.2 What to Create.....	3
2.2.1 GetExtensionVersion.....	3
2.2.2 HttpExtensionProc.....	5
2.2.3 TerminateExtension.....	6
2.2.4 Web Page for Invoking AN Application.....	7
2.2.5 Web Page for Processing the Resulting Output.....	7
2.3 Precautions to Take When Creating an ISAPI Application Using COBOL.....	8
2.4 Using the Web Application Development Support Function.....	8
Chapter 3 How to Use the ISAPI Subroutines.....	9
3.1 Functions of the ISAPI Subroutines.....	9
3.1.1 Basic Functions.....	9
3.1.2 File Upload Function.....	11
3.1.3 Session Management Function.....	13
3.2 ISAPI Subroutines Interface.....	15
3.2.1 Library Files.....	15
3.2.2 Calling ISAPI Subroutines.....	16
3.3 Set the Work Environment and Acquire Web Parameters.....	17
3.3.1 COBW3_INIT.....	17
3.4 Manipulate Web Parameters.....	17
3.4.1 COBW3_GET_VALUE, COBW3_GET_VALUE_XX, COBW3_GET_VALUE_NX, COBW3_GET_VALUE_XN, and COBW3_GET_VALUE_NN.....	17
3.4.2 COBW3_CHECK_VALUE, COBW3_CHECK_VALUE_X and COBW3_CHECK_VALUE_N.....	19
3.5 Output Processing Results.....	20
3.5.1 COBW3_PUT_HEAD.....	20
3.5.2 COBW3_PUT_HTML.....	21
3.5.3 COBW3_SET_CNV, COBW3_SET_CNV_XX, COBW3_SET_CNV_NX, COBW3_SET_CNV_XN, and COBW3_SET_CNV_NN.....	24
3.5.4 COBW3_DEL_CNV, COBW3_DEL_CNV_X and COBW3_DEL_CNV_N.....	27
3.5.5 COBW3_INIT_CNV.....	28
3.5.6 COBW3_SET_REPEAT, COBW3_SET_REPEAT_XX, COBW3_SET_REPEAT_NX, COBW3_SET_REPEAT_XN, and COBW3_SET_REPEAT_NN.....	29
3.5.7 COBW3_DEL_REPEAT, COBW3_DEL_REPEAT_X and COBW3_DEL_REPEAT_N.....	31
3.5.8 COBW3_INIT_REPEAT.....	32
3.5.9 COBW3_PUT_TEXT.....	33
3.6 Execute System Commands.....	34
3.6.1 COBW3_SYSTEM.....	34
3.7 Release ISAPI Execution Environment Resources.....	34
3.7.1 COBW3_FREE.....	34
3.8 Acquire Request Information.....	34
3.8.1 COBW3_RECEIVE_HEADER.....	34
3.8.2 COBW3_GET_REQUEST_INFO.....	35
3.8.3 COBW3_GET_AUTHORIZE.....	36
3.9 Manipulate Cookie Data.....	37
3.9.1 COBW3_SET_COOKIE, COBW3_SET_COOKIE_XX, COBW3_SET_COOKIE_NX, COBW3_SET_COOKIE_XN, and COBW3_SET_COOKIE_NN.....	37
3.9.2 COBW3_DEL_COOKIE, COBW3_DEL_COOKIE_X and COBW3_DEL_COOKIE_N.....	39
3.9.3 COBW3_INIT_COOKIE.....	40

3.9.4 COBW3_GET_COOKIE, COBW3_GET_COOKIE_XX, COBW3_GET_COOKIE_NX, COBW3_GET_COOKIE_XN, and COBW3_GET_COOKIE_NN.....	41
3.10 Manipulate Uploaded Files.....	42
3.10.1 COBW3_GET_UPLOADFILE_INFO, COBW3_GET_UPLOADFILE_INFO_X and COBW3_GET_UPLOADFILE_INFO_N.....	42
3.10.2 COBW3_GEN_UPLOADFILE, COBW3_GEN_UPLOADFILE_X and COBW3_GEN_UPLOADFILE_N.....	44
3.10.3 COBW3_DEL_UPLOADFILE.....	46
3.11 Manage Sessions.....	46
3.11.1 COBW3_START_SESSION.....	46
3.11.2 COBW3_END_SESSION.....	47
3.11.3 COBW3_SET_SESSION_DATA.....	47
3.11.4 COBW3_GET_SESSION_DATA.....	48
3.11.5 COBW3_ALTER_SESSION_TIMEOUT.....	49
3.11.6 COBW3_GET_SESSION_INFO.....	49
3.12 Other Subroutines.....	50
3.12.1 COBW3_NAME.....	50
3.12.2 COBW3_VALUE.....	51
3.12.3 COBW3_CNV_SET.....	52
3.12.4 COBW3_CNV_DEL.....	53
3.12.5 COBW3_CNV_INIT.....	53
3.13 Quantitative Limitations of ISAPI Subroutines.....	54
3.14 ISAPI Subroutine Classes.....	54
3.14.1 COBW3-SESSION-ADAPTER class.....	54
Chapter 4 Creating and Executing a Web Application.....	56
4.1 Execution Procedure.....	56
4.2 Compiling and Linking.....	56
4.3 IIS Settings.....	57
4.4 ISAPI Subroutine Environment Variable Settings.....	64
4.4.1 Required Settings in System Environment Variables.....	64
4.4.2 Required Settings in a System Environment Variable or a Run-Time Initialization File.....	65
4.5 Executing a Web Application.....	65
Chapter 5 Operation Check.....	66
5.1 Referencing Log Information.....	66
5.2 Checking the Operation Using the Interactive Debugger.....	66
5.2.1 Starting the Debugger.....	67
5.2.2 Debugging.....	67
5.2.3 Terminating the Debugger.....	67
5.2.4 Swapping Web Applications.....	68
5.3 Referencing an Error Detected by an ISAPI Subroutine.....	68
5.4 Referencing the Data Being Executed in a Display Format.....	69
Chapter 6 Sample Programs.....	70
6.1 Provided Programs.....	70
6.2 ISAPI Subroutines Referenced.....	70
6.3 Compiling the Programs.....	70
6.4 Linking the Programs.....	70
6.5 Environment Settings.....	71
6.6 Executing the Sample.....	71
6.7 Explanation of the Sample.....	71
6.7.1 Retrieving Cookie Data.....	75
6.7.2 Registering Cookie Data.....	76
6.7.3 Retrieving Request Information.....	76
6.7.4 Retrieving Header Information.....	76
Appendix A Questions and Answers.....	77

Appendix B Error Handling.....	83
Appendix C Concept Diagram of Creating Internet Server Applications in COBOL.....	89
Appendix D CGI to ISAPI Subroutines Conversion Guide.....	90
D.1 Conversion from CGI to ISAPI.....	90
D.2 Application Formats.....	91
D.3 Web Page for Invoking an Application.....	91
D.4 Compilation and Linkage Methods.....	91
D.5 Run Unit.....	92
D.6 Compilation Method.....	92
D.7 Access to a Shared Resource.....	92
D.8 Module Configuration.....	92
D.9 Interface Area with a Server.....	93
D.10 Manipulating an Environment Variable.....	94
D.11 Manipulating a CGI Environment Variable.....	95

Chapter 1 ISAPI Subroutines

1.1 Overview of ISAPI

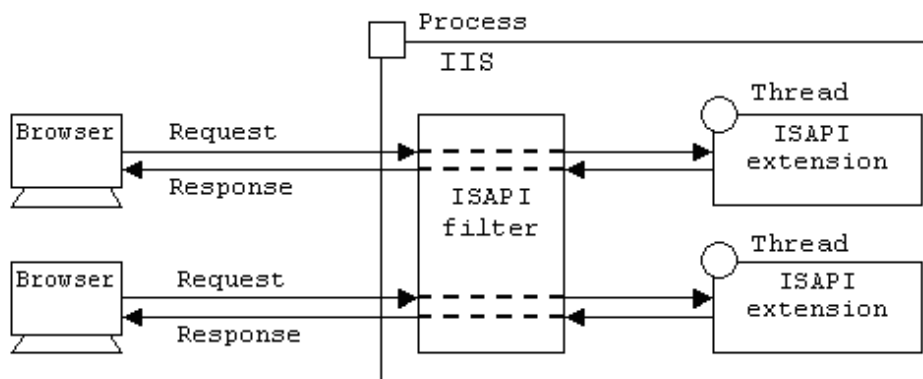
ISAPI (Internet Server Application Programming Interface) is a framework to expand IIS and a programming interface that can be used in the framework. When an Internet Server Application (hereafter abbreviated as ISA) is an application using ISAPI, it runs as a thread in IIS. Therefore, ISA's typically execute faster than traditional CGI applications. Also, less system resources such as memory are required. However, since ISA's run as a thread, they must be thread-safe. Do not forget to release resources, because, if a resource is not released, the resource may remain unusable until IIS itself is terminated. The following two types of applications can be created using ISAPI:

ISAPI extension

Unit to install each Web application. It is possible to change applications to be used for each request.

ISAPI filter

Web applications invoked in common throughout the Web site. ISAPI filter's are executed before or after an ISAPI extension and are used when processing common throughout applications is required.



1.2 ISAPI Application Configuration

This section explains regarding ISA's, the configuration of ISAPI extensions. However, ISAPI filters are not supported by COBOL ISAPI Subroutines. For more information on ISAPI Subroutines, see the books and IIS manuals.

An ISAPI extension must be a DLL that exports the following three functions. Each individual function runs as a multi-threaded function.

GetExtensionVersion

This function is called once when an application DLL is loaded. The HttpExtensionProc function noted below is executed only after this function is terminated.

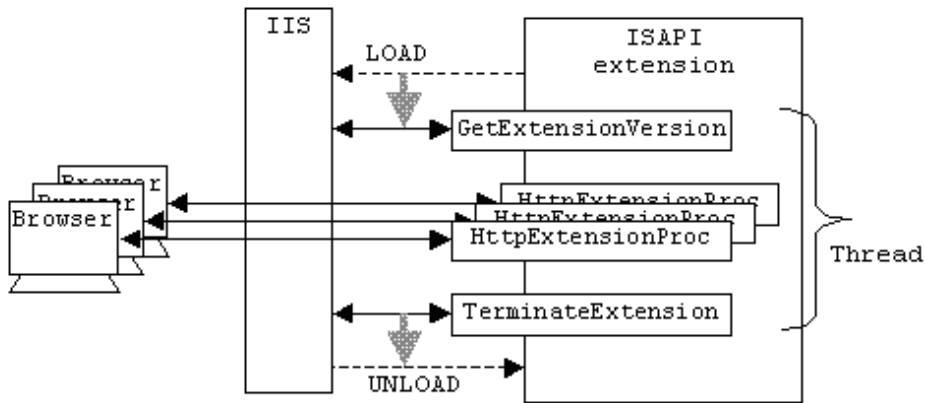
HttpExtensionProc

Application called for each request. This function is used to receive data sent from the WWW Browser and to send HTML documents to the WWW Browser and is provided as an interface to IIS.

TerminateExtension

This function is called when the application DLL is unloaded.

Figure 1.1 How each export function is called



1.3 COBOL ISAPI Subroutines

COBOL ISAPI Subroutines are Web subroutines that use ISAPI. By using these subroutines, an ISAPI extension can be created easily with COBOL. That is, a Web application using this subroutine inherits the features of ISAPI directly. Since ISAPI Subroutines are upward compatible with COBOL CGI subroutines, a Web application created by using COBOL CGI subroutines can be altered into a web application under IIS with a few modifications.

Chapter 2 Creating a Web Application Using ISAPI Subroutines

2.1 What is Required

The following lists what is required to create and execute an ISAPI application using ISAPI Subroutines.

IIS

Custom settings to run Web applications are needed.

NetCOBOL

Needed for compilation, link, and execution of Web applications.

For execution, the NetCOBOL server operation package is required.

For details regarding the installation and settings, consult the system administrator of the WWW Server to be used.

2.2 What to Create

To create an ISAPI application using SAPI Subroutines, the following minimum requirements must be met:

- Programs with the following three entry names and an ISAPI application (DLL) created with these functions as export functions
 - GetExtensionVersion
Function executed when DLL is loaded. This function is used for initialization.
 - HttpExtensionProc
Application executed for each request. ISAPI Subroutines can be used only in programs with this entry name.
 - TerminateExtension (created as required)
Executed when the DLL is unloaded. This program used for termination processing.
For information regarding compiling and linking these programs, see " - "Compiling and Linking" in Chapter 4.
- HTML document
 - Web page for invoking an application (for Web application activation)
 - Web page for processing result (Used for processing resulting output. This page is created as required)

The following explains each of the above functions and pages.

2.2.1 GetExtensionVersion

GetExtensionVersion is called only once when an ISAPI application is loaded into IIS. The next HttpExtensionProc is called only after processing of the program with this entry name has completed. Therefore, this program can be used for initialization that is common to the ISAPI application to be created.

For example, GetExtensionVersion can be used for initializing external data. However, as explained later, the timing to perform termination processing is not definite and so do not use this function to perform processing that needs termination processing.

A program with this entry name must be created.

The following explains the precautions to be taken when creating a program with this entry name.

IDENTIFICATION DIVISION

Write the program name in the PROGRAM-ID paragraph as shown below:

```
PROGRAM-ID. "GetExtensionVersion".
```



Write the program name correctly, paying attention to the upper-case/lower-case characters, as this name is case sensitive, when used with ISAPI. If the program name is not correct, the application will not run.

ENVIRONMENT DIVISION

None

DATA DIVISION

Include the library file (copy file) to be used to interface with IIS in the LINKAGE SECTION using the COPY statement as follows:

```
LINKAGE SECTION.
  COPY ISAPIINF.
```

This library file (ISAPIINF.cbl) is stored in the folder in which COBOL is installed.

PROCEDURE DIVISION

Write the following to fit to the calling interface from IIS.

```
PROCEDURE DIVISION WITH STDCALL LINKAGE USING ISAPI-INFO.
```

- Return code

A return code to IIS when this function is completed needs to be set. The following two return code values are available, and the value "Normal end" is usually set.



Note that, if the value "Abnormal end" is set, the next HttpExtensionProc is not called.

Return code	Meaning
1	Normal end
0	Abnormal end

To set "Normal end", for example, enter as follows:

```
MOVE 1 TO PROGRAM-STATUS.
EXIT PROGRAM.
```

On the basis of the above, the sample of this program will look like the following:

```
IDENTIFICATION DIVISION
PROGRAM-ID. "GetExtensionVersion".
ENVIRONMENT DIVISION.
DATA DIVISION.
  LINKAGE SECTION
    COPY ISAPIINF.
PROCEDURE DIVISION WITH STDCALL LINKAGE USING ISAPI-INFO.
*
* Write processing as required.
*
  MOVE 1 TO PROGRAM-STATUS.
  EXIT PROGRAM.
```

2.2.2 HttpExtensionProc

The Web application called for each request from the WWW Browser. ISAPI Subroutines can only call a program with this entry name. A program with this entry name must always be created.

The following explains the precautions to be taken when creating a program with this entry name.

IDENTIFICATION DIVISION

Write the program name in the PROGRAM-ID paragraph as shown below:

```
PROGRAM-ID. "HttpExtensionProc".
```



Write the program name correctly, paying attention to the upper-case/lower-case characters. It is case sensitive when working with ISAPI. If the program name is not correct, the application will not run.

ENVIRONMENT DIVISION

None

DATA DIVISION

- WORKING-STORAGE SECTION

Include the library file to be used to interface with ISAPI Subroutines using the COPY statement as follows:

```
WORKING-STORAGE SECTION.  
COPY COBW3.
```

This library file (COBW3.cbl) is stored in the folder in which COBOL is installed.

- LINKAGE SECTION

Include the library file to be used to interface with IIS using the COPY statement as follows:

```
LINKAGE SECTION.  
COPY ISAPICTX.
```

This library file (ISAPICTX.cbl) is stored in the folder in which COBOL is installed.

PROCEDURE DIVISION

Write the following to fit to the calling interface from IIS.

```
PROCEDURE DIVISION WITH STDCALL LINKAGE USING ISAPI-CTX-CNT.
```

Since ISAPI-CTX-CNT obtained in the above way is needed for ISAPI Subroutines to exchange data with IIS, set its pointer to COBW3-CONTEXT as shown below:

```
MOVE LOW-VALUE TO COBW3.  
MOVE FUNCTION ADDR(ISAPI-CTX-CNT) TO COBW3-CONTEXT.
```

- Return code

A return code to IIS when this function is completed needs to be set. The following two return code values are available, and "Normal end" is usually set. If "Abnormal end" is set, the WWW Server assumes that an error occurred in the relevant application and a message to that effect is sent to the WWW Browser by IIS.

Return code	Meaning
1	Normal end

4	Abnormal end
---	--------------

To set "Normal end", for example, enter as follows:

```
MOVE 1 TO PROGRAM-STATUS.
EXIT PROGRAM.
```

On the basis of the above, the sample of this program will look like the following:

```
IDENTIFICATION DIVISION.
PROGRAM-ID.      "HttpExtensionProc".
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
    COPY COBW3.
LINKAGE SECTION.
    COPY ISAPICTX.
PROCEDURE DIVISION WITH STDCALL LINKAGE USING ISAPI-CTX-CNT.
*
    MOVE LOW-VALUE TO COBW3.
    MOVE FUNCTION ADDR(ISAPI-CTX-CNT) TO COBW3-CONTEXT.
*
* Write processing as required.
*
    MOVE 1 TO PROGRAM-STATUS.
    EXIT PROGRAM.
```

2.2.3 TerminateExtension

This function is called only once by IIS when the DLL is unloaded and so can be used for termination processing that is common to Web applications. However, the timing to unload the DLL depends on each condition. The DLL is unloaded during termination processing or, in some cases, only by forced unloading. Therefore, this aspect should be considered carefully before writing initialization in the termination processing or GetExtensionVersion. The program with this entry name can be omitted.

The following explains the precautions to be taken when creating a program with this entry name.

IDENTIFICATION DIVISION

Write the program name in the PROGRAM-ID paragraph as shown below:

```
PROGRAM-ID. "TerminateExtension".
```



Note

Write the program name correctly, paying attention to the upper-case/lower-case characters. It is case sensitive when used with ISAPI. If the program name is not correct, this function will not be called.

ENVIRONMENT DIVISION

None

DATA DIVISION

Include the library file to be used in the interface with IIS in the linkage section using the COPY statement as follows:

```
LINKAGE SECTION.
    COPY ISAPIFLG.
```

This library file (ISAPIFLG.cbl) is stored in the folder in which COBOL is installed.

PROCEDURE DIVISION

Write the following to fit to the calling interface from IIS.

```
PROCEDURE DIVISION WITH STDCALL LINKAGE USING ISAPI-FLAG.
```

- Return code

A return code to IIS when this function is completed needs to be set. The following two return code values are available, and the "Normal end" is usually set. If the "Abnormal end" is set, the WWW Server assumes that an error occurred in the relevant application.

Return code	Meaning
1	Normal end
0	Abnormal end

To set the normal end, for example, enter as follows:

```
MOVE 1 TO PROGRAM-STATUS.  
EXIT PROGRAM.
```

The sample of this program will look like the following:

```
IDENTIFICATION DIVISION.  
PROGRAM-ID. "TerminateExtension".  
ENVIRONMENT DIVISION.  
DATA DIVISION.  
LINKAGE SECTION.  
COPY ISAPIFLG.  
PROCEDURE DIVISION WITH STDCALL LINKAGE USING ISAPI-FLAG.  
*  
* Write processing as required.  
*  
MOVE 1 TO PROGRAM-STATUS.  
EXIT PROGRAM.
```

2.2.4 Web Page for Invoking AN Application

The Web page for invoking an application is an HTML document used for activating Web applications. Use the <FORM> tag and <A> tag to activate a Web application.

The created dynamic link library ISAPI.DLL is executed with METHOD="POST"

```
<FORM METHOD="POST" ACTION=" ISAPI.DLL ">
```



Note

When a Web application is activated using the <A> tag, Web parameters cannot be passed.

For other topics regarding HTML documents, see " *NetCOBOL Web Guide Appendix A*".

2.2.5 Web Page for Processing the Resulting Output

The Web page for processing the resulting output is an HTML document used to return execution results of the Web application to the WWW Browser. Since response data can also be returned directly from a Web application, the Web page for processing result output need not necessarily be prepared. However, by preparing the Web page for processing the resulting output, flexibility and maintainability can be improved; the independence of data to be returned and programs are enhanced and the layout of the output results can be changed without re-compiling the Web application again.

It is also possible to output the contents of a Web page for processing the resulting output by converting it dynamically.

To output a static HTML document (an HTML document whose output does not depend on processing results), HTML creation tools can directly be used.

To output a dynamic HTML document (an HTML document whose output depends on processing results), data can be replaced dynamically for output by specifying an item name (conversion name) enclosed by `"//COBOL//"` in the Web page used for processing the resulting output. Since all data contained in an HTML document is intended for conversion, it is also possible to replace the tags to change characters in the Web page for processing the resulting output and to specify the background color. See `"COBW3_PUT_HTML"` in *Chapter 3*.

The Web page for processing the resulting output can also be output by dividing the title/text/header into multiple HTML documents or by combining it with plain text.

2.3 Precautions to Take When Creating an ISAPI Application Using COBOL

Note the following points:

- Do not use the following functions for screen operations in COBOL programs to be used in Web applications. For details, see *"Programs Running under a Service"* in the *"NetCOBOL User's Guide"*.
 - Presentation file function (window handling function)
 - Screen handling function
 - ACCEPT/DISPLAY function

(However, the operational functions regarding environmental variables, date, and time can be used)

- ISAPI Subroutines can be used only within a program with the entry name: `HttpExtensionProc`.
- Before using ISAPI Subroutines, be sure to obtain (`FUNCTION ADDR`) the pointer, which is provided by the address of the group item (`ISAPI-CTX-CNT`) and set the value of `COBW3-CONTEXT` to it.
- Do not share `COBW3`, `ISAPI-CTX-CNT`, or its pointer data among threads, or IIS may not work properly.
- Do not use the `REPLACING` statement in the `COPY` statement that includes a library file (`COBW3.cbl`, `ISAPICTX.cbl`, `ISAPIINF.cbl`, and `ISAPIFLG.cbl`) provided by the ISAPI Subroutines. Do not specify a `REPLACE` statement so that a `COPY` statement would be replaced either.
- When data is initialized (such as the `VALUE` clause) in the `DATA DIVISION`, depending on the operational environment, initial values extending over multiple requests may not be guaranteed. If the initial values should be guaranteed for each request from the WWW Browser, be sure to initialize data in the Procedure Division to be executed.
- A web application is an application that runs under the services provided by the Web server. For applications running under services, please refer to *"21.1.3 Programs running under a Service"* of the *NetCOBOL User's Guide*.

2.4 Using the Web Application Development Support Function

The COBOL Project Manager provides the Web application development support function that assists you in developing Web applications.

COBOL programs using ISAPI Subroutines can easily be created by using the Web Application Wizard of the Project Manager. Resources such as COBOL programs using ISAPI Subroutines can automatically be created with the help of the Web Application Wizard. In addition, using the project relationship with the Project Manager, the created program can be compiled, linked, and maintained more easily.

For details of the Web application development support function, see *"NetCOBOL Web Development Tools Guide"*.

Chapter 3 How to Use the ISAPI Subroutines

3.1 Functions of the ISAPI Subroutines

ISAPI Subroutines provide the following functions.

3.1.1 Basic Functions

The basic functions enable you to create Web applications that perform business in response to requests from WWW Browsers.

Table 3.1 ISAPI Subroutines (COBW3_xxxxx)

Function	Subroutine name	Usage
Work environment settings and Web parameter acquisition	COBW3_INIT	Work environment settings of ISAPI Subroutines and acquisition of Web parameters
Operation of Web parameters	COBW3_GET_VALUE COBW3_GET_VALUE_XX COBW3_GET_VALUE_NX COBW3_GET_VALUE_XN COBW3_GET_VALUE_NN	Search of NAME in the Web parameters and acquisition of associated VALUE
	COBW3_CHECK_VALUE COBW3_CHECK_VALUE_X COBW3_CHECK_VALUE_N	Search of VALUE in the Web parameters
Output of processing results	COBW3_PUT_HEAD	HTTP Header output
	COBW3_PUT_HTML	Output of the Web page for processing result (HTTP Contents)
	COBW3_SET_CNV COBW3_SET_CNV_XX COBW3_SET_CNV_NX COBW3_SET_CNV_XN COBW3_SET_CNV_NN	Registration of converted data in the Web page for processing result
	COBW3_DEL_CNV COBW3_DEL_CNV_X COBW3_DEL_CNV_N	Deletion of converted data in the Web page for processing result
Output of processing results	COBW3_INIT_CNV	Initialization of converted data in the Web page for processing result
	COBW3_SET_REPEAT COBW3_SET_REPEAT_XX COBW3_SET_REPEAT_NX COBW3_SET_REPEAT_XN COBW3_SET_REPEAT_NN	Registration of repeatedly converted data in the Web page for processing result
	COBW3_DEL_REPEAT COBW3_DEL_REPEAT_X COBW3_DEL_REPEAT_N	Deletion of repeatedly converted data in the Web page for processing result

	COBW3_INIT_REPEAT	Initialization of repeatedly converted data in the Web page for processing result
	COBW3_PUT_TEXT	Data output (addition of the line feed code)
System command execution	COBW3_SYSTEM	Execution of the specified system command
Release of ISAPI execution environment resources	COBW3_FREE	Release of resources acquired by the ISAPI Subroutines
Request information acquisition	COBW3_RECEIVE_HEADER	Acquisition of the HTTP header
	COBW3_GET_REQUEST_INFO	Acquisition of information about the requests
	COBW3_GET_AUTHORIZE	Acquisition of authentication information
Cookie data operation	COBW3_SET_COOKIE COBW3_SET_COOKIE_XX COBW3_SET_COOKIE_NX COBW3_SET_COOKIE_XN COBW3_SET_COOKIE_NN	Registration of Cookie data
	COBW3_DEL_COOKIE COBW3_DEL_COOKIE_X COBW3_DEL_COOKIE_N	Deletion of registered Cookie data
	COBW3_INIT_COOKIE	Initialization of registered Cookie data
	COBW3_GET_COOKIE COBW3_GET_COOKIE_XX COBW3_GET_COOKIE_NX COBW3_GET_COOKIE_XN COBW3_GET_COOKIE_NN	Acquisition of Cookie data from Web parameters

- It is necessary to use the following subroutines in the ASCII environment.

- COBW3_GET_VALUE
- COBW3_CHECK_VALUE
- COBW3_SET_CNV
- COBW3_DEL_CNV
- COBW3_SET_REPEAT
- COBW3_DEL_REPEAT
- COBW3_SET_COOKIE
- COBW3_DEL_COOKIE
- COBW3_GET_COOKIE

- It is necessary to use subroutines such as COBW3_XXXXX_XX and COBW3_XXXXX_N in the Unicode environment. Moreover, the meaning is as follows.

Subroutine name	Description
COBW3_XXXXX_XX	Both the NAME/VALUE parameters are assumed to be alphanumeric items.
COBW3_XXXXX_NX	The NAME parameter is assumed to be a national data item and the VALUE parameter is assumed to be an alphanumeric data item.

Subroutine name	Description
COBW3_XXXX_XN	The NAME parameter is assumed to be an alphanumeric data item and the VALUE parameter is assumed to be a national data item.
COBW3_XXXX_NN	Both the NAME/VALUE parameters are assumed to be national data items.
COBW3_XXXX_X	Input parameters are assumed to be alphanumeric data items.
COBW3_XXXX_N	Input parameters are assumed to be national data items.

- Any code system can use the subroutines not listed above.
- The operation of the following subroutines provided up to this time is guaranteed only in environments where Web applications run using the ASCII code system. For development of applications that run under any other code system than ASCII, use the alternative subroutines in the following table.

Subroutine name	Alternative subroutine
COBW3_NAME	COBW3_GET_VALUE_XX COBW3_GET_VALUE_NX COBW3_GET_VALUE_XN COBW3_GET_VALUE_NN
COBW3_VALUE	COBW3_CHECK_VALUE_X COBW3_CHECK_VALUE_N
COBW3_CNV_SET	COBW3_SET_CNV_XX COBW3_SET_CNV_NX COBW3_SET_CNV_XN COBW3_SET_CNV_NN
COBW3_CNV_DEL	COBW3_DEL_CNV_X COBW3_DEL_CNV_N
COBW3_CNV_INIT	COBW3_INIT_CNV

3.1.2 File Upload Function

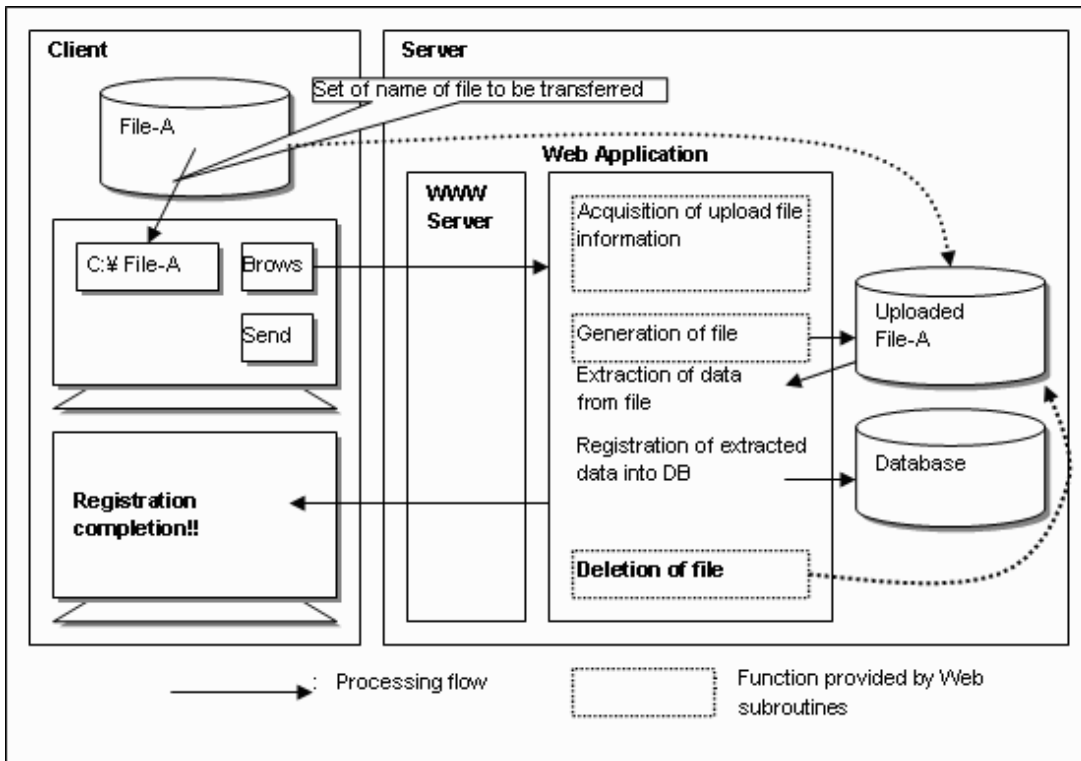
The file upload function transfers client-side-generated data (files) to the WWW Server.

This function is valid in the following cases:

- When a large amount of data that may cause timer expiration in online processing is to be entered.
- When a mobile environment with meter-rate accounting is to be used.
- When a data entry tool is already prepared at the client side and WWW Server linkage using the tool output data (files) is requested instead of a request to change the data entry tool into a Web tool.
- When only the HTTP ports can be accessed in file transfer because of security conditions. (File transfer in conventional systems is done using FTP.)

Using the file upload function, file transfer can be associated with applications. Therefore, data can be checked and registered immediately after file transfer.

The following conceptual chart of processing which uses the file upload function is shown below:



Web page for invoking the applications

```

:
<FORM METHOD="POST"
ENCTYPE="multi-part/form-data"
ACTION="sample/action.scr">
<P>
Send file:
<INPUT TYPE="file"
NAME="UPFILE"><BR>
<INPUT TYPE="submit"
VALUE="Send">
</FORM>
:

```

Web application (COBOL source)

```

:
WORKING-STORAGE SECTION.
COPY COBW3.
PROCEDURE DIVISION.
*
CALL "COBW3_INIT" USING COBW3.
*
*GET INFORMATION OF UPLOADED FILE
MOVE "UPFILE" TO
COBW3-SEARCH-DATA.
CALL "COBW3_GET_UPLOADFILE_INFO"
USING COBW3.
IF COBW3-SEARCH-FLAG-NON THEN
Error Processing
END-IF.
*SET GENERATION FILE-NAME
MOVE "File-A" TO
COBW3-UPLOADED-FILENAME.
CALL "COBW3_GEN_UPLOADFILE"
USING COBW3.
IF COBW3-STATUS NOT = ZERO THEN
Error Processing
END-IF.
*****
*
* Any processing for uploaded files
*
*****
*DELETION UPLOADED FILES
CALL "COBW3_DEL_UPLOADFILE"
USING COBW3.
CALL "COBW3_PUT_HTML"

```

Web subroutines provide the following functions to easily create such application programs:

Function	Subroutine name	Usage
File upload	COBW3_GET_UPLOADFILE_INFO COBW3_GET_UPLOADFILE_INFO_X COBW3_GET_UPLOADFILE_INFO_N	Acquires the information on an upload file.
	COBW3_GEN_UPLOADFILE COBW3_GEN_UPLOADFILE_X COBW3_GEN_UPLOADFILE_N	Generates an uploaded file on the WWW Server.
	COBW3_DEL_UPLOADEDFILE	Deletes an uploaded file.

To upload a file, the following values must be written on the Web page for invoking an application:

Tag name	Attributes	Value
FORM	METHOD	POST
	ENCTYPE	multipart/form-data
INPUT	TYPE	file

For details of these tags and attributes, see the "*NetCOBOL Web Guide*" and other documentation resources that explain HTML.

The above tags can be used together with the INPUT tag of other types (that is, other than "file") in the same FORM. The data that corresponds to these INPUT tags can be acquired by a subroutine such as COBW3_GET_VALUE regardless of whether the file upload function is used.

Note

- In this Web subroutines, size of the uploaded files should be 999,999,999 bytes or less. Even if it is a file of 999,999,999 bytes, data cannot occasionally be received according to WWW Server.
- The NAME value specified in the INPUT tag (TYPE="file") for uploading files cannot be used in the INPUT tag of other types (that is, other than "file"). Otherwise, the normal operation of the Web subroutine is not assured.

Example

```

:
<INPUT TYPE="text" NAME="FILE1">
<INPUT TYPE="file" NAME="FILE1" & Do not specify
:

```

3.1.3 Session Management Function

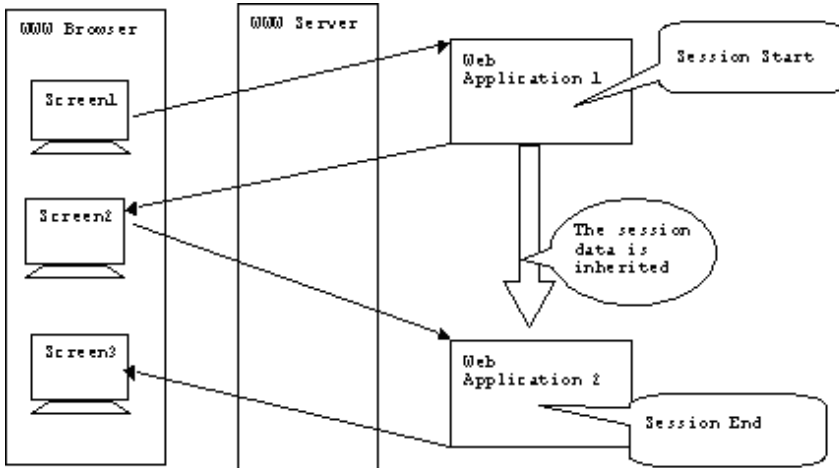
By using the session management function, a Web application that performs business extending over multiple requests from a specific client (WWW Browser) can be created.

Function	Subroutine name	Usage
Session management	COBW3_START_SESSION	Session start
	COBW3_END_SESSION	Session end
	COBW3_SET_SESSION_DATA	Registration of session data
	COBW3_GET_SESSION_DATA	Acquisition of session data

Function	Subroutine name	Usage
	COBW3_ALTER_SESSION_TIMEOUT	Change of the session timeout period
	COBW3_GET_SESSION_INFO	Acquisition of current session information

The session management function cannot be used in the COBOL CGI subroutines.

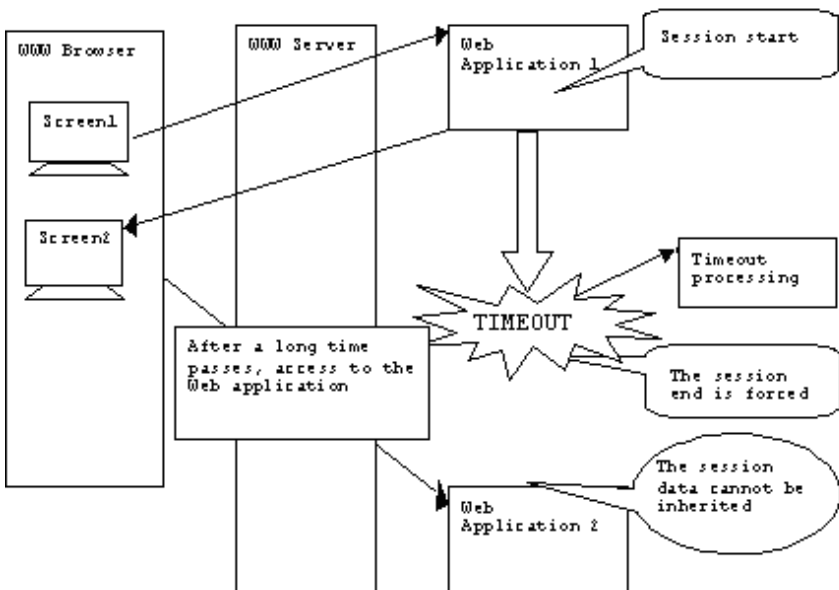
By opening a session using the session management function, connection to a specific client (WWW Browser) can be maintained. Since, within the same session, data entered in the previous page can be inherited by the next page using the session data, business programs in which multiple screens (pages) appear can easily be created.



If business is interrupted because the operator leaves his or her seat too long during screen operation or the WWW Browser is closed while a session is in progress, a timeout of the session occurs.

Timeout means the time elapsed within the same session between the instant when a response is returned from the Web application to the WWW Browser and the instant when the next request is issued by the WWW Browser reaches the specified time interval.

When a timeout occurs, the session end is forced by the session management. Thus, if a timeout occurs, resources of Web applications may remain in a state in which the session is suspended. The session management provides, as a countermeasure to handle such conditions, the registration features of timeout processing.





If a Web application using sessions is activated doubly by double-clicking, for example, the SUBMIT button (button specifying "SUBMIT" as the TYPE attribute of the INPUT tag), the operation of the Web application is undefined. Thus, it is necessary to inhibit double activation using, for example, JavaScript on the WWW Browser. For general precautions to be taken when operating the WWW Browser, see "NetCOBOL Web Guide".

3.2 ISAPI Subroutines Interface

This section explains the interface for using ISAPI Subroutines.

How to write

Explains how to write code to call ISAPI Subroutines.

Data setting for calling

Provides information to be set before calling an ISAPI Subroutine.

If a condition-name is written, sets a value to the condition name using the SET statement.

An underlined condition name indicates that its value is the initial value.

The data value set when an ISAPI Subroutine is called does not change before and after the call.

Processing result data

Explains the processing results obtained by executing an ISAPI Subroutine.

If a condition-name is written, evaluate data based on the condition-name conditions.

For errors and warnings of ISAPI Subroutines, see COBW3-STATUS.

Value	Meaning
0	Normal end
Other than 0	Error No.

For the error No., see "Appendix B Error Handling".

For the execution results of an ISAPI Subroutine, see PROGRAM-STATUS.

Value	Meaning
0	Normal end
-1	Abnormal end



If, before calling a subroutine, processing to change an area other than data made public by the data settings when the subroutine called is performed, operations and contents of the area are not guaranteed.

Before calling "COBW3_INIT", initialize COBW3 of the interface area using LOW-VALUE.

3.2.1 Library Files

Library files are provided as the interface for the calling parameters of ISAPI Subroutines and return parameters after execution. To call an ISAPI Subroutine, include the following library file in the WORKING-STORAGE SECTION using the COPY statement. This library file is stored in the folder in which COBOL is installed.

Library name

COBW3.cbl

How to write

Write as shown below:

```
WORKING-STORAGE SECTION.  
    COPY COBW3.
```



Do not use the REPLACING specification of COPY statement that includes the library of ISAPI Subroutines or a REPLACE statement in which this COPY statement is to be replaced.

If the library is changed, operations are not guaranteed.

3.2.2 Calling ISAPI Subroutines

It is necessary to call first "COBW3_INIT" to set the ISAPI Subroutine environment and to acquire Web parameters. When terminating ISAPI Subroutine processing (at the end of the procedure), be sure to call "COBW3_FREE" to release resources allocated by ISAPI Subroutines.

Call other subroutines using the CALL statement as needed.

When calling other subroutines in a program different from the program that called "COBW3_INIT", pass the interface area (COBW3) defined by the program that called "COBW3_INIT" to other programs that need to make ISAPI Subroutine calls and call other subroutines using the same area.



When "COBW3_INIT" is called by the parent program and "COBW3_FREE" is called by the child program B.

Parent program:

```
IDENTIFICATION DIVISION.  
PROGRAM-ID.      "HttpExtensionProc".  
DATA             DIVISION.  
WORKING-STORAGE SECTION.  
    COPY COBW3.  
LINKAGE          SECTION.  
    COPY ISAPICTX.  
PROCEDURE DIVISION WITH STDCALL LINKAGE USING ISAPI-CTX-CNT.  
    MOVE LOW-VALUE TO COBW3.  
    MOVE FUNCTION ADDR(ISAPI-CTX-CNT) TO COBW3-CONTEXT.  
    CALL "COBW3_INIT" USING COBW3.  
    :  
    CALL "B"          USING COBW3.  
*  
    MOVE 1 TO PROGRAM-STATUS.  
    EXIT PROGRAM.
```

Child program:

```
IDENTIFICATION DIVISION.  
PROGRAM-ID.      B.  
DATA             DIVISION.  
LINKAGE          SECTION.  
    COPY COBW3.  
PROCEDURE DIVISION USING COBW3.  
    :  
    :
```

```
CALL "COBW3_FREE" USING COBW3.  
EXIT PROGRAM.
```

3.3 Set the Work Environment and Acquire Web Parameters

3.3.1 COBW3_INIT

This subroutine sets the work environment of ISAPI Subroutines. It also acquires Web parameters and sets them to the work area used by ISAPI Subroutines.



Note

Before calling "COBW3_INIT", initialize COBW3 of the interface area using LOW-VALUE.

How to write

```
CALL "COBW3_INIT" USING COBW3.
```

Data setting for calling

COBW3-CONTEXT [required]

Specify the pointer to the interface area with IIS. Never change the specified value.

COBW3-DMODE [optional]

Specify the output of error messages of ISAPI Subroutines to the WWW Browser.

COBW3-DMODE is valid only when the operation code system is ASCII.

Condition name	Value	Meaning
COBW3-DMODE-NODBG	LOW-VALUE	Do not output error messages.
COBW3-DMODE-DBG	"1"	Output error messages to the WWW Browser.

Processing result data

None



Note

If "COBW3_INIT" is called twice or more in succession, acquired Web parameters may be re-initialized and thus the subsequent operations are not guaranteed. Therefore, it is necessary to create a program so that "COBW3_INIT" is always called once before another ISAPI Subroutine is called each time the WWW Browser is connected.

3.4 Manipulate Web Parameters

3.4.1 COBW3_GET_VALUE, COBW3_GET_VALUE_XX, COBW3_GET_VALUE_NX, COBW3_GET_VALUE_XN, and COBW3_GET_VALUE_NN

These subroutines search any name (NAME) from Web parameters obtained by "COBW3_INIT" to acquire the corresponding value (VALUE). The following lists the meaning of each subroutine:

ASCII environment

COBW3_GET_VALUE

Searches the alphanumeric character string name (NAME) to return the corresponding value (VALUE) as an alphanumeric character string.

Unicode environment

COBW3_GET_VALUE_XX

Searches the alphanumeric character string name (NAME) to return the corresponding value (VALUE) as an alphanumeric character string.

COBW3_GET_VALUE_NX

Searches the national character string name (NAME) to return the corresponding value (VALUE) as an alphanumeric character string.

COBW3_GET_VALUE_XN

Searches the alphanumeric character string name (NAME) to return the corresponding value (VALUE) as a national character string.

COBW3_GET_VALUE_NN

Searches the national character string name (NAME) to return the corresponding value (VALUE) as a national character string.

The padding string (padding character) used when the acquired value (VALUE) is blank.

How to write

```
CALL "COBW3_GET_VALUE" USING COBW3.
```

```
CALL "COBW3_GET_VALUE_XX" USING COBW3.
```

```
CALL "COBW3_GET_VALUE_NX" USING COBW3.
```

```
CALL "COBW3_GET_VALUE_XN" USING COBW3.
```

```
CALL "COBW3_GET_VALUE_NN" USING COBW3.
```

Data setting for calling

COBW3-SEARCH-DATA and COBW3-SEARCH-DATA-N

Set the name (NAME) to be searched (The name here is that specified in name of the HTML document of the Web page for the invoking application).

For COBW3_GET_VALUE, COBW3_GET_VALUE_XX and COBW3_GET_VALUE_XN, set the name to COBW3-SEARCH-DATA.

For COBW3_GET_VALUE_NX and COBW3_GET_VALUE_NN, set the name to COBW3-SEARCH-DATA-N.

COBW3-SEARCH-LENGTH [optional]

If the name has a valid blank at the end, set the string length (byte length) of the name including the blank.

Value	Meaning
0	Searches the name using the length up to the last character excluding the blank.
1 to 1024	Searches the name using the specified string length.

COBW3-NUMBER [optional]

If multiple entities with the same name (NAME) exist in Web parameters, set the order of appearance of names to be searched.

Condition name	Value	Meaning
COBW3-NUMBER-INIT	1	Searches the name that matches first.

Condition name	Value	Meaning
-	2 to 9999	Searches the names in the specified order of appearance.

Processing result data

COBW3-SEARCH-FLAG

Condition name	Value	Meaning
COBW3-SEARCH-FLAG-NON	"0"	The name to be searched does not exist.
COBW3-SEARCH-FLAG-EXIST	"1"	The name to be searched exists.

COBW3-GET-DATA and COBW3-GET-DATA-N

The value (VALUE) corresponding to the name (NAME) to be searched is set.

For COBW3_GET_VALUE, COBW3_GET_VALUE_XX and COBW3_GET_VALUE_NX, the value is set to COBW3-GET-DATA.

For COBW3_GET_VALUE_XN and COBW3_GET_VALUE_NN, the value is set to COBW3-GET-DATA-N.

COBW3-GET-LENGTH

The string length (byte length) of the value (VALUE) corresponding to the name (NAME) to be searched is set.

3.4.2 COBW3_CHECK_VALUE, COBW3_CHECK_VALUE_X and COBW3_CHECK_VALUE_N

These subroutines search any value (VALUE) from Web parameters obtained by "COBW3_INIT". This subroutine is used, for example, to evaluate the items checked by the WWW Browser based on the value (VALUE) when activated from a Web page for the invoking application with the check boxes. The following lists the meaning of each subroutine:

ASCII environment

COBW3_CHECK_VALUE

Searches the value (VALUE) of an alphanumeric character string.

Unicode environment

COBW3_CHECK_VALUE_X

Searches the value (VALUE) of an alphanumeric character string.

COBW3_CHECK_VALUE_N

Searches the value (VALUE) of a national character string.

How to write

```
CALL "COBW3_CHECK_VALUE" USING COBW3.
```

```
CALL "COBW3_CHECK_VALUE_X" USING COBW3.
```

```
CALL "COBW3_CHECK_VALUE_N" USING COBW3.
```

Data setting for calling

COBW3-SEARCH-DATA and COBW3-SEARCH-DATA-N

Set the value (VALUE) to be searched.

For COBW3_CHECK_VALUE and COBW3_CHECK_VALUE_X, set the value to COBW3-SEARCH-DATA.

For COBW3_CHECK_VALUE_N, set the name to COBW3-SEARCH-DATA-N.

COBW3-SEARCH-LENGTH [optional]

If the value (VALUE) has a valid blank at the end, set the string length (byte length) of the value including the blank.

Value	Meaning
0	Searches the value using the length up to the last character excluding the blank.
1 to 1024	Searches the value using the specified string length.

COBW3-NUMBER [optional]

Sets the order of appearance of VALUES to be searched when there are multiple VALUES with the same value in the Web parameters.

Condition name	Value	Meaning
COBW3-NUMBER-INIT	1	Searches the value that matches first.
-	2 to 9999	Searches the values in the specified order of appearance.

Processing result data

COBW3-SEARCH-FLAG

Condition name	Value	Meaning
COBW3-SEARCH-FLAG-NON	"0"	The value to be searched does not exist.
COBW3-SEARCH-FLAG-EXIST	"1"	The value to be searched exists.

3.5 Output Processing Results

3.5.1 COBW3_PUT_HEAD

This subroutine outputs the specified optional HTTP header.

When outputting an HTML document of a Web page for processing result output, there is no need to output the header using this subroutine because the header is output as text/html by ISAPI Subroutines. Use this subroutine only when files (such as plain text) other than HTML documents are output or any HTTP header should be output to the WWW Browser.



Use this subroutine before using COBW3_PUT_HTML or COBW3_PUT_TEXT. If this subroutine is used after COBW3_PUT_HTML or COBW3_PUT_TEXT is used even once, the specified information will become invalid.

If any value other than the default value is set to COBW3-CONTENT-TYPE or COBW3-STATUS-CODE, do not change the value.

How to write

```
CALL "COBW3_PUT_HEAD" USING COBW3.
```

Data setting for calling

COBW3-PUT-HEAD

Set the string to be output to the HTTP header.

COBW3-PUT-HEAD-LENGTH [optional]

If the string has a valid blank at the end, set the length (byte length) of the string including the blank.

Value	Meaning
0	Outputs the string using the length up to the last character excluding the blank. However, if COBW3-PUT-HEAD is completely blank, the string length is set to zero (line feed only) for processing.
1 to 512	Outputs the string using the specified string length.

COBW3-CONTENT-TYPE [optional]

Set the file type (attribute) (Content-type) of response data. If this subroutine is called multiple times, the first specification becomes valid.

Condition name	Value	Meaning
-	LOW-VALUE	Outputs an HTML document.
COBW3-CONTENT-TYPE-NON	HIGH-VALUE	Do not output the Content-type.
COBW3-CONTENT-TYPE-HTML	"text/html"	Outputs an HTML document.
COBW3-CONTENT-TYPE-TEXT	"text/plain"	Outputs a text file.
-	Any string	Specifies any string to the Content-type of the HTTP header.

COBW3-STATUS-CODE [optional]

Set the status code (Status-code). If this subroutine is called multiple times, the first specification becomes valid.

Condition name	Value	Meaning
-	LOW-VALUE	Outputs the normal end code "200" to the Status-code.
COBW3-STATUS-CODE-NON	HIGH-VALUE	Do not output the Status-code.
COBW3-STATUS-CODE-200	"200"	Outputs the normal end code "200" to the Status-code.
-	Any code	Outputs any Status-code.



Note

Avoid the use of COBW3-CONTENT-TYPE-NON and COBW3-STATUS-CODE-NON where possible. In CGI, the specification of -NON is valid because the Content-type can be specified from a batch file. However, in DLL applications such as Internet Server Applications, -NON cannot be specified from outside.

Also in Internet Server Applications, if -NON is specified and the Content-type or Status-code is not specified, the operation is undefined.

3.5.2 COBW3_PUT_HTML

This subroutine outputs the Web page for processing result output (HTML document) to the WWW Browser after execution of the Web application. If, at this point, there is any conversion name enclosed "//COBOL//" or "//COBOL_RERAET//" in the Web page for processing result output to be output, the conversion name is converted into the conversion string registered in, such as COBW3_SET_CNV or COBW3_SET_REPEAT in advance.

How to write

```
CALL "COBW3_PUT_HTML" USING COBW3.
```

Data setting for calling

COBW3-HTML-FILENAME

Set the file name of the Web page for processing result output.

Processing result data

None

How to specify the conversion name in the Web page for processing result output:

Enclose the string (conversion name) in the Web page for processing result output to be converted for output during execution of the Web application with "//COBOL//".



Note

"//COBOL//" cannot be specified extending over multiple lines.

```
× × × //COBOL//conversion name//COBOL// × × ×
```

The following table lists the behaviors depending on how the conversion name is specified.

How the conversion name is specified	Processing description
This is a //COBOL//NAME//COBOL// test.	If the conversion name "NAME" is registered, "//COBOL//NAME//COBOL//" is replaced by the registered conversion string.
This is a //COBOL// test.	Since the conversion name specification is not correct, an error occurs.
This is a //COBOL test.	Output as it is.
This is a //COBOL// NAME//COBOL// test.	Since the conversion name specification is not correct, an error occurs.
This is a //COBOL// NAME//COBOL	Since the conversion name specification is not correct, an error occurs.
This is a //COBOL////COBOL//	Since the conversion name specification is not correct, an error occurs.
This is a //cobol//NAME//cobol//.	Output as it is.



Example

a.htm (Web page for processing result output)

```
      :  
<BODY>  
  <BR>  
Name //COBOL//GET-NAME//COBOL//  
  <BR>  
      :
```

COBOL program

```
      :  
* Set the name.  
MOVE "GET-NAME" TO COBW3-CNV-NAME.  
MOVE "Fujitsu Taro" TO COBW3-CNV-VALUE.  
* Register conversion data in the Web page for processing result output.  
CALL "COBW3_SET_CNV" USING COBW3.  
      :  
* Set the file name of the Web page for processing result output.  
MOVE "a.htm" TO COBW3-HTML-FILENAME.  
* Output the Web page for processing result output.  
CALL "COBW3_PUT_HTML" USING COBW3.  
      :
```

Output results on the WWW Browser

```

      :
Name  Fujitsu Taro
      :

```

How to specify the repetitive range in the Web page for processing result output:

Specify the repetitive range by using "//COBOL_REPEAT_START//" and "//COBOL_REPEAT_END//" in the location of the Web page for processing result output to be repeated during execution of the Web application. The number of times of repetition is the same as the number of registrations of the repetitive conversion string to be converted.

- Repetitive range start

Write "//COBOL_REPEAT_START//" at the start location of the repetitive range of data.

- Repetitive item

Write data to be output repeatedly. Write conversion names in data where to be converted for output during execution of the Web application.

If a different conversion string should be output in each repetition, enclose the string (conversion name) with "//COBOL_REPEAT//".
If a fixed conversion string should be output in each repetition, enclose the string (conversion name) with "//COBOL//".

- Repetitive range end

Write "//COBOL_REPEAT_END//" at the end location of the repetitive range of data.

Be sure to register the conversion string corresponding to the conversion name enclosed by "//COBOL_REPEAT//" at least once.

If no conversion name enclosed by "//COBOL_REPEAT//" is written in the repetitive item, the number of times of repetition becomes one.

"//COBOL//" and "//COBOL_REPEAT//" cannot be specified extending over multiple lines.

The line containing "//COBOL_REPEAT_START//" or "//COBOL_REPEAT_END//" indicating the start/end of the repetitive range is deleted when output to the WWW Browser. Therefore, do not write tags related to the display in the same line.

The registered number must be the same for all repetitive conversion strings written in the repetitive range. If the registered numbers are different, an error occurs because there is a conversion name that does not have enough registered data for the number of times of repetition.

If the repetitive range start "//COBOL_REPEAT_START//" is not written, no repetitive output is performed and all code other than "//COBOL//" is regarded as data for output.

If the repetitive range end "//COBOL_REPEAT_END//" is not written, the Web page for processing result output is output to the end and then an error occurs.

```

//COBOL_REPEAT_START//
× × //COBOL_REPEAT//conversion name//COBOL_REPEAT//× ×
× × //COBOL//conversion name//COBOL//× ×
//COBOL_REPEAT_END//

```

The behavior depends on how the repetitive range is specified is the same as that depending on how the conversion name is specified in the Web page for processing result output.

Example

b.htm (Web page for processing result output)

```

      :
<TABLE>
  <TR>
    <TD>Name</TD>
    <TD>Age</TD>
    <TD>Company</TD>
  </TR>
  //COBOL_REPEAT_START//
  <TR>

```

```

<TD>//COBOL_REPEAT//GET-NAME//COBOL_REPEAT//</TD>
<TD>//COBOL_REPEAT//GET-AGE//COBOL_REPEAT//</TD>
<TD>//COBOL//GET-OFFICE//COBOL//</TD>
</TR>
//COBOL_REPEAT_END//
</TABLE>
:

```

COBOL program

```

:
PERFORM UNTIL END-FLAG="END"
* Input Contents
READ IN-FILE AT END MOVE "END" TO END-FLAG
* Set the name.
MOVE "GET-NAME" TO COBW3-CNV-NAME
MOVE name TO COBW3-CNV-VALUE
* Register repetitive conversion data in the Web page for processing result output.
CALL "COBW3_SET_REPEAT" USING COBW3
* Set the age.
MOVE "GET-AGE" TO COBW3-CNV-NAME
MOVE AGE TO COBW3-CNV-VALUE
* Register repetitive conversion data in the Web page for processing result output.
CALL "COBW3_SET_REPEAT" USING COBW3
:
END-PERFORM.
:
* Set the company name.
MOVE "GET-OFFICE" TO COBW3-CNV-NAME.
MOVE "Fujitsu" TO COBW3-CNV-VALUE.
* Register conversion data in the Web page for processing result output.
CALL "COBW3_SET_CNV" USING COBW3.
:
* Set the file name of the Web page for processing result output.
MOVE "b.htm" TO COBW3-HTML-FILENAME.
* Output the Web page for processing result output.
CALL "COBW3_PUT_HTML " USING COBW3.
:

```

Output results on the WWW Browser

```

:

```

Name	Age	Company
Suzuki	35	Fujitsu
Sato	26	Fujitsu
Tanaka	23	Fujitsu

```

:

```

3.5.3 COBW3_SET_CNV, COBW3_SET_CNV_XX, COBW3_SET_CNV_NX, COBW3_SET_CNV_XN, and COBW3_SET_CNV_NN

These subroutines register the conversion character string for the conversion name enclosed by "//COBOL//" specified in the Web page for processing result output to be output by "COBW3_PUT_HTML".

The registered information is referenced during execution of COBW3_PUT_HTML, and the conversion is carried out in the Web page for processing result output according to the registered conversion data.

The following lists the meaning of each subroutine:

ASCII environment

COBW3_SET_CNV

Registers the conversion character string corresponding to an alphanumeric character string conversion name as an alphanumeric character string.

Unicode environment

COBW3_SET_CNV_XX

Registers the conversion character string corresponding to an alphanumeric character string conversion name as an alphanumeric character string.

COBW3_SET_CNV_NX

Registers the conversion character string corresponding to a national character string conversion name as an alphanumeric character string.

COBW3_SET_CNV_XN

Registers the conversion character string corresponding to an alphanumeric character string conversion name as a national character string.

COBW3_SET_CNV_NN

Registers the conversion character string corresponding to a national character string conversion name as a national character string.



If multiple different conversion names are specified in the Web page for processing result output, code a separate call for each conversion name "COBW3_SET_CNV" or other subroutines to register the conversion character string.

How to write

```
CALL "COBW3_SET_CNV" USING COBW3 .
```

```
CALL "COBW3_SET_CNV_XX" USING COBW3 .
```

```
CALL "COBW3_SET_CNV_NX" USING COBW3 .
```

```
CALL "COBW3_SET_CNV_XN" USING COBW3 .
```

```
CALL "COBW3_SET_CNV_NN" USING COBW3 .
```

Data setting for calling

COBW3-CNV-NAME and COBW3-CNV-NAME-N

Set the conversion name to be converted.

For COBW3_SET_CNV, COBW3_SET_CNV_XX and COBW3_SET_CNV_XN, set the conversion name to COBW3-CNV-NAME.

For COBW3_SET_CNV_NX and COBW3_SET_CNV_NN, set the conversion name to COBW3-CNV-NAME-N.

COBW3-CNV-NAME-LENGTH [optional]

If the conversion name has a valid blank at the end, set the string length (byte length) of the conversion name including the blank.

Value	Meaning
0	Searches the conversion name using the length up to the last character excluding the blank. However, if COBW3-CNV-NAME or COBW3-CNV-NAME-N is completely blank, the string length is set to zero for processing.
1 to 30	Searches the conversion name using the specified string length.

COBW3-CNV-VALUE and COBW3-CNV-VALUE-N

Set the conversion results (conversion string).

For COBW3_SET_CNV, COBW3_SET_CNV_XX and COBW3_SET_CNV_NX, set the conversion string to COBW3-CNV-VALUE.

For COBW3_SET_CNV_XN and COBW3_SET_CNV_NN, set the conversion string to COBW3-CNV-VALUE-N.

COBW3-CNV-VALUE-LENGTH [optional]

If the conversion string has a valid blank at the end, set the string length (byte length) of the conversion character string including the blank.

Value	Meaning
0	Registers the conversion character string using the length up to the last character excluding the blank. However, if COBW3-CNV-VALUE or COBW3-CNV-VALUE-N is completely blank, the string length is set to zero for processing.
1 to 1024	Registers the conversion string using the specified string length.

COBW3-CNV-MODE [optional]

Set the conversion type.

Condition name	Value	Meaning
COBW3-CNV-MODE-ADDREP	LOW-VALUE	Adds conversion data if the specified conversion name has not been registered.
COBW3-CNV-MODE-REPLACE	"1"	Replaces conversion data. An error is output and no conversion is carried out if the specified conversion name has not been registered.
COBW3-CNV-MODE-ADD	"2"	Adds conversion data. If the specified conversion name has been registered, an error is output and no addition is made.

COBW3_SANITIZE_CNV [Optional]

If characters that are vulnerable to a cross site scripting attack are found in conversion data, those characters are automatically replaced. This process is referred to as "sanitizing".

For more details on cross site scripting, refer to Appendix P, Security, in the NetCOBOL User's Guide.

COBW3_SANITIZE_CNV is valid when either COBW3_SET_CNV_XX or COBW3_SET_CNV_NX is used. However, if the code set is Unicode, COBW3_SANITIZE_CNV is also valid when COBW3_SET_CNV_XN or COBW3_SET_CNV_NN is used.

Condition name	Value	Explanation
COBW3-SANITIZE-CNV-OFF	LOW-VALUE	Does not sanitize.
COBW3-SANITIZE-CNV-ON	"1"	Sanitize.

Note

The sanitization procedure replaces the five characters that are vulnerable to a cross site scripting attack (&, <, >, ", ') with the following escape characters:

Unicode environment

COBW3_DEL_CNV_X

Deletes conversion data registered for an alphanumeric character string conversion name.

COBW3_DEL_CNV_N

Deletes conversion data registered for a national character string conversion name.

How to write

```
CALL "COBW3_DEL_CNV" USING COBW3.
```

```
CALL "COBW3_DEL_CNV_X" USING COBW3.
```

```
CALL "COBW3_DEL_CNV_N" USING COBW3.
```

Data setting for calling

COBW3-CNV-NAME and COBW3-CNV-NAME-N

Set the conversion name to be deleted.

For COBW3_DEL_CNV and COBW3_DEL_CNV_X, set the conversion name to COBW3-CNV-NAME.

For COBW3_DEL_CNV_N, set the conversion name to COBW3-CNV-NAME-N.

COBW3-CNV-NAME-LENGTH [optional]

If the conversion name has a valid blank at the end, set the string length (byte length) of the conversion name including the blank.

Value	Meaning
0	Searches the conversion name using the length up to the last character excluding the blank. However, if COBW3-CNV-NAME or COBW3-CNV-NAME-N is completely blank, the string length is set to zero for processing.
1 to 30	Searches the conversion name using the specified string length.

Processing result data

None

3.5.5 COBW3_INIT_CNV

This subroutine initializes all conversion data registered by, for example, COBW3_SET_CNV.

How to write

```
CALL "COBW3_INIT_CNV" USING COBW3.
```

Data setting for calling

None

Processing result data

None

3.5.6 COBW3_SET_REPEAT, COBW3_SET_REPEAT_XX, COBW3_SET_REPEAT_NX, COBW3_SET_REPEAT_XN, and COBW3_SET_REPEAT_NN

These subroutines register the repetitive conversion result (conversion character string) for the conversion name enclosed by "// COBOL_REPEAT//" specified in the Web page for processing result output to be output by "COBW3_PUT_HTML".

The registered information is referenced during execution of COBW3_PUT_HTML, and the conversion name specified in the repetitive range of the Web page for processing result output is converted according to the number of registrations of the repetitive conversion result (conversion character string) registered for the same conversion name.

The following lists the meaning of each subroutine:

ASCII environment

COBW3_SET_REPEAT

Registers the conversion character string corresponding to an alphanumeric character string conversion name as an alphanumeric character string.

Unicode environment

COBW3_SET_REPEAT_XX

Registers the conversion character string corresponding to an alphanumeric character string conversion name as an alphanumeric character string.

COBW3_SET_REPEAT_NX

Registers the conversion character string corresponding to a national character string conversion name as an alphanumeric character string.

COBW3_SET_REPEAT_XN

Registers the conversion character string corresponding to an alphanumeric character string conversion name as a national character string.

COBW3_SET_REPEAT_NN

Registers the conversion character string corresponding to a national character string conversion name as a national character string.

If multiple different conversion names are specified in the Web page for processing result output, code a separate call for each conversion name "COBW3_SET_REPEAT" or other subroutines to register the repetitive conversion result (conversion character string).

The conversion character strings to be converted during repetitive conversion are converted according to the order in which they are registered by calling "COBW3_SET_REPEAT or other subroutines".

Register the repetitive conversion result (conversion character string) for the conversion names contained in the same repetitive range so that their numbers of registrations become the same.

How to write

```
CALL "COBW3_SET_REPEAT" USING COBW3.
```

```
CALL "COBW3_SET_REPEAT_XX" USING COBW3.
```

```
CALL "COBW3_SET_REPEAT_NX" USING COBW3.
```

```
CALL "COBW3_SET_REPEAT_XN" USING COBW3.
```

```
CALL "COBW3_SET_REPEAT_NN" USING COBW3.
```

Data setting for calling

COBW3-CNV-NAME and COBW3-CNV-NAME-N

Set the conversion name to be converted.

For COBW3_SET_REPEAT, COBW3_SET_REPEAT_XX and COBW3_SET_REPEAT_XN, set the conversion name to COBW3-CNV-NAME.

For COBW3_SET_REPEAT_NX or COBW3_SET_REPEAT_NN, set the conversion name to COBW3-CNV-NAME-N.

COBW3-CNV-NAME-LENGTH [optional]

If the conversion name has a valid blank at the end, set the string length (byte length) of the conversion name including the blank.

Value	Meaning
0	Registers the conversion name using the length up to the last character excluding the blank. However, if COBW3-CNV-NAME or COBW3-CNV-NAME-N is completely blank, the string length is set to zero for processing.
1 to 30	Registers the conversion name using the specified string length.

COBW3-CNV-VALUE and COBW3-CNV-VALUE-N

Set the conversion results (conversion character string).

For COBW3_SET_REPEAT, COBW3_SET_REPEAT_XX and COBW3_SET_REPEAT_NX, set the conversion character string to COBW3-CNV-VALUE.

For COBW3_SET_REPEAT_XN or COBW3_SET_REPEAT_NN, set the conversion string to COBW3-CNV-VALUE-N.

COBW3-CNV-VALUE-LENGTH [optional]

If the conversion character string has a valid blank at the end, set the string length (byte length) of the conversion value including the blank.

Value	Meaning
0	Registers the conversion value using the length up to the last character excluding the blank. However, if COBW3-CNV-VALUE or COBW3-CNV-VALUE-N is completely blank, the string length is set to zero for processing.
1 to 1024	Registers the conversion name using the specified string length.

To specify a blank in the conversion results, register repetitive conversion data by specifying blanks to the conversion string and zero to the conversion string length.

COBW3_SANITIZE_CNV [Optional]

If characters that are vulnerable to a cross site scripting attack are found in conversion data, those characters are automatically replaced. This process is referred to as "sanitizing".

For more details on cross site scripting, refer to Appendix P, Security, in the NetCOBOL User's Guide.

COBW3_SANITIZE_CNV is valid when either COBW3_SET_REPEAT_XX or COBW3_SET_REPEAT_NX is used. However, if the code set is Unicode, COBW3_SANITIZE_CNV is also valid when COBW3_SET_REPEAT_XN or COBW3_SET_REPEAT_NN is used.

Condition name	Value	Explanation
COBW3-SANITIZE-CNV-OFF	LOW-VALUE	Does not sanitize.
COBW3-SANITIZE-CNV-ON	"1"	Sanitize.



Note

The sanitization procedure replaces the five characters that are vulnerable to a cross site scripting attack (&, <, >, ", ') with the following escape characters:

ASCII environment

COBW3_DEL_REPEAT

Deletes repetitive conversion data registered for an alphanumeric character string conversion name.

Unicode environment

COBW3_DEL_REPEAT_X

Deletes repetitive conversion data registered for an alphanumeric character string conversion name.

COBW3_DEL_REPEAT_N

Deletes repetitive conversion data registered for a national character string conversion name.



If multiple pieces of repetitive conversion data are registered for the specified conversion name, all registered data is deleted.

How to write

```
CALL "COBW3_DEL_REPEAT" USING COBW3 .
```

```
CALL "COBW3_DEL_REPEAT_X" USING COBW3 .
```

```
CALL "COBW3_DEL_REPEAT_N" USING COBW3 .
```

Data setting for calling

COBW3-CNV-NAME and COBW3-CNV-NAME-N

Set the conversion name to be deleted.

For COBW3_DEL_REPEAT and COBW3_DEL_REPEAT_X, set the conversion name to COBW3-CNV-NAME.

For COBW3_DEL_REPEAT_N, set the conversion name to COBW3-CNV-NAME-N.

COBW3-CNV-NAME-LENGTH [optional]

If the conversion name has a valid blank at the end, set the string length (byte length) of the conversion name including the blank.

Value	Meaning
0	Searches the conversion name using the length up to the last character excluding the blank. However, if COBW3-CNV-NAME or COBW3-CNV-NAME-N is completely blank, the string length is set to zero for processing.
1 to 30	Searches the conversion name using the specified string length.

Processing result data

None

3.5.8 COBW3_INIT_REPEAT

This subroutine initializes all repetitive conversion data registered by, for example, COBW3_SET_REPEAT.

How to write

```
CALL "COBW3_INIT_REPEAT" USING COBW3 .
```

Data setting for calling

None

Processing result data

None

3.5.9 COBW3_PUT_TEXT

This subroutine outputs data obtained by adding the line feed code to the specified string to the WWW Browser.

How to write

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

Data setting for calling

COBW3-PUT-STRING

Set the string to be output.

COBW3-PUT-STRING-LENGTH [optional]

If the string has a valid blank at the end, set the string length (byte length) of the string including the blank.

Value	Meaning
0	Outputs the string using the length up to the last character excluding the blank. However, if COBW3-PUT-STRING is completely blank, the string length is set to zero (line feed only) for processing.
1 to 1024	Outputs the string using the specified string length.

Processing result data

None

How to output string data (including the line feed code) to the WWW Browser:

The following shows an example of writing a program to output "<CENTER>Thank you very much for your cooperation.</CENTER>" to the WWW Browser.



Example

COBOL program

```
      :  
000360* Set of strings to be output  
000370 MOVE "<CENTER> Thank you very much for your cooperation.  
      </CENTER>"  
000380                                TO COBW3-PUT-STRING.  
000390* Output string data.  
000400 CALL "COBW3_PUT_TEXT" USING COBW3.  
      :
```

Output results on the WWW Browser

```
      :  
Thank you very much for your cooperation.  
      :
```

3.6 Execute System Commands

3.6.1 COBW3_SYSTEM

This subroutine executes any system command from a Web application.

How to write

```
CALL "COBW3_SYSTEM" USING COBW3.
```

Data setting for calling

COBW3-SYSTEMINFO

Set the system command to be executed and its parameters.

Processing result data

None



Since there is always a danger of illegal intrusion into the system and system damage by an outsider, avoid using a design so that system commands can be executed directly from the client (WWW Browser), such as executing command characters input from the WWW Browser as they are in COBW3_SYSTEM.

3.7 Release ISAPI Execution Environment Resources

3.7.1 COBW3_FREE

This subroutine releases resources acquired by ISAPI Subroutines. Thus, be sure to call this subroutine at the end of a Web application.

How to write

```
CALL "COBW3_FREE" USING COBW3.
```

Data setting for calling

None

Processing result data

None

3.8 Acquire Request Information

3.8.1 COBW3_RECEIVE_HEADER

This subroutine acquires the HTTP header from a request.

How to write

```
CALL "COBW3_RECEIVE_HEADER" USING COBW3.
```


Data setting for calling

COBW3-HEADER-NAME

Specify the header name to be acquired.

COBW3-HEADER-NAME-LENGTH [optional]

If the header name has a valid blank at the end, set the string length (byte length) of the header name including the blank.

Value	Meaning
0	Searches the header name using the length up to the last character excluding the blank.
1 to 64	Searches the header name using the specified string length.

Since there is normally no header name with a blank at its end, specify 0.

Processing result data

COBW3-HEADER-VALUE

The header value corresponding to the requested header name is set.

COBW3-HEADER-VALUE-LENGTH

The string length (byte length) of the header value corresponding to the requested header name is set.

3.8.2 COBW3_GET_REQUEST_INFO

This subroutine acquires various information about a request.

How to write

```
CALL "COBW3_GET_REQUEST_INFO" USING COBW3.
```

Data setting for calling

COBW3-REQUEST-INFO-TYPE

Specify the type of information to be acquired.

Condition name	Value	Meaning
COBW3-URI	LOW-VALUE	Request URI
COBW3-URL	"1"	Request URL
COBW3-SERVER-TYPE (COWB3-LISTENERTYPE)	"2"	Server type
COBW3-VIRTUALPATH	"3"	Virtual directory name of a request
COBW3-PHYSICALPATH	"4"	Physical path name for the virtual directory of a request
COBW3-QUERYSTRING	"5"	Web parameter
COBW3-LANGUAGE	"6"	List of languages that can be received by the WWW Browser
COBW3-ENCODING	"7"	List of encoding systems that can be received by the WWW Browser
COBW3-REQMIMETYPE	"8"	MIME type of a request
COBW3-REQUEST-METHOD	"9"	Request method
COBW3-PATH-INFO	"A"	Extended path name sent from the WWW Browser
COBW3-PATH-TRANSLATED	"B"	Physical path name for an extended path
COBW3-REMOTE-ADDR	"C"	IP address of a client

Condition name	Value	Meaning
COBW3-REMOTE-HOST	"D"	Host name of a client
COBW3-GATEWAY-INTERFACE	"E"	Revision of the supported CGI
COBW3-SERVER-NAME	"F"	Host name of a WWW Server
COBW3-SERVER-PORT	"G"	Connected port number of a WWW Server
COBW3-SERVER-PROTOCOL	"H"	Protocol used by a WWW Server
COBW3-SERVER-PORT-SECURE	"I"	"1" is returned if secured communication is conducted. Otherwise, "0" is returned.



Note

The MIME type of a request cannot be acquired.

Processing result data

COBW3-REQUEST-INFO

The acquired request information is set.

COBW3-REQUEST-INFO-LENGTH

The string length (byte length) of the acquired request information is set.

3.8.3 COBW3_GET_AUTHORIZE

This subroutine acquires authentication information from the WWW Browser. The user ID, password, and IP address of the client passed in response to an authentication request are acquired.

How to write

```
CALL "COBW3_GET_AUTHORIZE" USING COBW3.
```

Data setting for calling

None

Processing result data

COBW3-USERID

The acquired user ID is set.

COBW3-USERID-LENGTH

The string length (byte length) of the user ID is set.

COBW3-PASSWORD

The acquired user password is set.

COBW3-PASSWORD-LENGTH

The string length (byte length) of the user password is set.

COBW3-IP-ADDRESS

The acquired IP address of the client is set.

COBW3-IP-ADDRESS-LENGTH

The string length (byte length) of the IP address is set.

COBW3-AUTH-TYPE

The acquired authentication type is set.

COBW3-AUTH-TYPE-LENGTH

The string length (byte length) of the authentication type is set.



With basic authentication, all information can be acquired and "Basic" is set as the authentication type. For anonymous authentication, no information other than the IP address can be acquired. When Microsoft encryption authentication is used, all information excluding the user password can be acquired and "NTLM" is set as the authentication type.

3.9 Manipulate Cookie Data

3.9.1 COBW3_SET_COOKIE, COBW3_SET_COOKIE_XX, COBW3_SET_COOKIE_NX, COBW3_SET_COOKIE_XN, and COBW3_SET_COOKIE_NN

These subroutines register Cookie data to be sent to the WWW Browser. The registered Cookie data is used during header output.

The following lists the meaning of each subroutine:

ASCII environment

COBW3_SET_COOKIE

Registers an alphanumeric character string Cookie name and the corresponding Cookie value as alphanumeric character strings.

Unicode environment

COBW3_SET_COOKIE_XX

Registers an alphanumeric character string Cookie name and the corresponding Cookie value as alphanumeric character strings.

COBW3_SET_COOKIE_NX

Registers a national character string Cookie name and the corresponding Cookie value as alphanumeric character strings.

COBW3_SET_COOKIE_XN

Registers an alphanumeric character string Cookie name and the corresponding Cookie value as national character string.

COBW3_SET_COOKIE_NN

Registers a national character string Cookie name and the corresponding Cookie value as national character string.



Use this subroutine before using COBW3_PUT_HTML or COBW3_PUT_TEXT. If this subroutine is used after COBW3_PUT_HTML or COBW3_PUT_TEXT is used even once, the specified information will become invalid.

The Cookie character code sent to the WWW Browser is always ASCII, regardless of the COBOL operation code system.

How to write

```
CALL "COBW3_SET_COOKIE" USING COBW3.
```

```
CALL "COBW3_SET_COOKIE_XX" USING COBW3.
```

```
CALL "COBW3_SET_COOKIE_NX" USING COBW3.
```

```
CALL "COBW3_SET_COOKIE_XN" USING COBW3.
```

```
CALL "COBW3_SET_COOKIE_NN" USING COBW3.
```

Data setting for calling

COBW3-COOKIE-NAME and COBW3-COOKIE-NAME-N

Set the COOKIE name to be registered.

For COBW3_SET_COOKIE, COBW3_SET_COOKIE_XX and COBW3_SET_COOKIE_XN, set the Cookie name to COBW3-COOKIE-NAME.

For COBW3_SET_COOKIE_NX or COBW3_SET_COOKIE_NN, set the Cookie name to COBW3-COOKIE-NAME-N.

COBW3-COOKIE-NAME-LENGTH [optional]

If the Cookie name has a valid blank at the end, set the string length (byte length) of the Cookie name including the blank.

Value	Meaning
0	Registers the Cookie name using the length up to the last character excluding the blank. However, if COBW3-COOKIE-NAME or COBW3-COOKIE-NAME-N is completely blank, the string length is set to zero for processing.
1 to 64	Registers the Cookie name using the specified string length.

COBW3-COOKIE-VALUE and COBW3-COOKIE-VALUE-N

Set the Cookie value.

For COBW3_SET_COOKIE, COBW3_SET_COOKIE_XX and COBW3_SET_COOKIE_NX, set the Cookie value to COBW3-COOKIE-VALUE.

For COBW3_SET_COOKIE_XN or COBW3_SET_COOKIE_NN, set the Cookie value to COBW3-COOKIE-VALUE-N.

COBW3-COOKIE-VALUE-LENGTH [optional]

If the Cookie value has a valid blank at the end, set the string length (byte length) of the Cookie value including the blank.

Value	Meaning
0	Registers the Cookie value using the length up to the last character excluding the blank. However, if COBW3-COOKIE-VALUE or COBW3-COOKIE-VALUE-N is completely blank, the string length is set to zero for processing.
1 to 1024	Registers the Cookie value using the specified string length.

COBW3-COOKIE-EXPIRES [optional]

Used to specify the expiration date of Cookie. This item has the following items (year, month, day, hour, minute, and second). Use GMT (Greenwich mean time) to specify the expiration date.

- COBW3-COOKIE-EXPIRES-YEAR (year) (1582 to 9999)
- COBW3-COOKIE-EXPIRES-MONTH (month) (01 to 12)
- COBW3-COOKIE-EXPIRES-DAY (day) (01 to 31. However, the last day depends on the month)
- COBW3-COOKIE-EXPIRES-HOUR (hour) (00 to 23)
- COBW3-COOKIE-EXPIRES-MIN (minute) (00 to 59)
- COBW3-COOKIE-EXPIRES-SEC (second) (00 to 59)

Note

If the expiration date is omitted, the Cookie data is deleted when the WWW Browser is terminated. It is also possible to delete Cookie data held currently by the WWW Browser by specifying a past time.

The time-zone difference between JST (Japan standard time) and GMT is nine hours. Therefore, the value obtained by subtracting 9 from JST becomes GMT.

To specify the expiration date, use a date of October 15, 1582 or later. If a date before this date is specified, the day of week cannot be calculated automatically and so the expiration date becomes invalid.

COBW3-COOKIE-DOMAIN [optional]

Specify the domain where Cookie is valid.

COBW3-COOKIE-PATH [optional]

Specify the virtual directory where Cookie is valid.

COBW3-COOKIE-SECURE [optional]

Specify the security. When security is enabled, Cookie data can be sent from the WWW Browser only if SSL is used.

Condition name	Value	Meaning
COBW3-COOKIE-SECURE-OFF	LOW-VALUE	Disables the security.
COBW3-COOKIE-SECURE-ON	"1"	Enables the security.

COBW3-COOKIE-MODE [optional]

Set the registration type of Cookie data.

Condition name	Value	Meaning
COBW3-COOKIE-MODE-ADDREP	LOW-VALUE	Adds Cookie data if the specified Cookie name has not been registered.
COBW3-COOKIE-MODE-REPLACE	"1"	Replaces Cookie data. If the specified Cookie name has not been registered, an error is output and no replacement is carried out.
COBW3-COOKIE-MODE-ADD	"2"	Adds Cookie data. If the specified Cookie name has been registered, an error is output and no addition is made.

Processing result data

None

3.9.2 COBW3_DEL_COOKIE, COBW3_DEL_COOKIE_X and COBW3_DEL_COOKIE_N

These subroutines delete Cookie data registered by, for example, COBW3_SET_COOKIE.

The following lists the meaning of each subroutine:

ASCII environment

COBW3_DEL_COOKIE

Deletes Cookie data corresponding to an alphanumeric string Cookie name.

Unicode environment

COBW3_DEL_COOKIE_X

Deletes Cookie data corresponding to an alphanumeric string Cookie name.

COBW3_DEL_COOKIE_N

Deletes Cookie data corresponding to a national character string Cookie name.



Use this subroutine before using COBW3_PUT_HTML or COBW3_PUT_TEXT. If this subroutine is used after COBW3_PUT_HTML or COBW3_PUT_TEXT is used even once, the specified information will become invalid.

How to write

```
CALL "COBW3_DEL_COOKIE" USING COBW3 .
```

```
CALL "COBW3_DEL_COOKIE_X" USING COBW3 .
```

```
CALL "COBW3_DEL_COOKIE_N" USING COBW3 .
```

Data setting for calling

COBW3-COOKIE-NAME and COBW3-COOKIE-NAME-N

Set the Cookie name to be deleted.

For COBW3_DEL_COOKIE, COBW3_DEL_COOKIE_X, set the Cookie name to COBW3-COOKIE-NAME.

For COBW3_DEL_COOKIE_N, set the Cookie name to COBW3-COOKIE-NAME-N.

COBW3-COOKIE-NAME-LENGTH [optional]

If the Cookie name has a valid blank at the end, set the string length (byte length) of the Cookie name including the blank.

Value	Meaning
0	Deletes the Cookie name using the length up to the last character excluding the blank. However, if COBW3-COOKIE-NAME or COBW3-COOKIE-NAME-N is completely blank, the string length is set to zero for processing.
1 to 64	Deletes the Cookie name using the specified string length.

Processing result data

None

3.9.3 COBW3_INIT_COOKIE

This subroutine initializes all Cookie data registered by, for example, COBW3_SET_COOKIE.



Use this subroutine before using COBW3_PUT_HTML or COBW3_PUT_TEXT. If this subroutine is used after COBW3_PUT_HTML or COBW3_PUT_TEXT is used even once, the specified information will become invalid.

How to write

```
CALL "COBW3_INIT_COOKIE" USING COBW3 .
```

Data setting for calling

COBW3-COOKIE-INIT-MODE [optional]

Set the initialization type of Cookie data.

Condition name	Value	Meaning
COBW3-COOKIE-INIT-MODE-NORMAL	LOW-VALUE	Initializes all Cookie data.
COBW3-COOKIE-INIT-MODE-REQUEST	"1"	Initializes with the initial value of Cookie data of requested data.

Processing result data

None

3.9.4 COBW3_GET_COOKIE, COBW3_GET_COOKIE_XX, COBW3_GET_COOKIE_NX, COBW3_GET_COOKIE_XN, and COBW3_GET_COOKIE_NN

These subroutines search any Cookie name from the requested Cookie data acquired by "COBW3_INIT". Also, they acquire the Cookie value corresponding to the Cookie name.

The following lists the meaning of each subroutine:

ASCII environment

COBW3_GET_COOKIE

Searches an alphanumeric character string Cookie name to return the corresponding Cookie value as an alphanumeric character string.

Unicode environment

COBW3_GET_COOKIE-XX

Searches an alphanumeric character string Cookie name to return the corresponding Cookie value as an alphanumeric character string.

COBW3_GET_COOKIE-NX

Searches a national character string Cookie name to return the corresponding Cookie value as an alphanumeric character string.

COBW3_GET_COOKIE-XN

Searches an alphanumeric character string Cookie name to return the corresponding Cookie value as a national character string.

COBW3_GET_COOKIE-NN

Searches a national character string Cookie name to return the corresponding Cookie value as a national character string.

The padding string (padding character) used when the acquired Cookie value is blank.

How to write

```
CALL "COBW3_GET_COOKIE" USING COBW3.
```

```
CALL "COBW3_GET_COOKIE_XX" USING COBW3.
```

```
CALL "COBW3_GET_COOKIE_NX" USING COBW3.
```

```
CALL "COBW3_GET_COOKIE_XN" USING COBW3.
```

```
CALL "COBW3_GET_COOKIE_NN" USING COBW3.
```

Data setting for calling

COBW3-COOKIE-NAME and COBW3-COOKIE-NAME-N

Set the Cookie name to be searched.

For COBW3_GET_COOKIE, COBW3_GET_COOKIE_XX and COBW3_GET_COOKIE_XN, set the Cookie name to COBW3-COOKIE-NAME.

For COBW3_GET_COOKIE_NX or COBW3_GET_COOKIE_NN, set the Cookie name to COBW3-COOKIE-NAME-N.

COBW3-COOKIE-NAME-LENGTH [optional]

If the Cookie name has a valid blank at the end, set the string length (byte length) of the Cookie name including the blank.

Value	Meaning
0	Searches the Cookie name using the length up to the last character excluding the blank. However, if COBW3-COOKIE-NAME or COBW3-COOKIE-NAME-N is completely blank, the string length is set to zero for processing.
1 to 64	Searches the Cookie name using the specified string length.

Processing result data

COBW3-SEARCH-FLAG

Condition name	Value	Meaning
COBW3-SEARCH-FLAG-NON	"0"	The name to be searched does not exist.
COBW3-SEARCH-FLAG-EXIST	"1"	The name to be searched exists.

COBW3-COOKIE-VALUE and COBW3-COOKIE-VALUE-N

The Cookie value corresponding to the Cookie name to be searched is set.

For COBW3_GET_COOKIE, COBW3_GET_COOKIE_XX and COBW3_GET_COOKIE_NX, the Cookie value is set to COBW3-COOKIE-VALUE.

For COBW3_GET_COOKIE_XN or COBW3_GET_COOKIE_NN, the Cookie value is set to COBW3-COOKIE-VALUE-N.

COBW3-COOKIE-VALUE-LENGTH

The string length (byte length) of the Cookie value corresponding to the Cookie name to be searched is set.

3.10 Manipulate Uploaded Files

3.10.1 COBW3_GET_UPLOADFILE_INFO, COBW3_GET_UPLOADFILE_INFO_X and COBW3_GET_UPLOADFILE_INFO_N

These subroutines acquire the information regarding an uploaded file. Details of these subroutines are as follows:

ASCII environment

COBW3_GET_UPLOADFILE_INFO

Retrieves the name of an uploaded file (alphanumeric character string specified in NAME), and acquires the information about that uploaded file.

Unicode environment

COBW3_GET_UPLOADFILE_INFO_X

Retrieves the name of an uploaded file (alphanumeric character string specified in NAME), and acquires the information about that uploaded file.

COBW3_GET_UPLOADFILE_INFO_N

Retrieves the name of an uploaded file (national language character strings specified in NAME), and acquires the information about that uploaded file.

How to write

```
CALL "COBW3_GET_UPLOADFILE_INFO" USING COBW3.
```

```
CALL "COBW3_GET_UPLOADFILE_INFO_X" USING COBW3.
```

```
CALL "COBW3_GET_UPLOADFILE_INFO_N" USING COBW3.
```

Data setting for calling

COBW3-SEARCH-DATA and COBW3-SEARCH-DATA-N

Set the name (NAME) of an upload file to be retrieved (The name specified in "name" of the HTML document (Web page for invoking application) must be set instead of the file name).

For COBW3_GET_UPLOADFILE_INFO and COBW3_GET_UPLOADFILE_INFO_X, set the name to COBW3-SEARCH-DATA.

For COBW3_GET_UPLOADFILE_INFO_N, set the name to COBW3-SEARCH-DATA-N.

COBW3-SEARCH-LENGTH [optional]

If the name (NAME) has a valid blank at the end, set the string length (byte length) of the name including the blank.

Value	Meaning
0	Searches the upload file name using the length up to the last character excluding the blank. However, if COBW3-SEARCH-DATA or COBW3-SEARCH-DATA-N is completely blank, the string length is set to zero for processing.
1 to 1024	Searches the upload file name using the specified string length.

COBW3-NUMBER [optional]

If multiple entities with the same name (NAME) exist in Web parameters, set the order of appearance of names to be searched.

Condition name	Value	Meaning
COBW3-NUMBER-INIT	1	Searches the name that matches first.
-	2 to 9999	Searches the names in the specified order of appearance.

Processing result data

COBW3-SEARCH-FLAG

Condition name	Value	Meaning
COBW3-SEARCH-FLAG-NON	"0"	The name to be searched does not exist, or size of the uploaded file is 0.
COBW3-SEARCH-FLAG-EXIST	"1"	The name to be searched exists.

COBW3-UPLD-CL-FILE-PATH

Sets the path name of the upload file in the client that corresponds to the retrieved name (NAME).

COBW3-UPLD-CL-FILE-PATH-LENGTH

Sets the character string length (byte length) of the path name of the upload file in the client that corresponds to the retrieved name (NAME).

COBW3-UPLD-CL-FILE-NAME

Sets the file name of the upload file in the client that corresponds to the retrieved name (NAME).

COBW3-UPLD-CL-FILE-NAME-LENGTH

Sets the character string length (byte length) of the file name of the upload file in the client that corresponds to the retrieved name (NAME).

COBW3-UPLD-CONTENT-TYPE

Sets the character string that indicates the Content-type of the upload file that corresponds to the retrieved name (NAME).

COBW3-UPLD-CONTENT-TYPE-LENGTH

Sets the character string length (byte length) of the character string that indicates the content type of the upload file that corresponds to the retrieved name (NAME).

COBW3-UPLD-FILE-SIZE

Sets the uploaded file size (byte length).



Note

When "\" is included in the uploaded file name when the client is UNIX, the file name and the path name cannot be correctly received.



Example

c.htm (Web page for invoking application)

```
      :
<FORM METHOD="POST" ENCTYPE="multipart/form-data"
      ACTION="sample/action.script">
<P>
Your Name<INPUT TYPE="text" NAME="NAME1"><BR>
Send file 1:<INPUT TYPE="file" NAME="FILE1"><BR>
<INPUT TYPE="submit" VALUE="Send">
<INPUT TYPE="reset" VALUE="Reset">
</FORM>
```

COBOL source program

```
      :
* Name setup
  MOVE "FILE1" TO COBW3-SEARCH-DATA.
* Acquisition of upload file information
  CALL "COBW3_GET_UPLOADFILE_INFO_X" USING COBW3.
      :
  IF COBW3-SEARCH-FLAG-EXIST THEN
* Setup of name of file to be generated
  MOVE "d.tmp" TO COBW3-UPLOADED-FILENAME
* File generation
  CALL "COBW3_GEN_UPLOADFILE_X" USING COBW3
      :
* File data processing
      :
* Deletion of generated file
  CALL "COBW3_DEL_UPLOADEDFILE" USING COBW3
  END-IF.
      :
```

3.10.2 COBW3_GEN_UPLOADFILE, COBW3_GEN_UPLOADFILE_X and COBW3_GEN_UPLOADFILE_N

These subroutines generate an uploaded file by using a specified file name. The details of these subroutines are as follows:

ASCII environment

COBW3_GEN_UPLOADFILEX

Searches the name (NAME) of an alphanumeric character string, and generates the found uploaded file by using the specified file name.

Unicode environment

COBW3_GEN_UPLOADFILE_X

Searches the name (NAME) of an alphanumeric character string, and generates the found uploaded file by using the specified file name.

COBW3_GEN_UPLOADFILE_N

Searches the name (NAME) of a national character string, and generates the found uploaded file by using the specified file name.



Uploaded files must not be generated as executable files because of the threat to security. Especially, uploaded files must not be generated as files that can be automatically executed by the system.

How to write

```
CALL "COBW3_GEN_UPLOADFILE" USING COBW3.
```

```
CALL " COBW3_GEN_UPLOADFILE_X" USING COBW3.
```

```
CALL " COBW3_GEN_UPLOADFILE_N" USING COBW3.
```

Data setting for calling

COBW3-SEARCH-DATA and COBW3-SEARCH-DATA-N

Set the name (NAME) of the uploaded file to be searched (The name specified in "name" of the HTML document [Web page for invoking application] must be set instead of the file name).

For COBW3_GEN_UPLOADFILE and COBW3_GEN_UPLOADFILE_X, set the name to COBW3-SEARCH-DATA.

For COBW3_GEN_UPLOADFILE_N, set the name to COBW3-SEARCH-DATA-N.

COBW3-SEARCH-LENGTH [optional]

If the value (VALUE) has a valid blank at the end, set the string length (byte length) of the value including the blank.

Value	Meaning
0	Searches the name using the length up to the last character excluding the blank.
1 to 1024	Searches the name using the specified string length.

COBW3-NUMBER [optional]

Sets the order of appearance of NAMES to be searched when there are multiple NAMES with the same name in the Web parameters.

Condition name	Value	Meaning
COBW3-NUMBER-INIT	1	Searches the name that matches first.
-	2 to 9999	Searches the name in the specified order of appearance.

COBW3-UPLOADED-FILENAME

Specify the name of the file to be generated on the WWW Server.

Processing result data

COBW3-SEARCH-FLAG

Condition name	Value	Meaning
COBW3-SEARCH-FLAG-NON	"0"	The name to be searched does not exist.
COBW3-SEARCH-FLAG-EXIST	"1"	The name to be searched exists.

3.10.3 COBW3_DEL_UPLOADFILE

This subroutine deletes an uploaded file that was generated using a subroutine such as COBW3_GEN_UPLOADFILE.

This subroutine can delete a file that was generated using COBW3_GEN_UPLOADFILE, COBW3_GEN_UPLOADFILE_X or COBW3_GEN_UPLOADFILE_N within the same request. File generation or deletion among two or more requests (two or more pages) cannot be performed.

How to write

```
CALL "COBW3_DEL_UPLOADFILE" USING COBW3.
```

Data setting for calling

COBW3-UPLOADED-FILENAME

Set the name of the file to be deleted.

Processing result data

None

3.11 Manage Sessions

3.11.1 COBW3_START_SESSION

This subroutine starts a session. At this point, the format of the session data of the session to be opened and the timeout period are also set.



Note

Use this subroutine before using "COBW3_PUT_TEXT", "COBW3_PUT_HTML", or "COBW3_FREE". If this subroutine is called after using one of these subroutines even once, the session will not be started.

If this subroutine is called when a session is already started, an error occurs.

The format of the session data cannot be changed while a session is open. To change the format of the session data, call "COBW3_END_SESSION" to terminate the session temporarily.

How to write

```
CALL "COBW3_START_SESSION " USING COBW3.
```

Data setting for calling

COBW3-SESSION-DATA-TYPE

Specify the format of session data.

Condition name	Value	Meaning
COBW3-SESSION-DATA-GROUPITEM	"0"	Uses a group item as session data.

Condition name	Value	Meaning
COBW3-SESSION-DATA-OBJECT	"1"	Uses an object reference item as session data.

COBW3-SESSION-TIMEOUT

Specify the timeout period (s) of a session. The timeout period is the maximum wait time between the instant when "COBW3_FREE" is called and the instant when "COBW3_INIT" is called in the next thread within the same session.

Value	Meaning
1 to 999999	The timeout processing is performed when the specified time interval (s) passes.

Processing result data

None

3.11.2 COBW3_END_SESSION

This subroutine terminates a session. At this point, the information specified when the current session was started is deleted.



Note

Use this subroutine before using "COBW3_PUT_TEXT", "COBW3_PUT_HTML", or "COBW3_FREE". If this subroutine is called after using one of these subroutines, the session cannot be terminated.

If this subroutine is called when the session has already been terminated, an error occurs.

How to write

```
CALL "COBW3_END_SESSION " USING COBW3.
```

Data setting for calling

None

Processing result data

None

3.11.3 COBW3_SET_SESSION_DATA

This subroutine registers the session data to be used in the current session. The registered session data is shared in the same session.



Note

Call this subroutine while a session is open. If this subroutine is called before a session is started or after a session is terminated, session data cannot be registered.

The format of session data cannot be changed while a session is open. Register session data matching the format of the session data set when "COBW3_START_SESSION" was called.

How to write

```
CALL "COBW3_SET_SESSION_DATA" USING COBW3 session data.
```

Data setting for calling

COBW3-SESSION-DATA-SIZE

Specify the session data length (byte length). This specification is valid only if COBW3-SESSION-DATA-GROUPITEM is set as the format of session data when a session is started.

Value	Meaning
1 to 999999999	Registers session data with the specified data length.

Session data

Specify the session data to be registered. Specify the group item or object reference item matching the format of the session data when a session was started.



When specifying a group item as the session data, a group item containing an object reference item or pointer data item as a subordinate basic item cannot be specified.

When specifying an object reference item as the session data, the class of the object pointed by the object reference item must have inherited the COBW3-SESSION-ADAPTER class.

If an object reference item is specified as the session data, be sure to assign a NULL object to the object reference item before the thread terminates. If the NULL object is not assigned, an object that is referenced from nowhere may remain in memory after the session terminates, leading to memory shortage.

When the session data is registered, the actual contents of the registered data are held if the data is a group item. If session data is an object reference item, objects are held.

If the session data to be registered is in a different format from that set when a session was opened, operations of Web applications cannot be guaranteed.

Processing result data

None

3.11.4 COBW3_GET_SESSION_DATA

This subroutine acquires session data registered with the current session.



Call this subroutine while a session is open. If this subroutine is called before a session is started or after a session is terminated, session data cannot be acquired.

The format of session data cannot be changed while a session is opened. Acquire session data fitting to the format of session data set when "COBW3_START_SESSION" was called.

How to write

```
CALL "COBW3_GET_SESSION_DATA" USING COBW3 session data.
```

Data setting for calling

COBW3-SESSION-DATA-SIZE

Specify the group item length (byte length) in which the acquired session data is to be stored. This specification is valid only if COBW3-SESSION-DATA-GROUPITEM is set as the format of session data when a session is started.

Value	Meaning
1 to 999999999	Acquires session data using the specified length of the session data. An error occurs if the length is different from that of the session data registered with the session.

Session data

Specify the session data to be acquired. Specify a group item or an object reference item matching the format of the session data when a session was started.

Note

If object reference items are registered as session data, the object reference item specified as a call parameter of this subroutine must have been initialized by the NULL object. If not initialized by the NULL object, session data cannot be acquired. When specifying an object reference item as session data, specify an object reference item with the USAGE OBJECT REFERENCE clause without any other selection specification.

If session data to be acquired is in a different format from that set when a session was opened, operations of Web applications cannot be guaranteed.

When specifying a group item as the session data, we recommend using a library file to match the configuration of the session data in which the configuration of the subordinate basic items are registered.

Processing result data

None

3.11.5 COBW3_ALTER_SESSION_TIMEOUT

This subroutine changes the timeout period (s) of the current session.

Note: Call this subroutine while a session is open. If this subroutine is called before a session is started or after a session is terminated, the timeout period of the session cannot be changed.

How to write

```
CALL "COBW3_ALTER_SESSION_TIMEOUT" USING COBW3.
```

Data setting for calling

COBW3-SESSION-TIMEOUT

Specify the timeout period (s) of a session. The measurement of the timeout period is the maximum wait time between the instant when "COBW3_FREE" is called and the instant when "COBW3_INIT" is called in the next thread within the same session.

Value	Meaning
1 to 999999	The timeout processing is performed when the specified time interval (s) passes.

Processing result data

None

3.11.6 COBW3_GET_SESSION_INFO

This subroutine acquires information about the current session.

How to write

```
CALL "COBW3_GET_SESSION_INFO" USING COBW3.
```

Data setting for calling

None

Processing result data

COBW3-SESSION-TIMEOUT

The timeout period (s) of the current session is set.

COBW3-SESSION-ID

The number indicating the session ID of the current session is set.

COBW3-SESSION-DATA-SIZE

The session data length (byte length) of the current session is set. This data can be acquired only if COBW3-SESSION-DATA-GROUPITEM is set as the format of the session data when a session is started. Zero is set if COBW3-SESSION-DATA-OBJECT is set as the format of the session data.

COBW3-SESSION-DATA-TYPE

The format of the session data of the current session is set.

Condition name	Value	
COBW3-SESSION-DATA-GROUPITEM	"0"	Uses a group item as the session data.
COBW3-SESSION-DATA-OBJECT	"1"	Uses an object reference item as the session data.

COBW3-SESSION-STATUS

The status of the current session is set.

Condition name	Value	
COBW3-SESSION-STATUS-NON	"0"	A session has not been started or has been terminated.
COBW3-SESSION-STATUS-STARTED	"1"	A session has been started.

3.12 Other Subroutines

The following subroutines are provided to guarantee compatibility.



Operations of the following subroutines are guaranteed only if the operation code system of applications is ASCII.

3.12.1 COBW3_NAME

These subroutines search any name (NAME) from Web parameters acquired by "COBW3_INIT".

These subroutines also acquire the value (VALUE) corresponding to the name (NAME). The padding string (padding character) used when the acquired value (VALUE) is blank.

How to write

```
CALL "COBW3_NAME" USING COBW3.
```

Data setting for calling

COBW3-SEARCH-DATA

Set the name (NAME) to be searched (The name here is that specified in the name of the HTML document of the page for calling).

COBW3-SEARCH-LENGTH [optional]

If the name (NAME) has a valid blank at the end, set the string length (byte length) of the name including the blank.

Value	Meaning
0	Searches the name using the length up to the last character excluding the blank.
1 to 1024	Searches the name using the specified string length.

COBW3-NUMBER [optional]

If multiple entities with the same name (NAME) exist in the Web parameters, set the order of appearance of the names to be searched.

Condition name	Value	Meaning
COBW3-NUMBER-INIT	1	Searches the name that matches first.
-	2 to 9999	Searches the names in the specified order of appearance.

Processing result data

COBW3-SEARCH-FLAG

Condition name	Value	Meaning
COBW3-SEARCH-FLAG-NON	"0"	The name to be searched does not exist.
COBW3-SEARCH-FLAG-EXIST	"1"	The name to be searched exists.

COBW3-GET-DATA

VALUE corresponding to NAME to be searched is set.

COBW3-GET-LENGTH

The string length (byte length) of VALUE corresponding to NAME to be searched is set.

3.12.2 COBW3_VALUE

This subroutine searches any value (VALUE) from the Web parameters acquired by "COBW3_INIT". This subroutine is used, for example, to evaluate the items checked by the WWW Browser based on the value (VALUE) when activated from a page for calling with check boxes

How to write

```
CALL "COBW3_VALUE" USING COBW3.
```

Data setting for calling

COBW3-SEARCH-DATA

Set the value to be searched.

COBW3-SEARCH-LENGTH [optional]

If the value (VALUE) has a valid blank at the end, set the string length (byte length) of the value including the blank.

Value	Meaning
0	Searches the value using the length up to the last character excluding the blank.
1 to 1024	Searches the value using the specified string length.

COBW3-NUMBER [optional]

If multiple entities with the same value exist in Web parameters, set the order of appearance of values to be searched.

Condition name	Value	Meaning
COBW3-NUMBER-INIT	1	Searches the value that matches first.
-	2 to 9999	Searches the values in the specified order of appearance.

Processing result data

COBW3-SEARCH-FLAG

Condition name	Value	Meaning
COBW3-SEARCH-FLAG-NON	"0"	The value to be searched does not exist.
COBW3-SEARCH-FLAG-EXIST	"1"	The value to be searched exists.

3.12.3 COBW3_CNV_SET

This subroutine registers conversion data specified in the Web page for processing result output to be output by "COBW3_PUT_HTML".

The registered information is referenced during execution of COBW3_PUT_HTML and conversion in the Web page for processing result output is carried out according to the registered data.



Note

If multiple different conversion names are specified in the Web page for processing result output, call "COBW3_CNV_SET" for each conversion name to register conversion data.

How to write

```
CALL "COBW3_CNV_SET" USING COBW3.
```

Data setting for calling

COBW3-CNV-NAME

Set the conversion name to be converted.

COBW3-CNV-NAME-LENGTH [optional]

If the conversion name has a valid blank at the end, set the string length (byte length) of the conversion name including the blank.

Value	Meaning
0	Registers the conversion name using the length up to the last character excluding the blank. However, if COBW3-CNV-NAME is completely blank, the string length is set to zero for processing.
1 to 30	Registers the conversion name using the specified string length

COBW3-CNV-VALUE

Set the conversion results (conversion character string).

COBW3-CNV-VALUE-LENGTH [optional]

If the conversion string has a valid blank at the end, set the string length (byte length) of the conversion string including the blank.

Value	Meaning
0	Registers the conversion string using the length up to the last character excluding the blank. However, if COBW3-CNV-VALUE is completely blank, the string length is set to zero for processing.
1 to 1024	Registers the conversion string using the specified string length

COBW3-CNV-MODE [optional]

Set the conversion type.

Condition name	Value	Meaning
COBW3-CNV-MODE-ADDREP	LOW-VALUE	Adds conversion data if the specified conversion name has not been registered.
COBW3-CNV-MODE-REPLACE	"1"	Replaces conversion data. If the specified conversion name has not been registered, an error is output and no replacement is carried out.
COBW3-CNV-MODE-ADD	"2"	Adds conversion data. If the specified conversion name has been registered, an error is output and no addition is made

Processing result data

None

3.12.4 COBW3_CNV_DEL

This subroutine deletes conversion data registered by COBW3_CNV_SET.

How to write

```
CALL "COBW3_CNV_DEL" USING COBW3.
```

Data setting for calling

COBW3-CNV-NAME

Set the conversion name to be deleted.

COBW3-CNV-NAME-LENGTH [optional]

If the conversion name has a valid blank at the end, set the string length (byte length) of the conversion name including the blank.

Value	Meaning
0	Registers the conversion name using the length up to the last character excluding the blank. However, if COBW3-CNV-NAME is completely blank, the string length is set to zero for processing.
1 to 30	Registers the conversion name using the specified string length

Processing result data

None

3.12.5 COBW3_CNV_INIT

This subroutine initializes conversion data registered by CONV3_CNV_SET.

How to write

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

Data setting for calling

None

Processing result data

None

3.13 Quantitative Limitations of ISAPI Subroutines

The following table lists the quantitative limitations of data used by ISAPI Subroutines.

Item	Value
Maximum string length of NAME from Web parameters	1024 bytes
Maximum string length of VALUE from Web parameters	1024 bytes
Maximum record length of the Web page for processing result output specified in COBW3-HTML-FILENAME (*1)	1024 bytes
Length of one line of the Web page for processing result output after replacing conversion data	3072 bytes

*1 : The number of bytes of one line of HTML documents created by using HTML editors on the market may exceed 1024 bytes. In such cases, use a text editor such as Notepad to modify HTML documents so that they do not exceed 1024 bytes.

3.14 ISAPI Subroutine Classes

ISAPI Subroutines provide classes to be used by the session management function.

3.14.1 COBW3-SESSION-ADAPTER class

Class to manage objects that are registered as session data. Objects to be registered as session data need to be created from classes that are inherited from this class. Write the timeout processing in the method obtained by overriding the SWEEP-SESSION method defined in this class by using the OVERRIDE clause.



Timeout processing can be used only if an object reference is registered in the session data.

In timeout processing method, ISAPI Subroutines cannot be called.

Factory method

None

Object method

SWEEP-SESSION

Method called when the timeout of a session occurs. This method is used to close files or to perform DB commit/rollback processing as required.

Method argument

None

Method return information

None

When the timeout of a session occurs, ISAPI Subroutines call the SWEEP-SESSION method to assign a NULL object to the object reference registered in the session data.

If the SWEEP-SESSION method is not defined by overriding it, the SWEEP-SESSION method performs no processing.

The COBW3-SESSION-ADAPTER class is a class that inherits the FJBASE class.



Example

COBOL program

```
IDENTIFICATION DIVISION.  
CLASS-ID. SESSDATA INHERITS COBW3-SESSION-ADAPTER.  
:  
  IDENTIFICATION DIVISION.  
  OBJECT.  
  :  
  IDENTIFICATION DIVISION.  
  METHOD-ID. SWEEP-SESSION OVERRIDE.  
  DATA DIVISION.  
  :  
  PROCEDURE DIVISION.  
  :  
* Write the timeout processing as required.  
  :  
  END METHOD SWEEP-SESSION.  
  :  
  END OBJECT.  
END CLASS SESSDATA.
```

Chapter 4 Creating and Executing a Web Application

This chapter describes the basic methods for compiling, linking, and executing a Web application. For details about compile and link options not described in this chapter, see the "*NetCOBOL User's Guide*."

4.1 Execution Procedure

Execute a Web application as follows:

1. Compile and link a created program.

For details, see "Compiling and Linking" in Chapter 4.

2. Configure the environment.

- Make IIS environmental settings.

For details, see "IIS Settings" in Chapter 4.

- Make ISAPI Subroutine environmental settings.

For details, see "ISAPI Subroutine Environment Variable Settings" in Chapter 4.

3. Execute the Web application.

Bring up the Web page for invoking application (HTML document) registered on IIS to start the application. For details, see "*Executing a Web Application*" in Chapter 4.

4.2 Compiling and Linking

Compile and link programs with entry-names of GetExtensionVersion, HttpExtensionProc, and TerminateExtension as follows. In this explanation, the related files are Version.cob, MainProc.cob, and Terminate.cob. If you are using the project management function, use the build option in the Project Manager to compile and link these programs.

Compile the COBOL source program.

To compile a COBOL source program, either execute the WINCOB command in Project Manager to use the window operation or type a compilation command at the command prompt. This section provides an example of typing a compile command to compile a COBOL source program.

For details about compilation using the WINCOB command or a compile command, see the "*NetCOBOL User's Guide*."

```
COBOL32 -WC, "ALPHAL(WORD),THREAD(MULTI)" Version.cob
COBOL32 -WC, "ALPHAL(WORD),THREAD(MULTI)" MainProc.cob
COBOL32 -WC, "ALPHAL(WORD),THREAD(MULTI)" Terminate.cob
```



Note

Instead of the compile option ALPHAL (WORD), you can also specify NOALPHAL. Specify either of these options because the entry-name of a program to be used as an export function is case-sensitive. Also specify the compile option THREAD (MULTI) because ISAPI Applications should be multi-threaded applications.

Linking a COBOL object program (creating a DLL)

To link the object code created by compiling a COBOL source program, either execute the WINLINK command in the Project Manager to use the window operation or type a link command at the command prompt. This section provides an example of typing a link command to link a COBOL source program. For details about linkage using the WINLINK command or a link command, see the "*NetCOBOL User's Guide*."

In this explanation, the DLL to be created is ISASMPL.DLL. Before linking the program, prepare a module definition file as described later. In this explanation, this file is names ISASMPL.def, and contains the following statements:

```
LIBRARY ISASMPL
EXPORTS
    GetExtensionVersion
    HttpExtensionProc
    TerminateExtension
```

Link the program as follows:

```
LINK /DLL /OUT:ISASMPL.DLL Version.obj MainProc.obj Terminate.obj
      F3BICBDM.obj F3BISAPI.lib F3BICIMP.lib KERNEL32.LIB MSVCRT.LIB
      /DEF:ISASMPL.def /ENTRY:COBDMAIN
```

If you link F3BICBDM.obj and specify /ENTRY:COBDMAIN, you can place an initialization file (COBOL85.CBR) in the same folder as the DLL, regardless of the current folder. Since the current folder is undefined for an application under a WWW Server such as IIS, being able to place an initialization file in the same folder as the DLL is very convenient. For details, see the "*NetCOBOL User's Guide*."



ISAPI Subroutines are NOT downward compatible. This means that a library file with a version level newer than a subroutine to run (COBW3.cbl) cannot be used. Additionally, do not link an ISAPI subroutine with a version level older than a library file brought in by the Web application.

4.3 IIS Settings

To make IIS settings, simply register a virtual directory required to reference an HTML document such as a Web page for invoking the application or simply execute a Web application. See the following procedure. For advanced settings and details about settings, see a reference book or the online manual of IIS. To make IIS settings, use Internet Service Manager.

Regarding the IIS settings to execute Web subroutine, sample program (WSESSION) is used in the following explanation. Before configuring IIS, please install internet information service.

If debugging needs to be performed using debugger, please perform "[Checking the Operation Using the Interactive Debugger](#)".



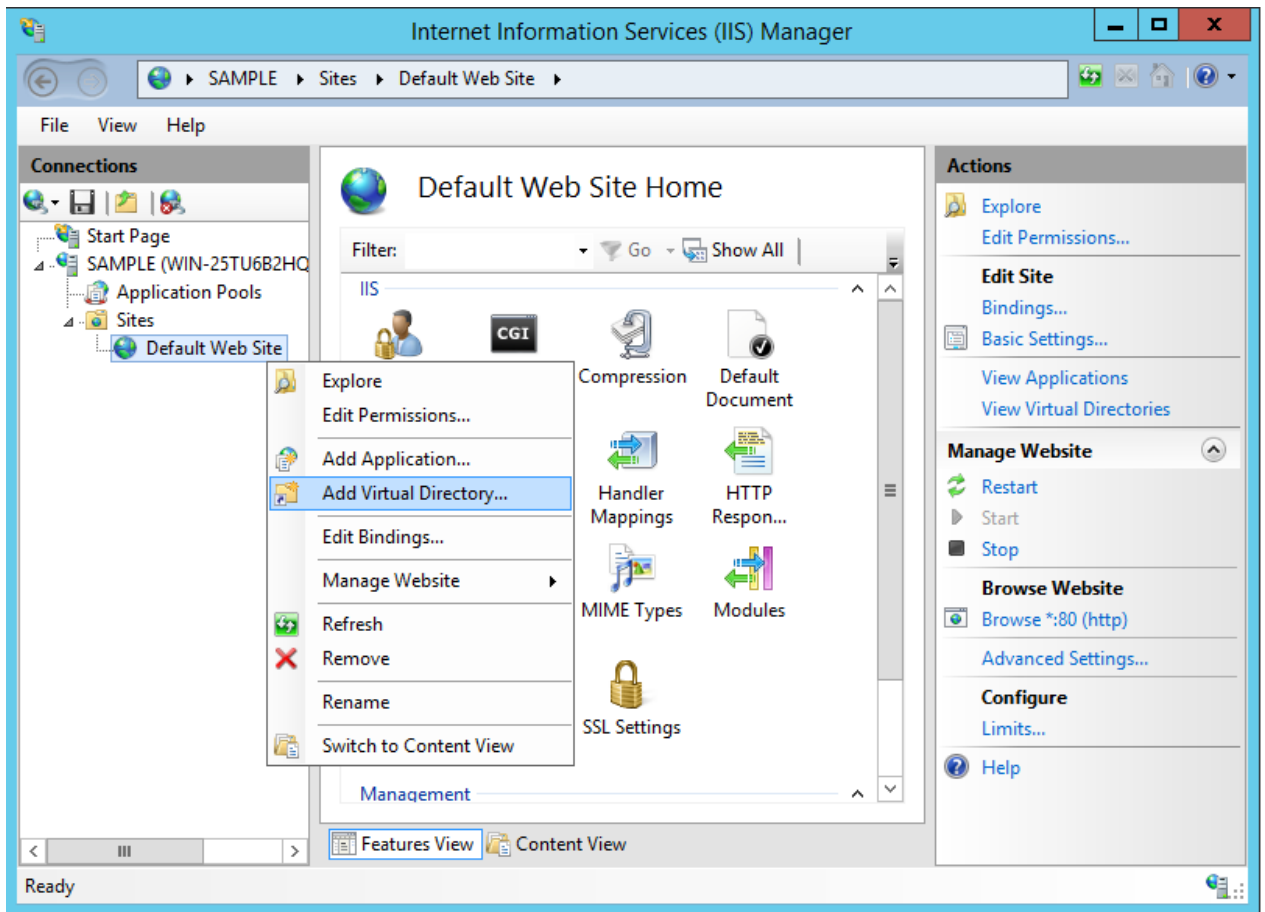
- CGI/ISAPI application configured with IIS is executed with [IUSR_<computername>] account by default.
- With [IUSR_<computername>] account, accessing files without permissions or execution of OBW3_SYSTEM subroutine and execution of system commands may fail. Check the security and permissions of directories and files, if necessary please provide appropriate permissions or modify the account.

IIS configuration steps (In case of IIS 8.5)

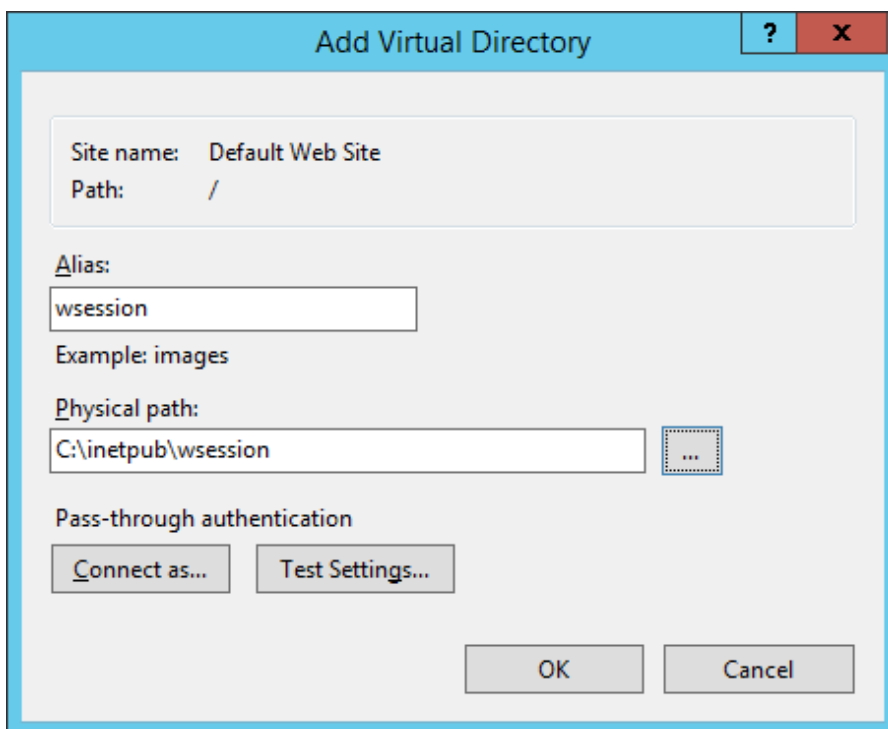
Web subroutine cannot be used if only Internet Information Services is installed. With the following steps, please enable ISAPI extension or CGI feature.

1. Run the [Programs and Features] of [Control Panel].
2. Run the [Turn Windows features on or off] task.
3. From the [World Wide Web service] of [Internet Information Service], select Application Development Features.
If [ISAPI Extensions] (CGI is used, CGI) should be additionally installed.

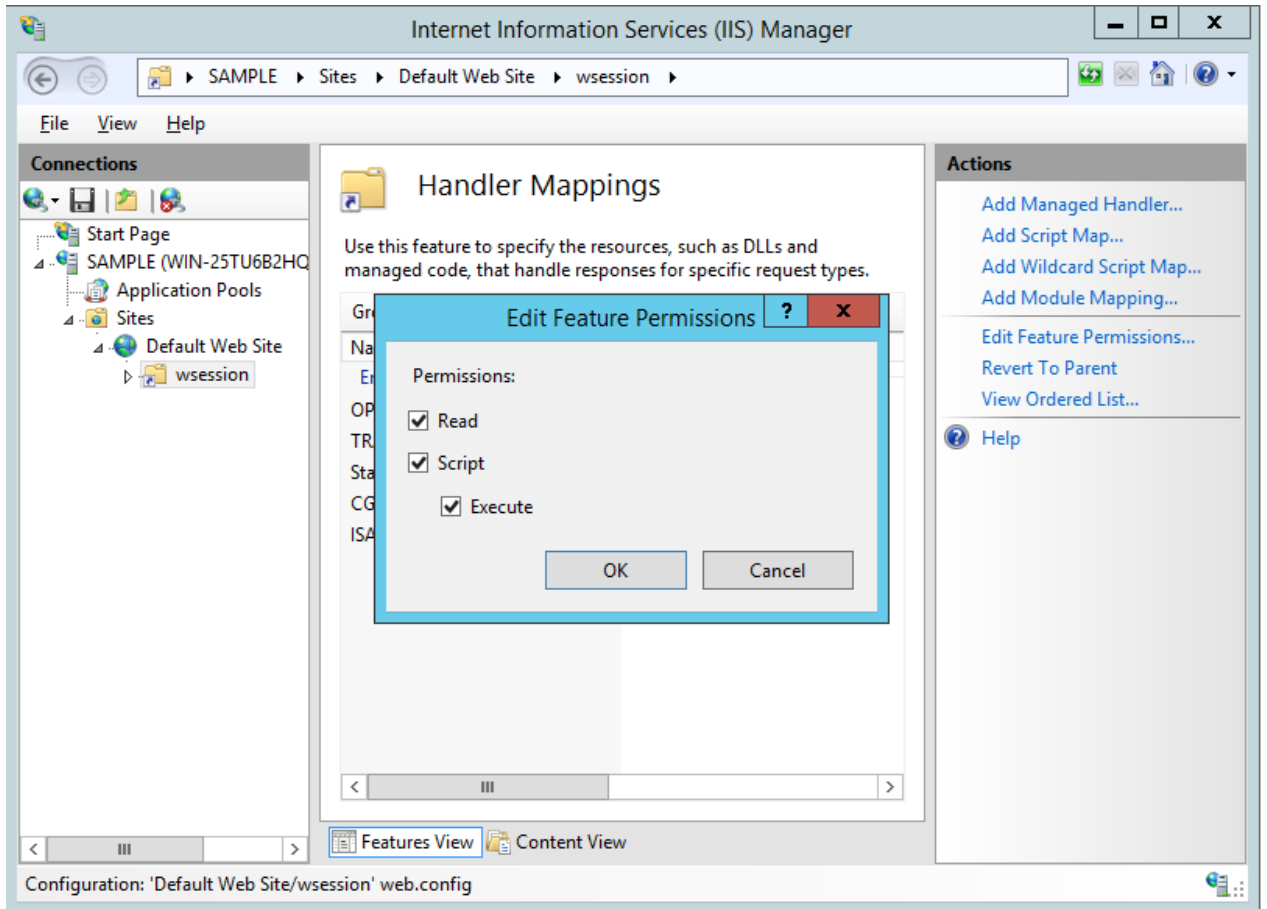
4. Start the [Internet Information Services (IIS) Manager], create a virtual directory for Web site.



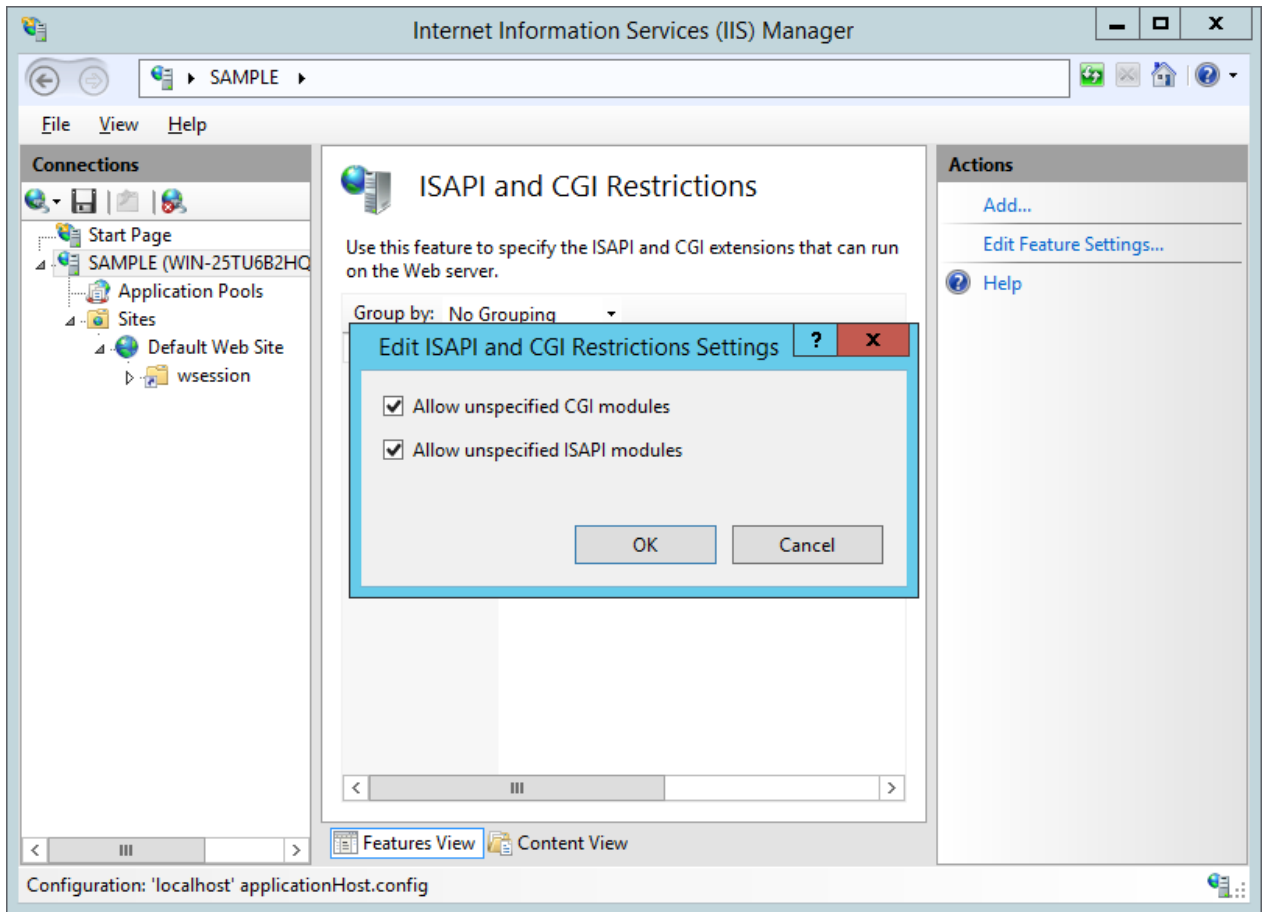
5. Specify an alias to access the virtual directory. Any name can be specified. Also, please specify the physical path that contains the contents to be published on the Web site.



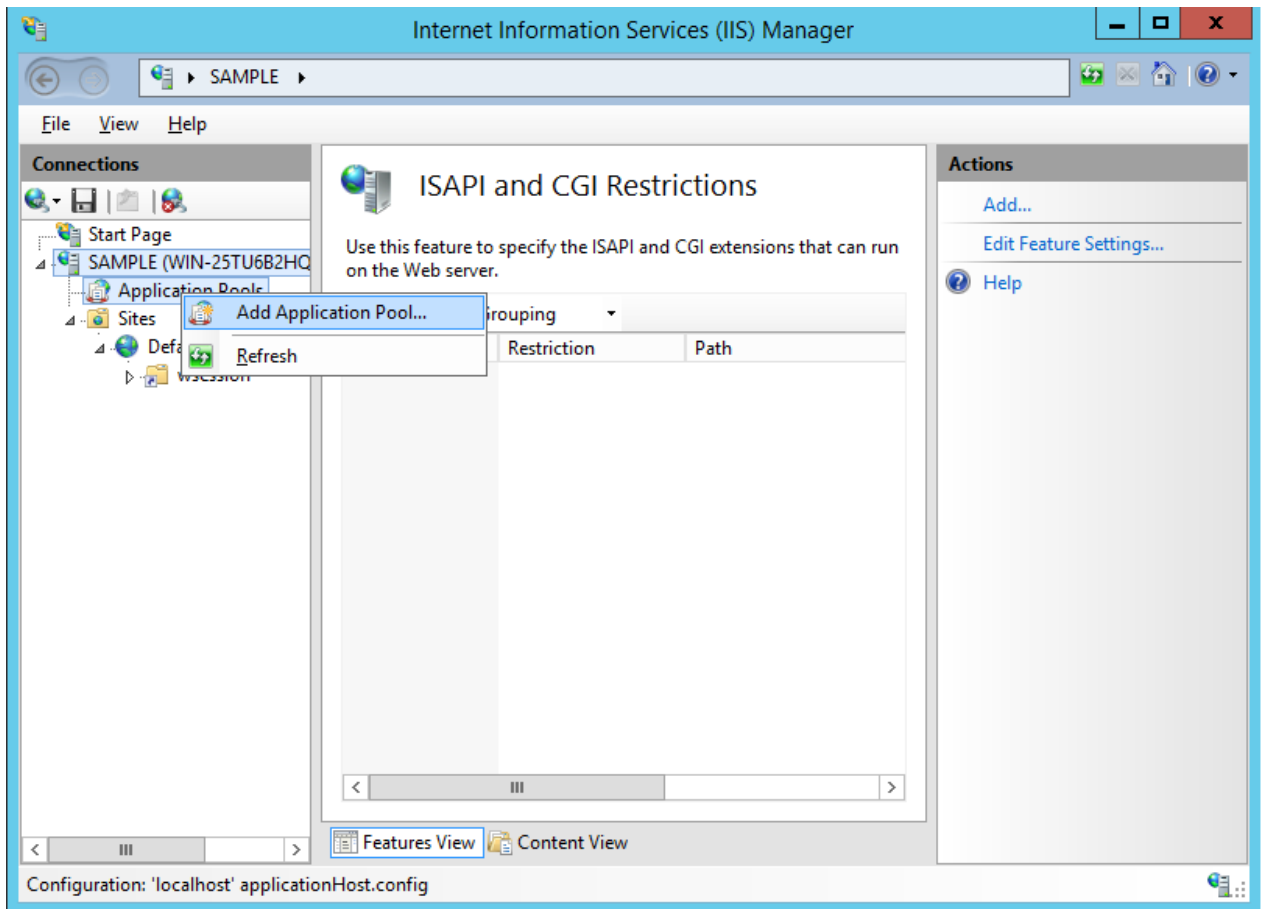
6. Select the virtual directory that is created, open the function [Handler Mapping] and then check [Execute] in the Edit Feature Permissions.



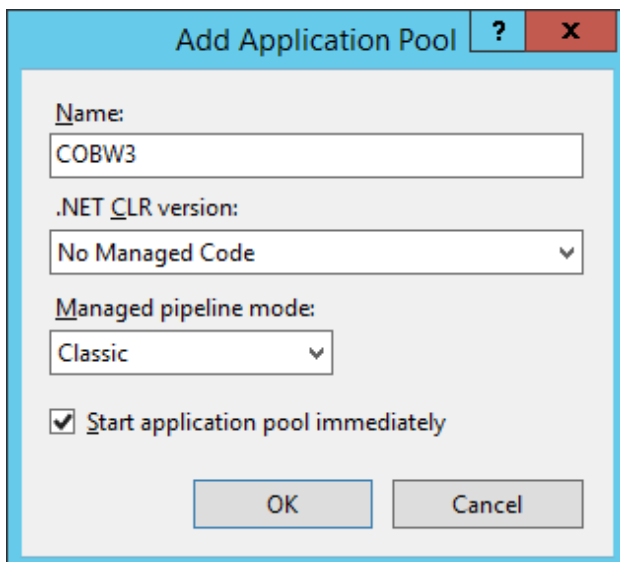
7. Open the [ISAPI and CGI Restrictions] feature available at the root of the Internet Information Services (IIS) Manager and then select [Edit Feature Settings]. Please check the [Allow unspecified ISAPI modules]. (If CGI is to be used, please check the [Allow unspecified CGI modules]).



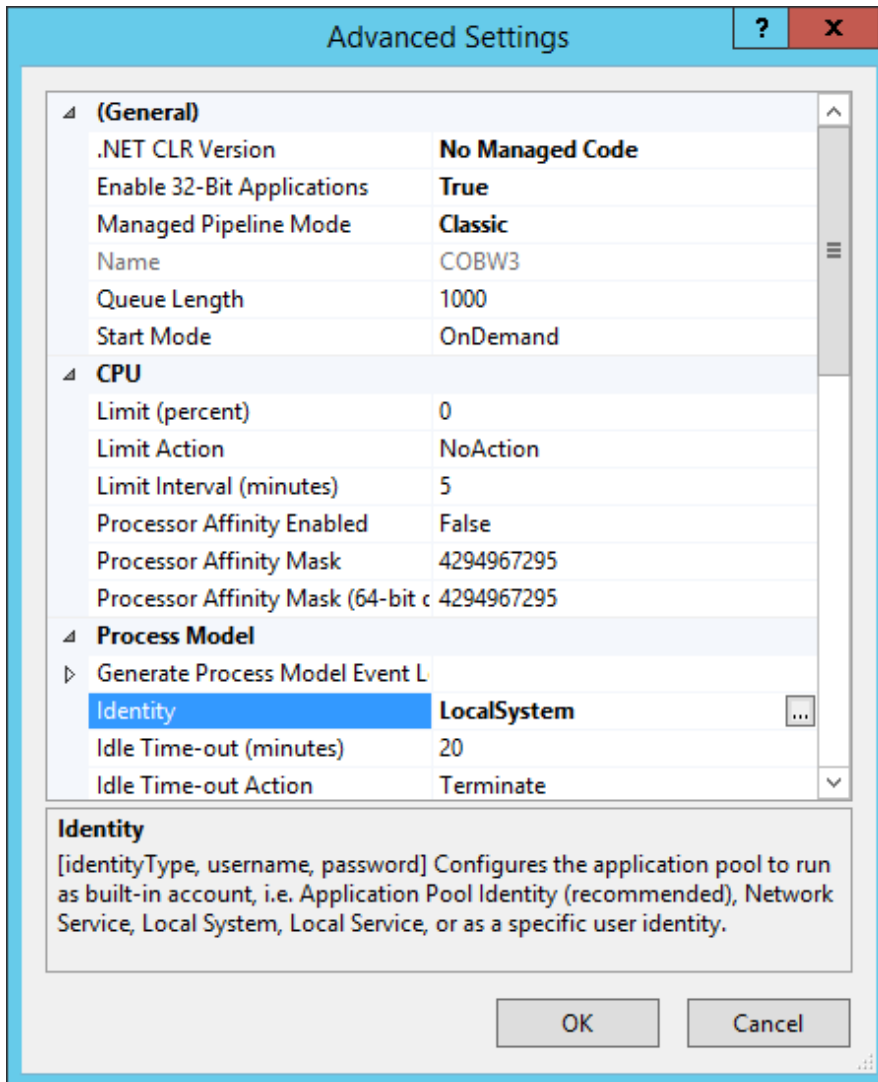
8. Create the application pool to start the Web subroutine.



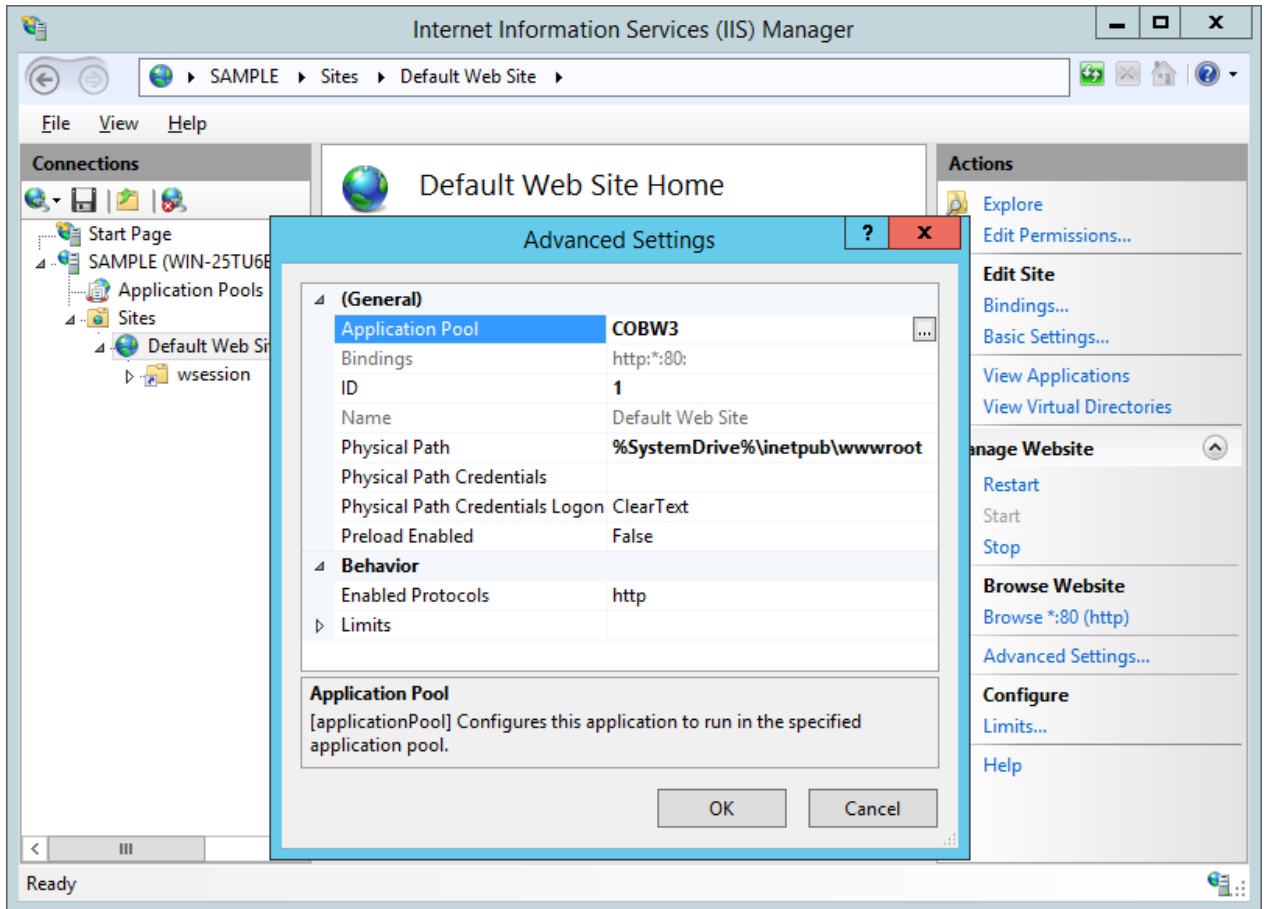
9. Any name can be specified to [Name]. Here COBW3 is used. Please select the [.Net CLR version] as [No Managed Code] and [Managed pipeline mode] as [Classic].



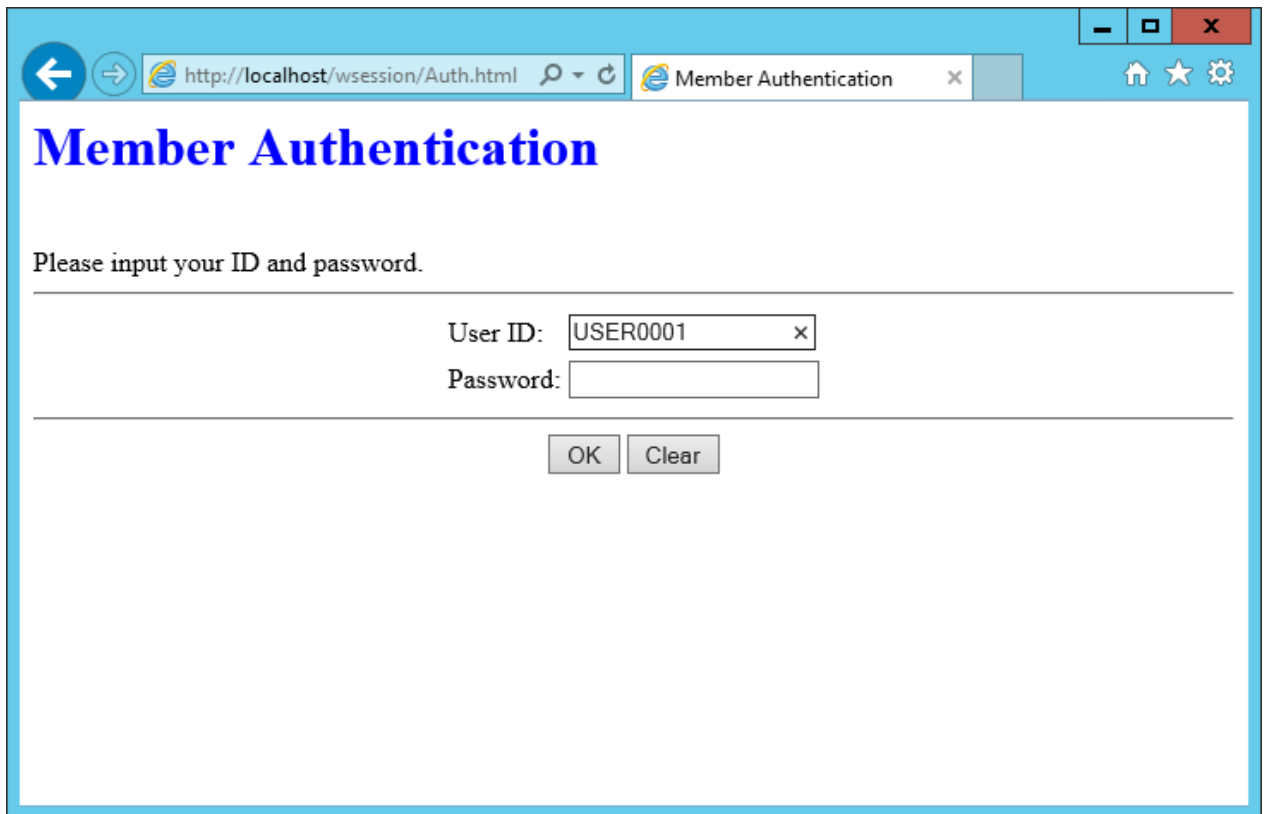
- Open the [Advanced Settings] of the application pool that is created, select the [Identity] of the Process Model and please select the [LocalSystem] from the pull-down menu. If 64-bit OS is in use, it is necessary to execute the CGI or ISAPI application in WOW64 mode. Please set [True] to [Enable 32-Bit Applications].



11. Open the Advanced Settings of "Default Web Site" and select [Application Pool] in the [General], from the pull-down menu specify the newly created application pool ID.



12. Confirm that the created virtual directory is accessible from the browser and check the operation.



Member Authentication

Please input your ID and password.

User ID: x

Password:

OK Clear

Note

After uninstalling NetCOBOL, please delete the application pool that was created.

Application pool cannot be recycled. Please do not perform the automatic recycling at regular intervals or recycle due to the processing of the application pool task. If recycling is performed, session of running ISAPI application will be terminated.

Web garden cannot be used. Please do not specify a value greater than 1 to the maximum number of worker processes.

4.4 ISAPI Subroutine Environment Variable Settings

Before a Web application using an ISAPI subroutine can be executed, the following run-time environment must be set in an environment variable or in a run-time initialization file (COBOL85.CBR).

To use a different run-time initialization file depending on the business, place each business application in a different folder and assign a different virtual directory to each business using IIS to configure it as a separated process. Additionally, link F3BICBDM.obj into each application so that the run-time initialization file of a folder storing the application can be referenced.

4.4.1 Required Settings in System Environment Variables

PATH

Specify the path of a dynamic link library (.DLL) to be used by a Web application.

4.4.2 Required Settings in a System Environment Variable or a Run-Time Initialization File

@CBR_ATTACH_TOOL=TEST [Start parameter]

It is specified to start the debugger from the program which wants to be debugged. When the Web application made in COBOL is debugged by using the debugger, it is necessary to set this environment variable.

Please refer to "Checking the Operation Using the Interactive Debugger" for details.

@MessOutFile = file-name

Specify the name of a file that stores all run-time messages output by the COBOL run-time system. This inhibits the appearance of a message box on the screen, for example. Use an absolute path to specify the file name. If a file with the same name already exists, messages are appended to that file.

@WindCloseMsg=OFF

Specify whether a confirmation message should appear (ON) or not (OFF) when a window is closed. Specify that a confirmation message should not appear (OFF) while a Web application is running.

@CBR_ISAPI_LOGFILE=log-file-name

Specify a file in which to store log information to be output by the ISAPI Subroutines. This information will be useful when you check the cause of a error. As with the above, use an absolute path on the WWW Server to specify the file name. By default, no log is output.

@CBR_ISAPI_SEVERITY=significance

Specify the significance of log information to be output by the ISAPI Subroutines. Specify one of the following values. If any value other than the following or no value is specified, the significance is assumed to be 0.

Significance	Output in a log file
0	Only critical errors are output.
1	Errors in addition to those with Significance 0 are output.
2	Warnings in addition to those with Significance 1 are output.
3	Traces of ISAPI Subroutines in addition to those with Significance 2 are output.

Note

Specify a value that causes only required errors to be output during operation. An application runs significantly slower as you specify a larger value. In particular, you are not recommended to specify 3 unless you are investigating the cause of an error.

For other run-time environment information and details, see the "*NetCOBOL User's Guide*."

4.5 Executing a Web Application

Place the Web application that you created in the physical path corresponding to a virtual directory and start it using an existing Web page for invoking the application.

While you are testing, you will need to frequently swap Web applications. Even in an operational environment, you will need to swap Web applications when you expand the functions of a Web application. However, a Web application, once executed, is loaded into IIS and made resident, and therefore cannot be swapped. For information on swapping Web applications, see "*Swapping Web applications*" in Chapter 5.

Chapter 5 Operation Check

You can check the operation of a program that you created (while you are testing or when a problem occurs) either by:

- Referencing log information,
- Checking the operation using the interactive debugger,
- Referencing an error detected by an ISAPI subroutine, or
- Referencing the data being executed in a display format

The following sections describe each of these debugging methods.

5.1 Referencing Log Information

ISAPI subroutines provide a mechanism that outputs log information to a log file. This log information is very useful because you can change the settings for it without changing an application. For information on settings, see *"ISAPI Subroutine Environment Variable Settings" in Chapter 4*. Note that this log information is a log of ISAPI subroutine activity only and is not useful for tracing the application itself. It should thus be used only as a guide. A log file has contents with the following structure:

```
process-ID thread-ID year-month-day hour: minute: second significance message
```

Specifically speaking, the contents are as follows:

```
0000000188 0000000187 1999-02-02 11:48:07 02 COB-06310:COBW3: Specified conversion name is already
registered. Conversion information already registered will be enabled.
0000000188 0000000081 1999-02-02 11:48:10 01 COB-04470:COBW3: Because specified conversion
information is not registered, it could not be changed.
```

To collect trace information of the actual application, use the TRACE function of COBOL. For details about the TRACE function, see the *"NetCOBOL Debugging Guide."*

5.2 Checking the Operation Using the Interactive Debugger

To debug a Web application created in COBOL using the debugger, start the debugger from the program to be debugged.

The Web application can be remotely debugged from the client which starts a WWW browser by using a remote debugger. Please refer to "NetCOBOL Debugging Guide" for the usage of a remote debugger.

Note the following points when you debug a Web application using the debugger because IIS reads and invokes Web applications in the same memory space.

- If a Web application performs an invalid operation, not only the Web application becomes abnormal but also other Web applications loaded at the same time or even IIS may become abnormal. It is recommended that you first debug a Web application while no other Web application is loaded and then later do so while some other Web applications are loaded.
- The debugger debugs not only a Web application but also other Web applications and even IIS. Since terminating the debugger also terminates other Web applications and IIS, you must restart IIS.

IIS provides ways of operating a Web application in the memory space of IIS (IIS process) or in a different memory space than IIS (isolated process or pool). Specifying a Web application as an isolated process or pool eliminates the above problem, thus facilitating the debugging process.



For IIS 7.0 or later, in order to run a Web application created in COBOL it is necessary to create an application pool. Please refer to the IIS Configuration steps (In case of IIS 8.5) and create application pool.

Neither the just-in-time debugging function nor the COBOL Error Report is tripped by an application error occurring in a Web application, which is detected and handled by IIS instead.

5.2.1 Starting the Debugger

To start debugging an application using the debugger:

1. Compile and link a Web application.

Create a Web application with the compile and link options used for debugging specified. For information on compiling and linking a Web application for debugging, see the "NetCOBOL Debugging Guide."

2. Set the run-time environment information.

Before starting the debugger from a program to be debugged, you need to set the following run-time environment information. For information on setting the run-time environment information, see "ISAPI Subroutine Environment Variable Settings" in Chapter 4.

@CBR_ATTACH_TOOL=TEST [start-parameter]

This specifies that you wish to start the debugger (TEST) when a COBOL application is executed.

Following "TEST", you can specify an optional debugger start parameter. For information on start parameters, see the "NetCOBOL Debugging Guide."

3. Log on to the computer on which the Web application to be debugged is to be executed. Log on as the administrator because you may need to stop and start IIS services during debugging.
4. Use a WWW Browser to start the Web application. The WWW Browser need not run on the computer on which the Web application is to be executed.
5. When the Web application is started, the debugger is automatically started. After the debugger has started, access the [Start Debugging] dialog to specify debugging information file storage folders and necessary information to start debugging.

5.2.2 Debugging

You can debug a Web application just like any regular COBOL application using the debugger. For information on using the debugger, see the "NetCOBOL Debugging Guide." Before debugging, be careful of the following:

- The debugger, when debugging is started, automatically interrupts execution at the entrance of the Web application, providing you with an opportunity for debugging operation. You need to set a breakpoint in advance to disable automatic interrupts occurring when the Web application, after responding to the WWW Browser, is executed by the next request. To interrupt the execution of a Web application loaded by the next request, specify the name of the program to be loaded at a command or in a dialog box to specify a breakpoint. For details, see "Debug functions for dynamic structured programs" in the "NetCOBOL Debugging Guide".
- The debugger cannot detect an application error (exception) occurring in a Web application, which is detected and handled by IIS instead. To have the debugger detect an application error, access the debugger's [Environments] dialog and then the [Operation] page and check [Break execution at the first signal]. For details, see "Debug functions for signals handled by the exception handler" in the "NetCOBOL Debugging Guide".
- A Web application, after responding to the WWW Browser, is running because it waits for the next request. To perform the debug operation including setting a breakpoint, select the [Break] command from the debugger's menu to interrupt the execution of a program before doing so.
- Note also the following precaution on debugging:
- While you are debugging a Web application using the debugger, any operation of the Web application takes longer than usual. Thus, a timeout may occur during debugging depending on the WWW Browser. Set an appropriate timeout value for the WWW Browser if possible. Additionally, set the IIS connection timeout value to an appropriate one.

5.2.3 Terminating the Debugger

Terminating the Debugger First

- If a Web application is an isolated process or pool
You need not restart IIS because terminating the debugger terminates only the process of the Web application and not IIS itself.
- If a Web application is an IIS process
You need to restart IIS because terminating the debugger terminates IIS itself.

Terminating a Web Application First

- If the Web application is an isolated process or pool

Use Internet Service Manager to open [Property] of a virtual directory to which a Web application belongs and press the [Unload] button to terminate the process of the Web application. When you terminate the process of the Web application, the Web application must be running until the process is terminated.

- If the Web application is an IIS process

Stop IIS.

Since IIS does not completely stop using Internet Service Manager, open the services to stop all IIS services (IIS Admin Service and the services dependent on it). When you stop IIS, the Web application must be running until the process is terminated.

5.2.4 Swapping Web Applications

Swap Web applications as follows:

- If a Web application is an isolated process or pool

Terminate the process of the Web application before swapping it with another Web application.

To terminate the process of the Web application, use Internet Service Manager to open [Property] of a virtual directory to which a Web application belongs and press the [Unload] button. When you terminate the process of the Web application, the Web application must be running until the process is terminated. If you cannot terminate the process of the Web application this way, do as shown in the section, "If a Web application is an IIS process".

- If a Web application is an IIS process

Stop IIS before swapping Web applications and then restart IIS.

Since IIS does not completely stop using Internet Service Manager, open the services to stop all IIS services (IIS Admin Service and the services dependent on it). When you stop IIS, the Web application must be running until the process is terminated.

To start IIS, either use Internet Service Manager or open the services and start all IIS services (IIS Admin Service and the services dependent on it).

5.3 Referencing an Error Detected by an ISAPI Subroutine

To reference an error detected by an ISAPI subroutine, set the debug mode just before invoking "COBW3_INIT" in a program.

```
:  
SET COBW3-DMODE-DBG TO TRUE  
CALL "COBW3_INIT" USING COBW3.  
:
```

This setting displays an error message in the WWW Browser if an error is detected by an ISAPI Subroutine.

Anything other than COBW3-CONTENT-TYPE-NON defined for the header output is assumed as COBW3-CONTENT-TYPE-HTML and COBW3_INIT declare the Content-type.

To set the debug mode, you need to modify a program and re-create the Web application. Depending on the debug mode setting, you can reference an error message displayed in the WWW Browser in the log information also. Thus, it is recommended to use log information to reference error messages without changing an application.



Note

If the Web application has Unicode as the operation code, the debug mode setting is disabled. Check the operation by, for example, referencing the log information.

5.4 Referencing the Data Being Executed in a Display Format

To reference the data being executed in a display format, use COBW3_PUT_TEXT. Specify data that you want to reference using COBW3_PUT_TEXT and reference any data on a WWW Browser to perform debugging.

Note

Debugging using COBW3_PUT_TEXT cannot display data not in a display format (e.g., binary data). The debugging operation tends to have an inferior efficiency because compilation and execution are frequently repeated. In such a case, it is recommended to debug the data being executed by referencing it using the interactive debugger.

Chapter 6 Sample Programs

This chapter describes the sample programs that are provided.

This sample shows how to use ISAPI subroutines to inherit data between applications using a Cookie or obtain WWW Server and WWW Browser information.

6.1 Provided Programs

This sample provides the following programs and related files.

- ISAMAIN.cob (COBOL source program)
- ISAINIT.cob (COBOL source program)
- ISATERM.cob (COBOL source program)
- ISASTART.htm (Web page for invoking application)
- ISARPLY1.htm (Web page for processing result)
- ISARPLY2.htm (Web page for processing result)
- ISAERROR.htm (Web page for processing result [for error handling])
- ISASMPL1.def (Module definition file)
- COBOL85.cbr (Run-time initialization file)

6.2 ISAPI Subroutines Referenced

This sample references the following ISAPI subroutines:

- COBW3_INIT
- COBW3_SET_CNV
- COBW3_PUT_HTML
- COBW3_RECEIVE_HEADER
- COBW3_GET_REQUEST_INFO
- COBW3_SET_COOKIE
- COBW3_GET_COOKIE
- COBW3_FREE

6.3 Compiling the Programs

Compile the programs using the COBOL32 command as shown below. You need not add the -I option if you have copied the libraries (COBW3.cbl, ISAPIINF.cbl, ISAPICTX.cbl, and ISAPIFLG.cbl) to be used by the ISAPI Subroutines in the folder that stores the sample. Note that the COBOL installation folder is assumed to be "C:\COBOL".

```
COBOL32-IC:\COBOL -WC, "ALPHAL(WORD), THREAD(MULTI)" ISAINIT.cob
COBOL32-IC:\COBOL -WC, "ALPHAL(WORD), THREAD(MULTI)" ISAMAIN.cob
COBOL32-IC:\COBOL -WC, "ALPHAL(WORD), THREAD(MULTI)" ISATERM.cob
```

6.4 Linking the Programs

Link the programs to create a DLL as follows:

```
LINK /dll /out:ISASMP1.dll ISAMAIN.obj ISAINIT.obj ISATERM.obj
C:\COBOL\F3BICBDM.obj F3BISAPI.lib F3BICIMP.lib
kernel32.lib MSVCRT.lib /ENTRY:COBDMAN /def:ISASMP1.def
```

6.5 Environment Settings

Modify the run-time initialization file (COBOL85.cbr) according to the desired environment. Register on IIS a virtual directory with the folder that stores the sample as the physical path. For information on registering a virtual directory on IIS, see *"IIS Settings" in Chapter 4*.

6.6 Executing the Sample

Check that IIS is running, access a WWW Browser and specify the URL that consists of a virtual directory registered in *"Environment Settings"* to display a Web page for invoking the application (ISASTART.htm). Then, do as you are instructed on the screen and press the "Submit" button.



This sample does not run correctly if the WWW Browser does not support COOKIES or if the WWW Browser is configured not to accept COOKIES.

6.7 Explanation of the Sample

To explain the sample, the following shows a Web page for invoking the application (ISASTART.htm), Web pages for processing results (ISARPLY1.htm and ISARPLY2.htm) and a COBOL program (ISAMAIN.cob) used in the sample. Note that only a COBOL program with the entry name HttpExtensionProc (ISAMAIN.cob) is shown. The programs with the entry names GetExtensionProc and TerminateExtension (ISAINIT.cob and ISATERM.cob) are omitted because they are the same as the models shown in *Chapter 2 "GetExtensionVersion"* and *"TerminateExtension"*.

Web page for invoking application (ISASTART.htm)

```
<HTML>
<HEAD>
  <TITLE>Start screen</TITLE>
</HEAD>

<BODY>
This is a sample which is used COBOL ISAPI subroutine.<BR>
If you confirm the operation, please press "Execute" button.<BR>
If you exit, please close this Browser.<BR>

<FORM METHOD="GET" ACTION="isasmpl1.dll">
  <INPUT TYPE="SUBMIT" NAME="GO" VALUE="Execute">
</FORM>
</BODY>
</HTML>
```

Web page for processing result (ISARPLY1.htm)

```
<HTML>
<HEAD>
  <TITLE>First Access Screen</TITLE>
</HEAD>

<BODY>
<FONT COLOR=BLUE>Thank you very much for having the first time</FONT><BR>
If you use continuously, please press "Execute" button.<BR>
```

```

If you exit, please close this Browser.<BR>

<FORM METHOD="GET" ACTION="isasmpl1.dll">
  <INPUT TYPE="SUBMIT" NAME="GO" VALUE="Execute">
</FORM>
</BODY>
</HTML>

```

Web page for processing result (ISARPLY2.htm)

```

<HTML>
<HEAD>
  <TITLE>Screen since first time</TITLE>
</HEAD>

<BODY>
Current situation is as follows.<BR>
<TABLE BORDER=2>
<TR>
  <TH>Hostname for accessed host</TH>
  <TH>Browser name</TH>
  <TH>Access count</TH>
</TR>
<TR>
  <TD>//COBOL//Hostname//COBOL//</TD>
  <TD>//COBOL//Browser Name//COBOL//</TD>
  <TD>//COBOL//Access count//COBOL//</TD>
</TR>
</TABLE>
<BR>

If you use continuously, please press "Execute" button.<BR>
If you exit, please close this Browser.<BR>
(Caution: When the Browser is closed, the access counter is reset.
Moreover, the counter value is different when accessed by a different
Browser or by a Browser on another machine.)

<FORM METHOD="GET ACTION="isasmpl1.dll">
  <INPUT TYPE="SUBMIT" NAME="GO" VALUE="Execute">
</FORM>
</BODY>
</HTML>

```

COBOL program (ISAMAIN.cob)

```

000010*-----
000020* All Rights Reserved, Copyright(C) FUJITSU LIMITED 1999-2002 *
000030* *
000040*  Filename: ISAMAIN.cob *
000050*  Abstract: Example for ISAPI subroutine *
000060*-----*
000070 identification division.
000080 program-id. "HttpExtensionProc".
000090 environment division.
000100 data division.
000110 working-storage section.
000120     copy COBW3.
000130*
000140 01 HTMLFilename          pic X(64).
000150 01 pathName              pic X(256).
000160 01 pathSize              pic 9(05).
000170 01 copyStartPos          pic 9(05).

```

```

000171 01 leftLength          pic 9(05).
000180 01 AccessCounter      pic 9(05).
000190*
000200 linkage section.
000210     copy IsapiCtx.
000220*
000230 procedure division with stdcall linkage using ISAPI-CTX-CNT.
000240*
000250 IsapiSample1-Start.
000260*
000270*  Initalize the work area for the ISAPI subroutine
000280     move low-value to COBW3.
000290     move function addr(ISAPI-CTX-CNT) to COBW3-CONTEXT.
000300*
000310*  Initialize the ISAPI subroutine work environment and obtain
000320*  a Web parameter
000330     call "COBW3_INIT" using COBW3.
000340*
000350     move space to pathName.
000360*
000370     move "Your Access Counter" to COBW3-COOKIE-NAME.
000380     call "COBW3_GET_COOKIE" using COBW3.
000390     if program-status not = zero then
000400         move "ISAERROR.htm" to HTMLFilename
000410         perform outputScreenProc
000420     else if COBW3-SEARCH-FLAG-NON then
000430         move 1 to COBW3-COOKIE-VALUE
000440         perform entryAccessCounterProc
000450         move "ISARPLY1.htm" to HTMLFilename
000460         perform outputScreenProc
000470     else
000480         perform outputContinuousScreenProc
000490     end-if.
000500*
000510 Finish-Pos.
000520*
000530*  Release the resources obtained by the ISAPI subroutine
000540     call "COBW3_FREE" using COBW3.
000550*
000560 IsapiSample1-End.
000570     move 1 to program-status.
000580     exit program.
000590*
000600 outputContinuousScreenProc section.
000610*  Registration of various conversion data
000620*  Get the Cookie data for AccessCounter
000630     compute AccessCounter = function NUMVAL(COBW3-COOKIE-VALUE).
000640     add 1 to AccessCounter.
000650     move AccessCounter to COBW3-COOKIE-VALUE.
000660     move zero to COBW3-COOKIE-VALUE-LENGTH.
000670*
000680*  The access counter value is registered in the conversion data.
000690     perform entryAccessCounterProc.
000700     move "Access Count" to COBW3-CNV-NAME.
000710     move AccessCounter to COBW3-CNV-VALUE.
000720     perform entryConversionDataProc.
000730*
000740*  Get and register remote hostname
000750     set COBW3-REMOTE-HOST to true.
000760     call "COBW3_GET_REQUEST_INFO" using COBW3.
000770     if program-status not = zero then
000780         move "ISAERROR.htm" to HTMLFilename
000790         perform outputScreenProc

```

```

000800         go to Finish-Pos
000810         end-if.
000820         move "Hostname" to COBW3-CNV-NAME.
000830         move COBW3-REQUEST-INFO to COBW3-CNV-VALUE.
000840         perform entryConversionDataProc.
000850*
000860* Gett and register Browser name
000870         move "User-Agent" to COBW3-HEADER-NAME.
000880         call "COBW3_RECEIVE_HEADER" using COBW3.
000890         if program-status not = zero then
000900             move "ISAERROR.htm" to HTMLFilename
000910             perform outputScreenProc
000920             go to Finish-Pos
000930         end-if.
000940         move "Browser Name" to COBW3-CNV-NAME.
000950         move COBW3-HEADER-VALUE to COBW3-CNV-VALUE.
000960         perform entryConversionDataProc.
000970*
000980* Output prototype HTML file
000990         move "ISARPLY2.htm" to HTMLFilename.
001000         perform outputScreenProc.
001010*
001020 outputContinuousScreenProc-End.
001030         exit.
001040*
001050*
001060 entryAccessCounterProc section.
001070* After the Browser ends, the content of the access counter can be left by setting
001080* an expiration. However, if the Browser is different, it is not significant
001090         call "COBW3_SET_COOKIE" using COBW3.
001100         if program-status not = zero then
001110             move "ISAERROR.htm" to HTMLFilename
001120             perform outputScreenProc
001130             go to Finish-Pos
001140         end-if.
001150 entryAccessCounterProc-End.
001160         exit.
001170*
001180 entryConversionDataProc section.
001190         call "COBW3_SET_CNV" using COBW3.
001200         if program-status not = zero then
001210             move "ISAERROR.htm" to HTMLFilename
001220             perform outputScreenProc
001230             go to Finish-Pos
001240         end-if.
001250 entryConversionDataProc-End.
001260         exit.
001270*
001280 outputScreenProc section.
001290* Get the physical path containing your application and
001300* edit the HTML document name.
001310         if pathName = space then
001320             perform getPhysicalPath
001330         end-if.
001340         move space to COBW3-HTML-FILENAME.
001350         move pathName(1:pathSize) to COBW3-HTML-FILENAME.
001360         compute copyStartPos = pathSize + 1.
001370         move "\" to COBW3-HTML-FILENAME(copyStartPos:1).
001380         compute copyStartPos = copyStartPos + 1.
001381         compute leftLength = 256 - copyStartPos.
001390         move HTMLFilename to COBW3-HTML-FILENAME(copyStartPos:256).
001400*
001410* Output HTML document

```



```

001420    call "COBW3_PUT_HTML" using COBW3.
001430*
001440 outputScreenProc-End.
001450    exit.
001460*
001470 getPhysicalPath section.
001480    move space to pathName.
001490    set COBW3-PHYSICALPATH to true.
001500    call "COBW3_GET_REQUEST_INFO" using COBW3.
001510    if COBW3-STATUS = zero then
001520        move COBW3-REQUEST-INFO to pathName
001530        move COBW3-REQUEST-INFO-LENGTH to pathSize
001540    end-if.
001550*
001560 getPhysicalPath-End.
001570    exit.
001580

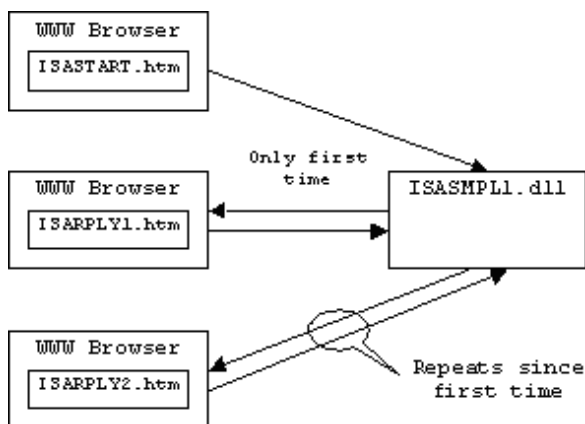
```

This sample does the following processing:

- Retrieves Cookie data
Retrieves Cookie data to be sent from a WWW Browser.
- Registers Cookie data
- Outputs a Web page for processing results

A different Web page for processing results is output according to the Cookie data. Additionally, the application registers conversion data as required to edit the Web page for processing results.

The screen and processing changes as follows:



The following section briefly describes using various functions in this sample and the reasons for using them.

6.7.1 Retrieving Cookie Data

A Cookie used in this sample has the name "Your Access Counter". Thus, to retrieve this Cookie data, edit the above Cookie name in COBW3-COOKIE-NAME and invoke "COBW3_GET_COOKIE" (Lines 370 through 380).

This sample uses the existence of Cookie data to determine the first time (whether this is the first-time startup from ISASTART.htm) and to keep the access count. The first-time processing is performed by utilizing the fact that no Cookie data exists and thus invoking COBW3_GET_COOKIE finds no Cookie data (Lines 420 through 460). On the other hand, the second time and later it is determined by utilizing the fact that Cookie data is always sent for the second time and later (Lines 470 through 480). Note that, in the first-time processing, the counter value is set to 1 and, for the second time and later, incremented by 1 every time.

6.7.2 Registering Cookie Data

For a Cookie that you want to register, define the name in COBW3-COOKIE-NAME and the contents in COBW3-COOKIE-VALUE and invoke COBW3_SET_COOKIE. In this sample, the Cookie data includes only "Your Access Counter" and the value first defined is used throughout (Line 370). The Cookie data thus registered is sent to the WWW Browser when a Web page for processing results is output.

6.7.3 Retrieving Request Information

Retrieving request information is simple. Just invoke COBW3_GET_REQUEST_INFO while specifying the condition name of the information that you want to retrieve. The information, if successfully retrieved, is set in COBW3-REQUEST-INFO.

This sample retrieves two pieces of request information: One is the information on a physical path corresponding to a virtual directory, used to determine the pathname of the Web page for processing results. This will be stored in the same path as the application (ISASMPL1.DLL) (Lines 1480 through 1540). Since a Web application has an undefined current directory on IIS, you need to determine the pathname based on this information or specify an absolute path. Another piece of information to be retrieved is the name of the host on which the WWW Browser is running, which is to be displayed on the screen (Lines 750 through 810).

6.7.4 Retrieving Header Information

Retrieving header information is also simple. Just set the name of an HTTP header that you want to retrieve in COBW3-HEADER-NAME and then invoke COBW3_RECEIVE_HEADER. The HTTP header information, if successfully retrieved, is set in COBW3-HEADER-VALUE.

This sample uses this function to retrieve information in the WWW Browser, which is to be displayed on the screen. The information in the WWW Browser can be retrieved from the "User-Agent" header (Lines 870 through 930).

Appendix A Questions and Answers

This appendix describes problems that you may encounter while using ISAPI Subroutines through questions and answers.

Q1.

Why was an error message displayed in the WWW Browser?

A1.

Check the WWW Server settings and whether the program name to start the Web application and that written in the FORM tag (ACTION) of the HTML document (Web page for invoking application) are identical.

Also, check whether the WWW Server settings, the program name, and the HTML document (Web page for processing result output) are stored at the correct locations. Furthermore, check the export function names, translation options, and contents of the module definition file.

Q2.

Is it necessary to protect data from outsiders?

A2.

Data in communication between the WWW Server and client may be referenced or changed by outsiders. To prevent such acts that could affect operations of a system, we recommend tasking necessary measures for network security such as SLL (Secure Socket Layer).

Consult with your WWW Server administrator.

Q3.

A file cannot be I/O correctly in a Web application.

A3.

To I/O a file, it is necessary to grant rights to I/O permission the file and the folder in which the file is stored to the user ID that executes the Web application.

Q4.

Is there any means to change the display contents of a Web application dynamically?

A4.

1. Using COBW3_SET_CNV, and the like

Write *//COBOL//conversion name//COBOL//* in the variable part of the Web page for processing result output. The page can be changed dynamically by substituting the conversion character string by using *COBW3_PUT_HTML* after registering the conversion name and a substitution character string (conversion character string) with such COBOL Web subroutine as *COBW3_SET_CNV*.

2. Using COBW3_PUT_TEXT.

The contents that are specified in such an output file as HTML document can usually be specified in *COBW3_PUT_TEXT*.

Thus, if output data is provided for each condition in the program, the page can be changed dynamically by specifying a data name in *COBW3_PUT_TEXT*.

3. Dividing a page into two or more files

An output file can be divided into two or more files.

For example, call *COBW3_PUT_HTML* after specifying the name of the file that describes the former half of a page to be output. For the latter half, prepare some files written adjusting to the conditions and change the file name to be output according to the condition in the program before calling *COBW3_PUT_HTML* again.

As is noted above, the latter half of a page can dynamically be changed.

In addition, by creating a page using components, it becomes unnecessary to prepare multiple similar files.

In addition, by combining the methods of 1) to 3) Above, pages that correspond to a variety of situations can be created.

Q5.

Why is a Web application not working correctly after starting it?

A5.

Check whether *COBW3_INIT* is specified at the start of the Web application. Alternatively, check whether at least one NAME is set within the page that starts the Web application.

If *COBW3_INIT* is specified, collect error information and do debugging while referencing "Operation Check" for investigation.

Q6.

The following message was displayed on the WWW Browser. Why? Server Error

The WWW Server encountered an internal error or mis-configuration and was unable to complete your request.

Or,

Specified Web application has returned only part of the Web header.

A6.

An error in Content-type can be assumed. Check the value of *COBW3-CONTENT-TYPE* specified in *COBW3_PUT_HEAD*.

1. When an HTML document (Web page for processing the resulting output) is output

Omit the *COBW3_CONTENT_TYPE* value (LOW-VALUE) that does not call *COBW3_PUT_HEAD* or call *COBW3_PUT_HEAD* after specifying *COBW3_CONTENT_TYPE_HTML*.

2. When text data is output

Call *COBW3_PUT_HEAD* after specifying *COBW3_CONTENT_TYPE_TEXT*.

Q7.

Why is the Content-type declaration displayed on the WWW Browser?

A7.

If *COBW3-DMODE-DBG* is specified in *COBW3_INIT*, header information such as Content-type is displayed in the WWW Browser as debug information.

Such information will not be displayed if *COBW3-DMODE* is not specified.

Q8.

A message "Do you want to save the file in disk?" was displayed on the WWW Browser after executing a Web application. Why?

A8.

Messages such as this may be output if the Content-type declaration of a Web application is not correct.

Check whether the Content-type declaration is correct.

Q9.

What should be kept in mind when creating a Web application with COBOL programs?

A9.

Most of the COBOL functions can be used in the Web application. However, notice that the following functions related to screen operations cannot be used.

- Presentation file module (Screen handling module)
- Screen handling module
- ACCEPT/DISPLAY function

(The environment variable, date, and time manipulation functions can be used.)

For details about receiving and referencing Web parameters and outputting processing results, see "How to Use the ISAPI Subroutines" In Chapter 3.

Q10.

How do you set the environmental variables required for starting the WWW Server?

A10.

- For Windows® 2000

Add the necessary environment variables in the [System Environment Variables (S):] in [Environment Variables (E)] in the [Details] sheet in the [System] folder in the [Control Panel] folder which can be displayed from the [Setup] menu in the [Start] button, then restart the WWW Server.

Otherwise, place the execution initialization file (COBOL85.CBR) in which the necessary execution environment information is specified in the same folder as for Web applications (.Dll).

Q11.

Can the COBOL debugger be used?

A11.

Yes, you can debug an application using the COBOL debugger in a Web environment. See *"Checking the Operation Using the Interactive Debugger" in Chapter 5.*

Q12.

The status code (Status-code) was displayed on the WWW Browser from the WWW Server. What does it mean?

A12.

See the basics of HTTP described in Appendix A, *"For Beginners Developing COBOL Web Applications"* in the *"NetCOBOL Web Guide"*.

Q13.

Is it possible to control the timeout period?

A13.

It is a function of the WWW Server. Refer to the manuals of the WWW Server.

Depending on the WWW Server, the timeout period may be controllable by changing the environment settings or based on HTTP header information from the Web application side.

Q14.

Is it possible to use an HTML document with frames in a COBOL Web application?

A14.

There is no need to write special settings or processing in a COBOL Web application to use the frame function.

If the frame function is supported by the WWW Browser, only the description of frames in HTML is needed.

Q15.

Why can the value in VALUE specified by the <INPUT> tag using *COBE3_CHECK_VALUE* not be retrieved correctly?

A15.

If NAME is omitted in the INPUT tag, the value of VALUE in the INPUT tag may not be entered in the Web parameter depending on the WWW Browser. Be sure to specify NAME if the INPUT tag needs the value of VALUE.

Q16.

Why is the retrieval using *COBW3_GET_VALUE* or *COBW3_CHECK_VALUE* not correct?

A16.

If the length of a character string for retrieval contained in the Web parameter exceeds a preset value, retrieval processing is performed by the Web subroutines within the limit of the string length. Therefore, expected values may not be obtained.

Q17.

Why is an error "F3BIPRCT.DLL cannot be found" displayed on the WWW Server?

A17.

Check whether the COBOL run-time system is installed correctly. When a Web application is operating, specified user environmental variables will not be enabled.

Set in advance the install folder of the COBOL run-time system to the system environmental variable PATH of the WWW Server machine. At this time, it is necessary to restart the WWW Server to enable the settings of system environmental variables.

Q18.

Why can a file specified using an absolute path not be accessed?

A18.

When specifying the storage location of COBOL files used for executing a Web application, it is necessary to specify the location according to the drive configuration of the WWW Server machine to be used.

Q19.

Can resources of the network environment be accessed?

A19.

When a Web application is executed, the configuration of network drives when the user logged in is not enabled. Access resources of the network environment to be used for execution by specifying UNC. For details, refer to the "*NetCOBOL User's Guide*".

Q20.

Why was an error "No data is contained in document" displayed on the WWW Browser?

A20.

Be sure to start a new line after the </HEAD> tag of an HTML document (Web page for processing the resulting output) output from a Web application.



Example

```

<HTML> -
<HEAD> -
xxx -
</HEAD> -
-
<BODY> -
xxx -
</BODY> -
</HTML> -

```

Q21.

What should you do if no response is received from the WWW Server?

A21.

It is possible that a window or message box is waiting for input on the WWW Server.

To execute a COBOL program on the WWW Server, write the following environmental variable information in the initialization file for execution or set it in the system environmental variables.

No window or message box will be displayed after specifying this environmental variable information and so the input wait state of the operator can be avoided.

Environment variable information	Meaning
@MessOutFile=file-name	Output all COBOL run-time messages to a specified file.
@WinCloseMsg=OFF	Disable the display of a message appearing when a window is closed.

For details about an environment variable, see the "*NetCOBOL User's Guide*."

Additionally, do not use in a COBOL program the following screen I/O functions.

- Presentation file module (Screen handling module)
- Screen handling module
- ACCEPT/DISPLAY function

(The environment variable, date, and time manipulation functions can be used.)

For details, see *"NetCOBOL User's Guide"*.

Furthermore, check IIS or ODBC settings (if ODBC is used) because they may be incorrect.

Q22.

Why is the COBOL debugger not started?

A22.

Check the following:

- The environment variable, @CBR_ATTACH_TOOL=TEST is set.
- The COBOL Tool Attaching Service is started.

Q23.

The COBOL debugger is started, but debugging does not start.

A23.

Check the [Debugging Information] page in the [Start Debugging] dialog to see if the [Debug information file storage folder] is set correctly.

Q24.

A Web application that uses the session management function, if started twice, sometimes does not run correctly.

A24.

A Web application that uses the session management function, if started twice, runs unstably. To avoid this problem, use JavaScript, etc. in the WWW Browser to avoid starting such a Web application more than once. For general precautions on manipulating a WWW Browser, see *"NetCOBOL Web Guide"*.

Q25.

What state does the Web applications executed in WWW Server when the WWW Browser is canceled by some reasons?

A25.

The following causes are thought by the condition that the WWW Browser is canceled.

- The WWW Browser is closed.
- The Connected time-out function works.

In general, the session is started by the request from the WWW Browser between the WWW Browser and the Web application process in HTTP. And, the session ends at once when the Web application is completed, and the transmission of the response data to the WWW Browser is completed.

Therefore, it is necessary to configure the Web application which works in WWW Server to complete all processing in the HTTP session of one time (For example, OPEN->READ/WRITE->CLOSE of the file is done).

In this case, only the HTTP session between the WWW Browser and WWW Server interrupts even if the WWW Browser is canceled, and there is no problem in the Web application as mentioned above configured. Therefore, the Web application continues and is executed.

However, the error occurs if the response is transmitted by using COBW3_PUT_HTML and COBW3_PUT_TEXT after the HTTP session is interrupted.

When the session management function of the Web subroutine is used, the collection processing of various resources is necessary. The collection processing of the resource must use the termination procedure when the time-out is generated.

For example, a certain system assumes that files OPEN when requests first time and files CLOSE when finally requesting. At this time, there is danger to which file CLOSE is not done and the file is broken when the WWW Browser is canceled when the middle requests. Therefore, the Web application for the recovery processing executed at the time-out is made, and consideration by which file CLOSE in that is necessary.

For details, see *"Session Management Function" in Chapter 3.*

Appendix B Error Handling

Any errors detected while ISAPI Subroutines are executed are handled by ISAPI Subroutines.

An ISAPI subroutine outputs a message in the following format:

```
COB-message-number: COBW3: message-text
```

- COB-message-number: The serial number of a message is displayed.
- message-text: The error description is displayed.

If you have enabled the log setting for ISAPI Subroutines, the same contents are output to a log.

The following lists the message numbers and message texts to be output:

Message number	Message text
00100	The Processing was not able to be continued. Because, the value was not set in COBW3-CONTEXT.
01100	It is thought that the work area or the object was destroyed. Check the object of the Web subroutine and the usage of the area.
01500	The Web subroutine cannot operate correctly in the calling COBOL application operation code system.
01501	The operation state of the calling application is abnormal. The Web subroutine cannot operate normally.
02050	Search character string length specified as a negative value. The processing continues as if 0 was specified.
02051	A value was specified that exceeds the limit for the character string length of the search string. The processing continues assuming that the maximum length was specified.
03000	Web application was called by a method without GET or POST.
03001	COBW3_INIT was called two times or more. Please call COBW3_FREE a second time before the call.
03002	The work area of the Web subroutine could not be acquired.
03020	The data that was passed by POST or GET was not acquired successfully.
03030	Failed in the acquisition of the work area of the Web subroutine.
03040	Uploaded file information could not be acquired.
03041	The Web parameter can't be processed because failed to acquire header information.
03042	multipart/form-data is specified for ENCTYPE of the FORM tag though the method is things except POST. The Web parameter and uploaded file information could not be acquired.
03045	An error in processing the output to the WWW Browser.
03065	Environment variable @CBR_WEB_OUT_CODE was set incorrectly. It is assumed that the conversion code was not set
03200	Conversion from UTF-8 to UCS-2 failed.
03201	Conversion from UCS-2 to UTF-8 failed.
03700	A code was specified that could not be recognized as a character. Processing continues.
03701	EUC and SJIS character strings are intermingled. Processing continues.
03850	Cookie was not acquired successfully.
04000	There is no Web parameter.
04001	The Web parameter length is 0.

Message number	Message text
04006	A negative value or "0" was specified for COBW3-NUMBER. The processing continues as if 1 was specified.
04008	The length of VALUE data exceeds the limit. The string is truncated to the supported length.
04009	The data passed with POST or GET method was nothing.
04400	The conversion name is not specified. Please set the conversion name.
04401	A negative value was specified for character string length of the conversion name. The processing continues as if 0 was specified.
04402	A value was specified that exceeds the limit for character string length of the conversion name. The processing continues assuming the maximum length was specified.
04450	Failed to delete the variable name specified in COBW3-CNV-NAME because it could not be found in the registered information.
04470	Failed to change the variable name specified in COBW3-CNV-NAME because it could not be found in the registered information.
04480	Failed to delete the variable name specified in COBW3-CNV-NAME.
05000	Failed in the execution of the system command.
05001	The system command is not specified.
05500	The file name is not specified. Please set the file name.
05501	There is no uploaded file information.
05510	Failed to acquire the path name in the client of the uploaded file.
05511	The character string length of the path name in the client of the uploaded file exceeded the limitation. The string is truncated to the supported length.
05520	Failed to acquire the file name in the client of the uploaded file.
05521	The character string length of the file name in the client of the uploaded file exceeded the limitation. The string is truncated to the supported length.
05530	Failed to acquire the content type of the uploaded file.
05531	The character string length of the content type of the uploaded file exceeded the limitation. The string is truncated to the supported length.
05550	The file was not generated because the specified file name had already existed.
05551	Failed to generate the file.
05552	Failed to generate data in the subroutine. The generated file can't be deleted with COBW3_DEL_UPLOADEDFILE.
05580	The specified file name is not a file generated in the same thread. The file was not deleted.
05581	Failed to delete the specified file.
06100	An I-O error occurred in the OPEN process of the specified HTML document file.
06101	An I-O error occurred in the READ process of the specified HTML document file.
06102	The length of the specified HTML document file is 0.
06105	A conversion name in the HTML document was not registered using COBW3_CNV_SET. Please register the conversion data using COBW3_CNV_SET.
06106	Failed in the output because the output work area of the HTML document was insufficient. Please confirm the conversion data length corresponding to the conversion name.
06107	Failed in the output because an error was found in the specified form of the conversion name within the HTML document.
06108	The string length of the specified conversion name is too long. Check the conversion name.

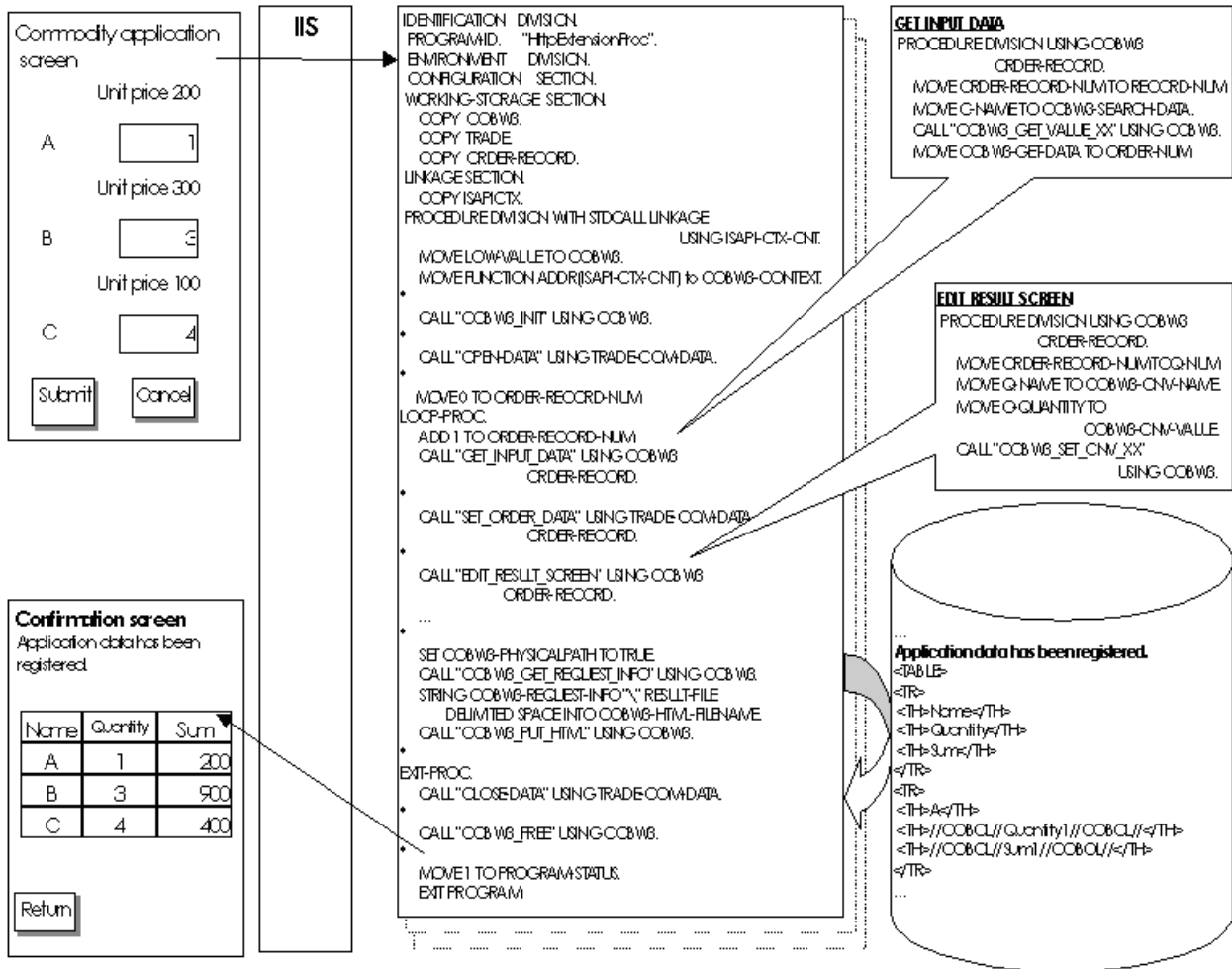
Message number	Message text
06120	Failed to output the HTML document to the WWW Server.
06130	The number of characters in one line of the HTML document exceeds the limit. The string is truncated to the supported length.
06300	The conversion name is not specified. Please set the conversion name.
06301	A negative value was specified for character string length of the conversion name. The processing is continued regarding for 0 to have been specified.
06302	A value was specified that exceeds the limit for character string length of the conversion name. The processing continues assuming the maximum length was specified.
06303	The conversion character string is not specified. Please confirm the conversion character string.
06304	A negative value was specified for character string length of the conversion character string. The processing continues assuming 0 was specified.
06305	A value was specified that exceeds the limit for character string length of the conversion character string. The processing continues assuming the maximum length was specified.
06310	The specified variable name has already been registered. The old conversion information remains effective.
06320	Failed in registration because the registration work area for the conversion information was insufficient. Please register again after calling COBW3_CNV_INIT and initializing conversion information.
06330	Failed to delete conversion information.
06502	Unexpected error occurred in writing data on standard output.
06503	A negative value was specified for the length of the string to be output. The processing continues assuming 0 was specified.
06504	A value was specified that exceeds the limit for output character string length. The processing continues assuming the maximum length was specified.
06700	COBW3_PUT_HTML or COBW3_PUT_TEXT has already been called. The output request for the header is invalid.
06704	A negative value was specified for COBW3-PUT-HEAD-LENGTH. The processing continues assuming 0 was specified.
06706	A value was specified that exceeded the limit for COBW3-PUT-HEAD-LENGTH. The processing continues assuming the maximum length was specified.
06708	The object output as a header was not specified.
06710	A value that had already been output and a different value were specified for COBW3-CONTENT-TYPE.
06714	A value that had already been output and a different value were specified for COBW3-STATUS-CODE.
06730	An error not anticipated in the declaration of Content-type occurred.
06734	An error not anticipated in the declaration of Status-code occurred.
06736	Unexpected error occurred in declaration of specified header.
06742	Failed in the output of the specified header.
06746	The output of the header could not be completed.
07010	The user name was not able to be acquired.
07011	The character string length of the user name exceeded the limitation. The maximum length is made effective.
07020	The password was not able to be acquired.

Message number	Message text
07021	The character string length of the password exceeded the limitation. The maximum length is made effective.
07030	The IP address was not able to be acquired.
07031	The character string length of the IP address exceeded the limitation. The maximum length is made effective..
07040	The authentication type was not able to be acquired.
07041	The character string length of the authentication type exceeded the limitation. The maximum is made effective.
07500	Session not started.
07510	Unable to start a session because COBW3_PUT_HTML or COBW3_PUT_TEXT already invoked.
07520	Unable to end a session because COBW3_PUT_HTML or COBW3_PUT_TEXT already invoked.
07558	Unable to start the timeout monitoring.
07562	Failure in the timeout monitoring for some reason.
07566	Failure in interrupting the timeout monitoring.
07570	Unable to prepare for the timeout monitoring for some reason. Unable to start a session.
07580	Unable to interrupt the timeout processing.
07584	Incorrect session ID got from Cookie data.
07588	Failure in getting the session information management area.
07590	Failure in executing the timeout monitoring.
07600	Session already started.
07602	Unable to start a session because the timeout time is specified as 0.
07604	Value exceeding the restriction specified for the timeout time. The processing is continued assuming that the maximum length is specified.
07606	Unable to start a session because an invalid value is specified in the session data type.
07608	Unable to start a session because of a failure in creating Cookie data for the session.
07610	Unable to register session data because 0 was specified for the session data size.
07614	Value exceeding the restriction specified for the session data size. The processing is continued assuming that the maximum length is specified.
07618	Incorrect object specified for the session data.
07650	Unable to get session data because a mismatch exists between the specified and registered session data sizes.
07660	Unable to get session data because a value other than null is set in the session data.
07670	No session data registered.
07680	Unable to change the timeout time because 0 was specified for the timeout time.
08004	A negative value was specified for COBW3-HEADER-NAME-LENGTH. The processing continues assuming 0 was specified.
08005	The script-name could not be acquired.
08006	A value was specified that exceeded the limit for COBW3-HEADER-NAME-LENGTH. The processing continues assuming the maximum length was specified.
08008	COBW3-HEADER-NAME is not specified.
08010	Failed to acquire the HTTP header.

Message number	Message text
08020	The length of COBW3_HEADER_VALUE exceeds the limit. The string is truncated to the supported length.
08024	The HTTP header specified for COBW3_HEADER_NAME could not be found.
08500	COBW3_PUT_HTML or COBW3_PUT_TEXT has already been called. Operation to cookie is invalid.
08510	The cookie name is not specified. Set the cookie name.
08512	A negative value was specified for the string length of the cookie name. The processing continues assuming 0 was specified.
08514	A value was specified that exceeded the limit for the string length of the cookie name. The processing continues assuming the maximum length was specified.
08520	The cookie value is not specified. Set the cookie value.
08522	A negative value was specified for the string length of the cookie value. The processing continues assuming 0 was specified.
08524	A value was specified that exceeded the limit for the string length of the cookie value. The processing continues assuming the maximum length was specified.
08530	Failed to change the specified cookie data because it was not registered.
08532	The specified cookie name has already been registered. The old cookie data remains effective.
08540	A date before Oct 15, 1582 is not valid as a cookie period. The period is assumed not to have been specified.
08541	The year for the cookie period is not specified correctly. The period is assumed not to have been specified.
08542	The month for the cookie period is not specified correctly. The period is assumed not to have been specified.
08543	The day for the cookie period is not specified correctly. The period is assumed not to have been specified.
08544	The hour for the cookie period is not specified correctly. The period is assumed not to have been specified.
08545	The minute for the cookie period is not specified correctly. The period is assumed not to have been specified.
08546	The second for the cookie period is not specified correctly. The period is assumed not to have been specified.
08550	An invalid value was specified for the cookie security. The security is assumed not to have been specified.
08560	An invalid value was specified for the cookie registration type. The registration type is assumed not to have been specified.
08570	Failed in registration because the registration work area for the cookie data was insufficient. Please register again after calling COBW3_INIT_COOKIE to initialize conversion information.
08610	Failed to delete the specified cookie data because it was not registered.
08620	Failed to delete the specified cookie data.
08710	An invalid value was specified for the initialization type of cookie information. The initialization type is assumed not to have been specified.
08720	Failed in initialization of the request data by cookie information because the registration work area for the cookie information was insufficient.
08810	The length of the cookie value exceeded the limit. The string is truncated to the supported length.
08820	Cookie data was not sent from the client.

Message number	Message text
09001	The correct value is not specified for COBW3_REQUEST_INFO_TYPE.
09010	Failed to acquire the request.
09020	The length of COBW3_REQUEST_INFO exceeded the limit. The string is truncated to the supported length.
09500	COBW3_CNV_SET can't be used in the Unicode environment.
09501	COBW3_CNV_DEL can't be used in the Unicode environment.
09502	COBW3_NAME can't be used in the Unicode environment.
09504	COBW3_VALUE can't be used in the Unicode environment.
09520	This subroutine can't be used in the ASCII environment.
99999	Can't open message catalog.

Appendix C Concept Diagram of Creating Internet Server Applications in COBOL



Appendix D CGI to ISAPI Subroutines Conversion Guide

This manual describes how to convert a Web application that uses CGI Subroutines into a Web application that uses COBOL ISAPI Subroutines.

D.1 Conversion from CGI to ISAPI

Because COBOL ISAPI subroutines basically guarantee upward compatibility of sources for the COBOL CGI subroutines, migration of Web Applications created with COBOL CGI subroutines to those created with COBOL ISAPI subroutines is easy. However, there are some ISAPI specific functions, and so some program corrections and changes of the Compilation and link methods are needed. The following describes some points to be noted for migration.

First, the following table lists the differences between applications created using COBOL CGI subroutines (hereafter referred to as CGI applications) and those created using the COBOL ISAPI subroutines (hereafter referred to as ISAPI applications).

No.	Item	CGI application	ISAPI application
1	Application format	Main program (EXE)	Subprogram (DLL)
2	Run unit	Process	Thread
3	Module configuration	Free	Prepare GetExtensionVersion, HttpExtensionProc, and TerminateExtension as export functions, and otherwise free..
4	Interface area with a WWW Server	Not required	Required
5	Return code	Not required or free	Required. The values have fixed meanings.
6	Manipulation of environment variables	Arbitrary	Limited
7	Manipulation of CGI environment variables	Can be manipulated using a provided subroutine or the DISPLAY/ACCEPT statements.	Can be manipulated using a provided subroutine.

To convert a CGI application into an ISAPI application, you need to modify the above differences. The details are described in "*Application Formats*" and later.

Note that the following subroutines, although provided as COBOL ISAPI subroutines to maintain compatibility, can be used only in an environment where a Web application runs in the ASCII code system. It is recommended to use the conversion to ISAPI as an opportunity to replace these subroutines with the alternative ones.

Old Subroutine name	New Alternative subroutine
COBW3_NAME	COBW3_GET_VALUE COBW3_GET_VALUE_XX COBW3_GET_VALUE_NX COBW3_GET_VALUE_XN COBW3_GET_VALUE_NN
COBW3_VALUE	COBW3_CHECK_VALUE COBW3_CHECK_VALUE_X COBW3_CHECK_VALUE_N

Old Subroutine name	New Alternative subroutine
COBW3_CNV_SET	COBW3_SET_CNV COBW3_SET_CNV_XX COBW3_SET_CNV_NX COBW3_SET_CNV_XN COBW3_SET_CNV_NN
COBW3_CNV_DEL	COBW3_DEL_CNV COBW3_DEL_CNV_X COBW3_DEL_CNV_N
COBW3_CNV_INIT	COBW3_INIT_CNV

D.2 Application Formats

Since CGI and ISAPI provide different application formats, you need to modify the following items:

- Description of the Web application to be invoked in a Web page use for invoking an application
- Compiling and Linking methods

D.3 Web Page for Invoking an Application

Change the extension of the Web application described in a Web page for invoking an application from EXE to DLL or, if required, change the entire application name.

In the following example where a CGI application is named CGI.EXE and an ISAPI application is named ISAPI.DLL, you need to make modifications as follows:

Web page for invoking a CGI application

```
<FORM ACTION="/sample/CGI.EXE">
...
</FORM>
```

Modified Web page for invoking an ISAPI application

```
<FORM ACTION="/sample/ISAPI.DLL">
...
</FORM>
```

D.4 Compilation and Linkage Methods

Compilation

For CGI applications, you need no particular options to compile a program except for the "MAIN" compile option that you must specify for a main program. However, do not specify the "MAIN" compile option for an ISAPI application because it must be created as a DLL. Additionally, you need the following compile options for an ISAPI application. For details about compile options, see the *NetCOBOL User's Guide*.

- ALPHAL(WORD) or NOALPHAL
- THREAD(MULTI)

Be sure to specify the compile option ALPHAL(WORD) or NOALPHAL for a program with three entry-names as described in *Module Configuration*. These options are used to identify the case-sensitivity of an entry-name. On the other hand, specify the THREAD(MULTI) compile option for any source program because ISAPI applications run in multiple threads.

Linkage

For CGI applications, link object files to create an EXE file. The following shows an example.

```
LINK business-program.obj
      F3BICIMP.lib MSVCRT.lib F3BICWSR.lib
      /OUT:run-format-name.exe
```

An ISAPI application must be created as a DLL. It uses different import libraries than a CGI application. The following shows an example.

```
LINK /DLL initial-program.obj business-program.obj terminate-
      program.obj
      F3BICBDM.obj F3BISAPI.lib F3BICIMP.lib KERNEL32.lib MSVCRT.LIB
      /OUT: run-format-name.dll /DEF: module-definition.def
      /ENTRY:COBDMAIN
```

Describe the three export functions described in "*Module Configuration*" in the module definition file (module-definition.def). Specifically, describe as follows:

```
LIBRARY run-format-name
EXPORTS
      GetExtensionVersion
      HttpExtensionProc
      TerminateExtension
```

D.5 Run Unit

Converting a CGI application to an ISAPI application means converting a process-based application to a thread-based application. To do so, note the following:

- Compilation method
- Access to a shared resource

D.6 Compilation Method

You need to specify the compile option, THREAD(MULTI) to create a multithreaded object.

D.7 Access to a Shared Resource

Since a CGI application runs as one thread (single thread) in the process of the CGI application itself, there can't be a resource shared among multiple threads in the same process.

Since an ISAPI application runs as multiple threads (multi-thread) in the process of IIS itself, a resource can be shared among the multiple threads.

If a resource is shared among multiple threads in an ISAPI application, synchronization control must be performed on the shared resource so that no contention occurs for it. Depending on the shared resource, the synchronization control is performed either automatically by the COBOL run-time system or by the ISAPI application itself using the "thread synchronization control subroutine" provided in COBOL.

For details, see "*Multithread*" of the "*NetCOBOL User's Guide*".

- "Sharing a Resource among Threads"
- "Thread Synchronization Control Subroutine"

D.8 Module Configuration

The module configuration of a CGI application is not restricted in particular except that it needs a main program. For an ISAPI application, however, the following three modules must be defined.

- GetExtensionVersion

- HttpExtensionProc
- TerminateExtension

HttpExtensionProc corresponds to the main program of a CGI application. GetExtensionVersion and TerminateExtension, corresponding to nothing of a CGI application, are the functions used for the initialization and termination processing, respectively.

For details about GetExtensionVersion, HttpExtensionProc, and TerminateExtension, see "*GetExtensionVersion*" "*HttpExtensionProc*" and "*TerminateExtension*" In *Chapter 2*. Additionally, the main program must be named HttpExtensionProc. For example, modify the application as follows:

- Description in a CGI application

```
IDENTIFICATION DIVISION.
PROGRAM-ID     MAINPROG.
ENVIRONMENT    DIVISION.
```

- Modified description in an ISAPI application

```
IDENTIFICATION DIVISION.
PROGRAM-ID     "HttpExtensionProc".
ENVIRONMENT    DIVISION.
```

D.9 Interface Area with a Server

For a CGI application, you need not be aware in particular of interfaces (except environment variables) with a WWW Server. For an ISAPI application, however, HttpExtensionProc, invoked as a function by IIS, receives an interface (parameter) area to exchange data with IIS. Thus, to convert a CGI application to an ISAPI application, you need to modify the LINKAGE SECTION and the PROCEDURE DIVISION. Since this interface area is used by COBOL ISAPI Subroutines, you need to set the address in COBW3 that is to interface with COBOL ISAPI Subroutines. For example, modify the application as follows:

- Description in a CGI application

```
WORKING-STORAGE SECTION.
  COPY COBW3.
PROCEDURE DIVISION.
  MOVE LOW-VALUE TO COBW3.
```

- Modified description in an ISAPI application

```
WORKING-STORAGE SECTION.
  COPY COBW3.
LINKAGE SECTION.
  COPY ISAPICTX.
PROCEDURE DIVISION WITH STDCLASS USING ISAPI-CTX-CNT.
  MOVE LOW-VALUE TO COBW3.
  MOVE FUNCTION ADDR(ISAPI-CTX-CNT) TO COBW3-CONTEXT.
```

Return Code

The use of a return code is arbitrary in the main program of a CGI application. However, as described in "*Interface Area with a Server*", HttpExtensionProc of an ISAPI application is invoked as a function while IIS expects a return value from this function. Thus, for an ISAPI application, you need to set a return code in PROGRAM-STATUS. For example, modify the application as follows:

- Description in a CGI application

```
END-PROC
STOP RUN.
```

- Modified description in an ISAPI application

```
END-PROC
MOVE 1 TO PROGRAM-STATUS.
EXIT PROGRAM.
```

You can set the following two return codes. Normally, set a normal end. If you set an abnormal end, the WWW Server assumes that an error occurred in the concerned application and sends a corresponding message to the WWW Browser.

Return code	Meaning
1	Normal end
4	Abnormal end

D.10 Manipulating an Environment Variable

For CGI applications, manipulating an environment variable from a Web application causes no problem on the system because one environment variable block is allocated to each process and a CGI application runs as a process. For an ISAPI application, however, modifying the contents of an environment variable may influence IIS or other applications because an ISAPI application runs as multiple threads in the process of IIS itself. Additionally, an application may not run as expected because the contents of an environment variable, when referenced, is not guaranteed to be unchanging. Thus, you need to change a CGI application's section that manipulates an environment variable in order not to use an environment variable, when running under ISAPI.

You need to also be careful of a function that implicitly uses the current folder because the current folder used when an application is invoked is not guaranteed to be fixed. One example is manipulating a file by specifying a relative path from the current folder. Instead of attempting to use this method, you can do as follows:

Specifying a file using an absolute path

- Description in a CGI application

```
MOVE "RESPONSE.HTM" TO COBW3-HTML-FILENAME.
CALL "COBW3_PUT_HTML" USING COBW3.
```

- Modified description in an ISAPI application

```
MOVE "C:\COBOL\ISAPI\RESPONSE.HTM" TO COBW3-HTML-FILENAME.
CALL "COBW3_PUT_HTML" USING COBW3.
```

Specifying a file using a physical path corresponding to the virtual directory

To get a physical path corresponding to the virtual directory, use COBW3_GET_REQUEST_INFO.

- Description in a CGI application

```
MOVE "RESPONSE.HTM" TO COBW3-HTML-FILENAME.
CALL "COBW3_PUT_HTML" USING COBW3.
```

- Modified description in an ISAPI application

```
SET COBW3-PHYSICALPATH TO TRUE.
CALL "COBW3_GET_REQUEST_INFO" USING COBW3.
MOVE COBW3-REQUEST-INFO TO COBW3-HTML-FILENAME.
MOVE "\RESPONSE.HTM" TO
      COBW3-HTML-FILENAME(COBW3-REQUEST-INFO-LENGTH + 1:13).
CALL "COBW3_PUT_HTML" USING COBW3.
```

In relation to the above description, be careful of specifying a file in the run-time initialization file (COBOL85.CBR). For a CGI application, you can specify both absolute and relative paths. For an ISAPI application, you can specify only an absolute path. Since no alternative method is provided, modify the run-time initialization file so that it uses only absolute paths.

- COBOL85.CBR in a CGI application

```
[program-name]
access-name=SEQFILE.DAT
```

- Modified COBOL85.CBR in an ISAPI application

```
[program-name]
access-name=C:\COBOL\ISAPI\SEQFILE.DAT
```

D.11 Manipulating a CGI Environment Variable

A CGI application can reference a "CGI environment variable" using the COBOL environment variable operation function while an ISAPI application cannot. Thus, the following three subroutines are used.

COBW3_RECEIVE_HEADER

Get an HTTP header.

COBW3_GET_REQUEST_INFO

Get various information on a request.

COBW3_GET_AUTHORIZE

Get authorization information.

A CGI application can set and reference Cookie data using the COBOL environment variable operation function while an ISAPI application cannot. Thus, the following special subroutines are used to handle Cookie data.

COBW3_SET_COOKIE_XX, etc.

Register Cookie data.

COBW3_DEL_COOKIE_X, etc.

Delete the existing Cookie data.

COBW3_INIT_COOKIE

Initialize Cookie data to be sent to a client.

COBW3_GET_COOKIE_XX, etc.

Get Cookie data included in a request.

For example, modify the processing that gets information from an HTTP header as follows:

- Description in a CGI application

```
DISPLAY "HTTP_USER_AGENT" UPON environment-variable-name
ACCEPT browser-information FROM environment-variable-value
    ON EXCEPTION
        MOVE "Error" TO browser-information
END-ACCEPT.
```

- Modified description in an ISAPI application

```
MOVE "User-agent" TO COBW3-HEADER-NAME.
CALL "COBW3_RECEIVE_HEADER" USING COBW3.
IF COBW3-STATUS=ZERO THEN
    MOVE COBW3-HEADER-VALUE TO browser-information
ELSE
    MOVE "Error" TO browser-information
END-IF.
```

Modify the Cookie manipulation as follows:

- Description in a CGI application

```
DATA DIVISION.
WORKING-STORAGE SECTION.
  COPY COBW3.
01 COOKIEDATA.
  02          PIC X(19) VALUE "Sample+Cookie+data=".
  02 COOKIEVALUE PIC X(32).
PROCEDURE DIVISION.
  :
  DISPLAY "HTTP_COOKIE" UPON environment-variable-name
  ACCEPT COOKIEVALUE FROM environment-variable-value
  ON EXCEPTION
    MOVE "Error" TO COOKIEVALUE
  END-ACCEPT.
```

- Modified description in an ISAPI application

```
DATA DIVISION.
WORKING-STORAGE SECTION.
  COPY COBW3.
01 COOKIEVALUE PIC X(32).
PROCEDURE DIVISION.
  :
  MOVE "Sample Cookie data" TO COBW3-COOKIE-NAME.
  CALL "COBW3_GET_COOKIE" USING COBW3.
  IF COBW3-STATUS=ZERO THEN
    MOVE COBW3-COOKIE-VALUE TO COOKIEVALUE
  ELSE
    MOVE "Error" TO COOKIEVALUE
  END-IF.
```

To handle Cookie data through environment variable manipulation, the user needs to encode and decode URLs in the Cookie data. To work around this problem, use only alphanumeric characters in Cookie data.

Note that an en-size space, after URL encoding, becomes "+".