VisualAge Pacbase

Pacbase Web Connection Developer's Guide

Version 3.0

DSOWB000301A

Note

Before using this document, read the general information under "Notices" after the Table of Contents.

According to your license agreement, you may consult or download the complete up-to-date collection of the VisualAge Pacbase documentation from the VisualAge Pacbase Support Center at:
http://www.ibm.com/software/ad/vapacbase/productinfo.htm

Consult the Catalog section in the Documentation home page to make sure you have the most recent edition of this document.

1rst Edition (August 2002)

This edition applies to the following licensed program:

- VisualAge Pacbase Version 3.0

Comments on publications (including document reference number) should be sent electronically through the Support Center Web site at:
http://www.ibm.com/software/ad/vapacbase/support.htm
or to the following postal address:
IBM Paris Laboratory
Support VisualAge Pacbase
1 place J.B. Clément
93881 Noisy-le-Grand Cedex FRANCE

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

NOTICES

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM's valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

Intellectual Property and Licensing
International Business Machines Corporation
North Castle Drive, Armonk, New-York 10504-1785
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of information which has been exchanged, should contact:

IBM Paris Laboratory
Département SMC
1 place J.B. Clément
93881 Noisy-le-Grand Cedex
FRANCE

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

IBM may change this publication, the product described herein, or both.

TRADEMARKS

IBM is a trademark of International Business Machines Corporation, Inc.
AIX, AS/400, CICS, CICS/MVS, CICS/VSE, COBOL/2, DB2, IMS, MQSeries, OS/2, PACBASE, RACF, RS/6000, SQL/DS, TeamConnection, and VisualAge are trademarks of International Business Machines Corporation, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States and/or other countries.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.
UNIX is a registered trademark in the United States and/or other countries licensed exclusively through X/Open Company Limited.

All other company, product, and service names may be trademarks of their respective owners.

Foreword

Typographical conventions in use

The purple **courier New** font is used for any character string to be entered, displayed or corresponding to generated code and for the names of files, directories, windows, menu or components.

Italics is used for titles in cross-references.

The following icons are used:

☞ Note, remark, important point,

Cross-reference to another location in the documentation or to another documentation,

 Important, caution.

Terminological conventions in use

The terms **user** and **developer** are used in the *Developer's Guide* with the following meanings:

- The **user** is the person who uses the final Web application. He/she works on a personal computer on which a browser is installed.
- The **developer** is the person in charge of installing Pacbase Web Connection generating the revamping Parameters files, and installing the final application on end-user computers.
- The term **screen** refers to the screens in an application developped with the On-line Systems Development generator.
- The term **page** is used for HTML pages generated with Pacbase Web Generator.
- The term **screen page** is used for a page describing a screen.

General Presentation

Pacbase Web Connection

Pacbase Web Connection is designed to generate automatically Web applications from your existing VisualAge Pacbase applications at a low cost. It offers both an opening to Internet/Intranet and a greater use comfort (mouse-support, drop-down lists and interactive help).

With its many generation options, screens generated as HTML pages can be entirely customized both at the user interface level (addition of push buttons, etc.) and at the processing level (addition of field controls, etc.) by using Java language (Javascript, Applets, etc.) or by using any HTML editor.

Components

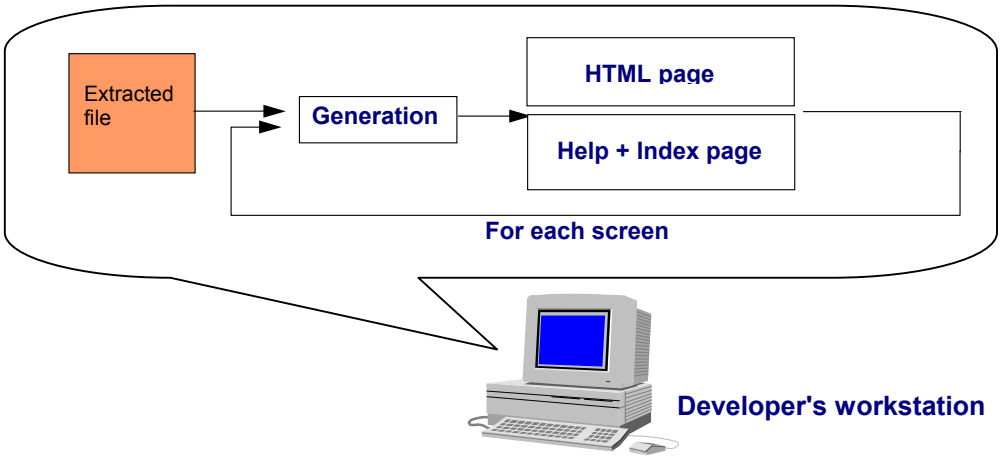
Pacbase Web Connection makes it possible to add a Web interface to traditional applications via the following components:

- The On-Line Systems Development generator by activating the option which enables the generation of a logical message.
- A sub-program which formats logical messages and transmits them to the communication layers, and inversely receives and decodes these messages.
- The communication layers.
- The context server which manages the user's context, merges the application messages with the HTML pages, and transmits the user's requests.
- The script to communicate with the context server: **`cgicgi`**.

Implementation Steps

The implementation consists in:

- Extracting, from VisualAge Pacbase Repository, the files containing the description of your screens and transferring them on the developer's workstation,
- Generating HTML pages, help and index pages,
- Parameterizing the context server and the HTTP server to make them communicate.



Operation of an Application via a Web Browser

When it is used via a Web browser, the application has the following operating mode:

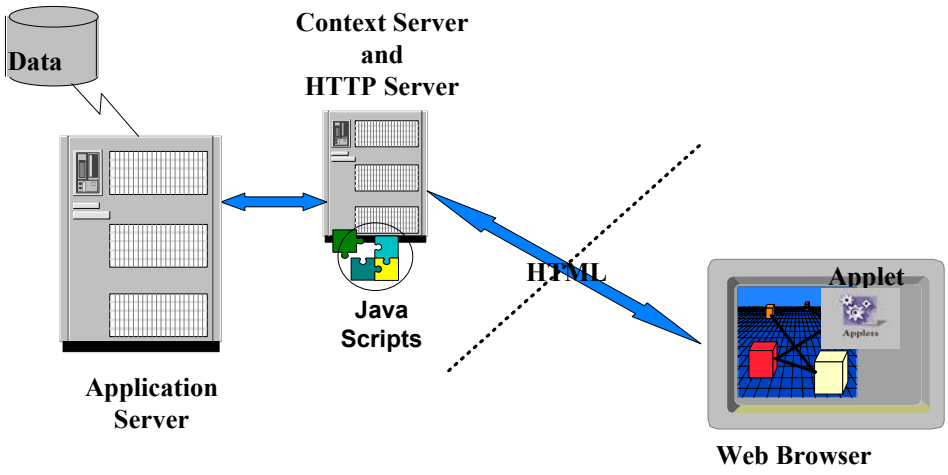
The user requests an access to the application by selecting a link in an HTML first page. This link activates the context server which:

- sets up a context for the management of the user's conversation and initializes a session if it is the first request, or retrieves an existing context if it is not the first request.
- transmits the request to the transactional application through the communication layers and transactional monitor.

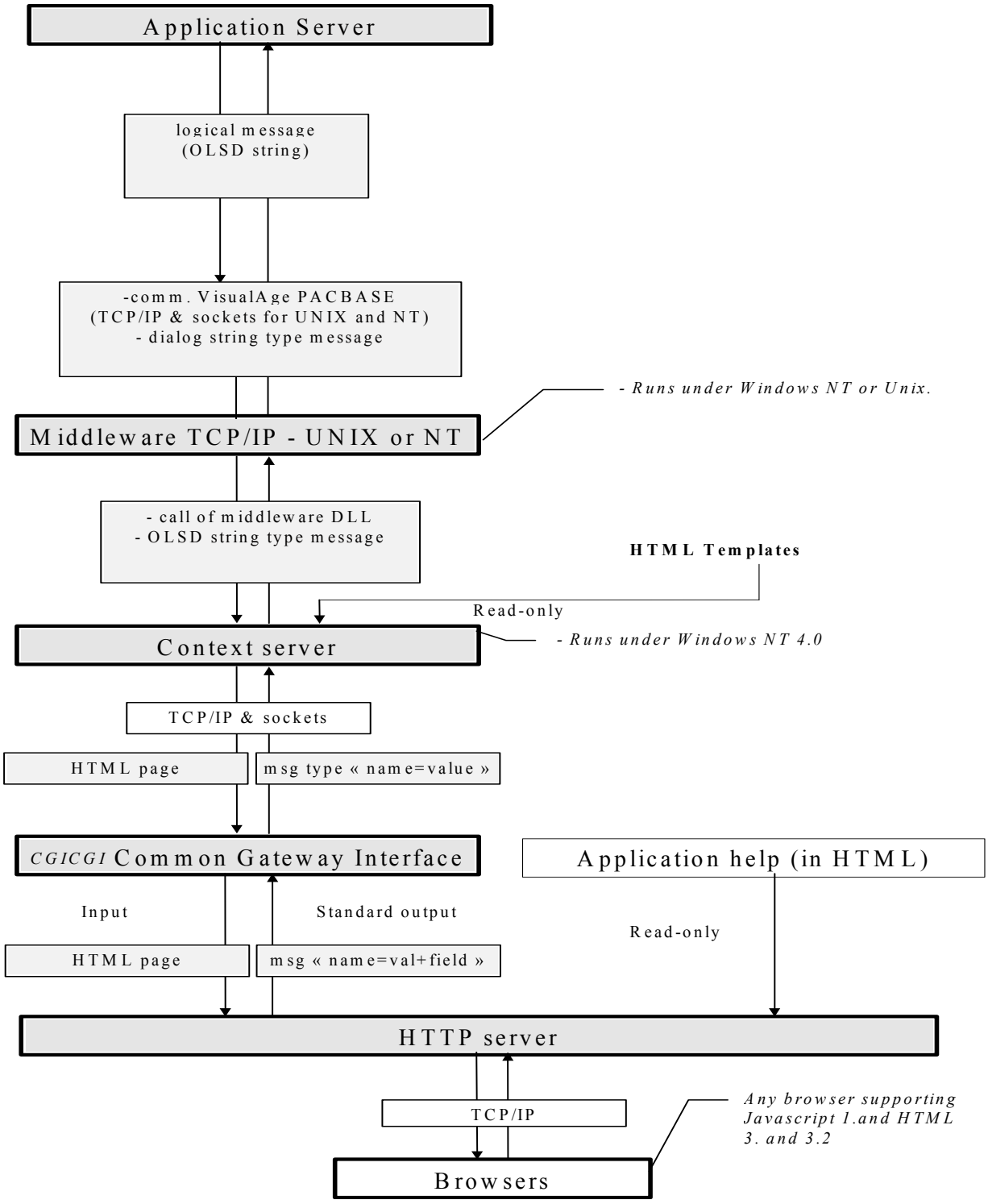
The application sends a logical message which corresponds to the requested screen.

The logical message is decoded by the context server and then merged with the HTLM description of the screen.

The HTLM page which results from the merge is transmitted to the user via the HTTP server.



Overall View of Communications Diagram



Servers

The context server converses with an HTTP server and with the application server. The context server manages the user context, merges the messages from the application with the HTML template pages. The result is sent to the HTTP server and transmitted to the user. In the other way, the user requests are edited by the context server to make them compatible with the application server.

Parameters of HTTP Server / Context Server

The HTTP server and the context server communicate via a socket. By default, both servers are supposed to be on the same machine and to use the socket number **2345**.

However you may install the servers on different machines and use another socket number.

In that case, you must modify the call of the CGI program which makes the link between the HTTP server and the context server. The name of the machine which supports the context server and the socket number must be added when CGI script is called in HTML pages. So you must modify the **cgicgicall.tpl** and **cgicgifirstcall.tpl** templates before generating the HTML pages of the On-Line Systems Development application.

If the machine on which the context server runs is symbolically named **srvpacweb** and if the socket number is **7801**, you must overwrite the following line:

- In the **cgicgicall.tpl** file :

```
#cgicgi%20+today.getTime()+'%20#monitor%20localhost'
```

by

```
'#cgicgi%20'+today.getTime()+'%20#monitor%20srvpacweb%207801'
```

- In the **cgicgifirstcall.tpl** file :

```
'#cgicgi%20' + today.getTime() + '%20#monitor%20' + MyUser
+ '%20' + MyPasswd + '%20localhost'
```

by

```
'#cgicgi%20' + today.getTime() + '%20#monitor%20' + MyUser
+ '%20' + MyPasswd + '%20srvpacweb%207801'
```

You may also directly use the IP address of the context server machine. In that case, you will get the following line:

```
'#cgicgi%20'+ today.getTime()+
'%20#monitor%20192.7.12.57%207801'
```

if *192.7.12.57* is the IP address of the machine.

You must also specify the socket number parameter at the context server activation context. For further details, refer to Paragraph *Socket number*.

Parameters of Context Server / Application Server

The communication protocol and the channel used betweeen the context server and the application server are identified by the values of the [Middleware] block in the **<Application name >.ini** file. These values are given by the developer in the Middleware tab of the generator window.

For more details, refer to Chapter *Revamping an Application*, Subchapter *Pacbase Web Generator Graphic Interface*, Section *Generation Options*, Paragraph *Middleware Options*.

Parameters of the HTTP Server

The HTTP server must be configured so that the **cgicgi.pl Perl** script can run, and the help pages and icons of the pages generated by Pacbase Web Generator can be loaded. To do this, you must modify the correspondence rules of the URL/ file requests in the configuration file (the file's extension depends on the type of server used).

The correspondence rules of an HTTP server managing an OLSD application are the following:

- Rule to run the Perl script

Exec /cgi/* <directory which contains the Perl interpreter and the cgicgi.pl script>

Example : **Exec /cgi/* c :\pacweb\Perl5***

- Rule to know the root directory of OLSD applications

The root directory is the directory where the first page of OLSD applications and the directories related to OLSD applications are located.

Example : Pass /pw/* c :\pacweb*

- Rules to load the JavaScript files, icons and help pages of an OLSD application.

For a generated application where **<ShortApplicationTitle>** (corresponding to the short title of the application specified in the **Short Application title** option of the generation window, **Application** tab) is **WE**, you will obtain:

```
Pass /WE /* c :\pacweb\WE \*
```

☞ Examples are given for information only, different input rules may apply depending on the server in use.

Use of the HTML page templates by the context server

The context server always uses HTML pages templates. The templates are merged with the application's data, and the template labels are dynamically replaced.

The context server should be capable to access the HTML page templates used to design the screens.

When the HTML page templates are generated on a computer different than the one allocated to the context server execution, the templates should be copied on the computer used to execute the context server.

These template files are to be found in the **<short_application_name>/app** directory issued from generation.

In the context server, you must specify the path to the directory which contains the template files. This information is given in the **Instal_Dir** line in the «**.ini** » file relating to the application.

The «**.ini**» file generated under Windows must be modified to take into account the AIX directory name in which the HTML pages are stored.

Examples:

If the directory choosen at generation is **c:\tmp** and the « short application name » is **WE**, you obtain the following lines in the «**.ini**» file:

```
Instal_Dir= c:\tmp\WE\App
```

If the directory choosen under AIX to install the context server is **/home/pacweb**, and if si you have copied the HTML templates under the **/home/pacweb/WEApp** directory, you will have to replace this line by the following one:

```
Instal_Dir= /home/pacweb/WEApp
```

You must add to the HTML pages templates the «**.ini**» file modified as above, which describes the parameters used for the communication system.

This «**.ini**» file must be transfered under the same directory as **pacweb.exe**.

Use of the middleware by the context server

General remarks

The context server should be capable to access the middleware to communicate with the application server.

The Windows **PATH** environment variable specifies the complete path to the directory containing the **.dll** file associated with the middleware type.

On the AIX system, the complete path to the directory containing the middleware (**ixo*.o**) must be specified in the **LIBPATH** environment variable. If the middleware and the context server are installed under the same directory, the middleware can be started with the following commands:

```
LIBPATH=. pacweb.exe
```

If the middleware mode selected at installation is **debug**, the middleware traces are activated when you set the environment variables as follows:

- ✓ **IXOTRACE=1**,
- ✓ **IXOTRACE_FILE** environment variables with the name of the file in which traces are recorded.

These environment variables are to be set in the **<Application short name>.ini** file resulting from the generation. This file includes the whole middleware parameters you specified in the **Middleware tab** of the notebook.

CPIC-XCP2 specific variables

The **DACSID** and **DACCFG** environment variables must be set so as to conform the instructions given in CPI-C/OSI Administrator's Guide.

VAPTCP variable used with a application server running on MVS or GCOS7

The **VAPTCP** environment variable must be initialized with the complete path to the file containing the transcoding tables when you use an application server running on MVS or GCOS7. However, by default, the context server uses the **819-297.nls** file of the current working directory as a transcoding table file.

Environment variables used by the context server

You can set the following environment variables in the command file (ex: **pacweb.bat** for Windows or **pacweb.cmd** for DOS) :

✓ **PWC_DELAY=3600** (default value)

This variable is used to specify the duration in seconds of a context. A context which were not modified during this period of time would be destroyed automatically and a message would be sent to the monitor to warn against deconnection.

The contexts validity test is performed every 10 minutes.

The following variables by **pacwebtr.exe** only:

✓ **PWC_DIR =C:\temp** (for Windows)

=/tmp (for AIX)

This variable indicates the directory under which the trace files are created.

✓ **PWC_MAXTRACE=4** (default value)

This variable specifies the maximum number of files used to save traces. Each file contains 5,000 lines. When the n files are full, the first file is reused so that the last traces are not saved.

✓ **PWC_TRACE =15** (default value)

This variable specifies the traces detail-level from 0 (no trace) up to 64.

Activation of the context server

The context server must fetch the **pacweb.ini** files of the applications it manages. These files must be located under the current directory of the context server.

To start the context server, you can:

- click on its icon directly, if it is located in the same directory as the **<name_appli>.ini** files,
- or create a shortcut by indicating the directory containing the **<name_appli>.ini** file as being a current directory,
- or start it directly from a DOS session.

Options for the context server activation

These options can be entered either directly when you type the context server command line from a DOS session or in the **pacweb.bat** file which means that the options are saved for further connections.

Socket number

By default, the socket number used by the context server and the HTTP server is **2345**.

If another number is used, the context server must be activated with this socket number parameter via the **-s** option.

Example from a DOS session: ***pacweb -s 6789***

Inhibit the dynamic change of presentation

If you want to inhibit the dynamic change of presentation attributs, you must set the **-a** option.

In this case, the context server does not modify the presentation of fields in a page. The fields are displayed as they had been defined in the page at generation time.

Monitor the context server execution

This option is used to monitor the context server execution, messages are displayed in the window in which the server was started.

To visualize these messages, you must set the **-d** option.

Backup of contexts in external files

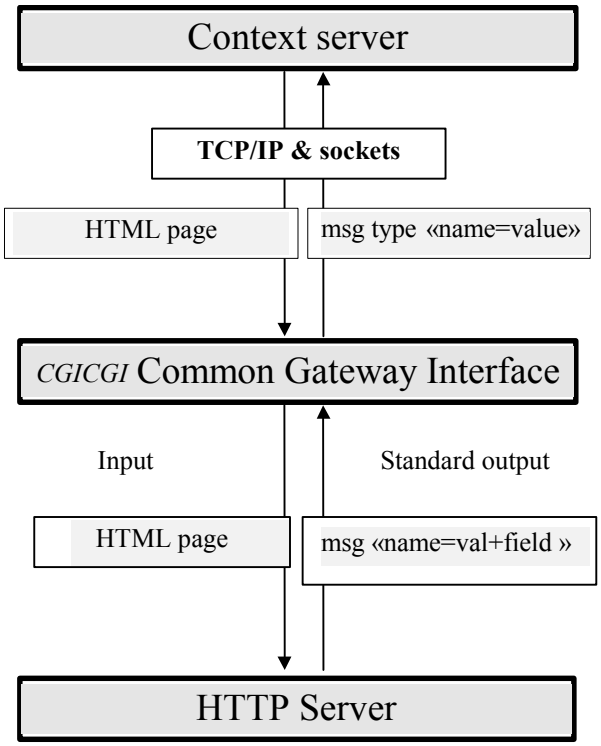
To enable the backup of contexts in external files, set the **-c** option.

This function is useful if you want to stop and restart a context server without disrupting already connected users.

Script cgicgi.pl

The **cgicgi.pl** script is used to transmit messages coming from the HTTP server to the context server.

The script is written in Perl language. It is translated from the HTTP server, for each HTML validation.

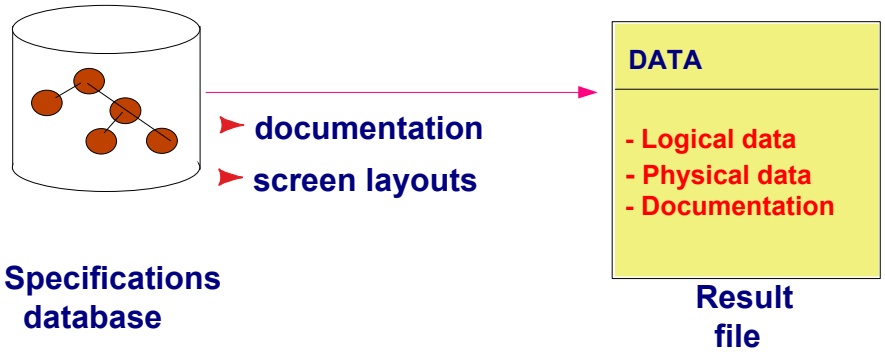

Context server
TCP/IP & sockets
HTML page
msg type «name=value»
CGICGI Common Gateway Interface
Input
Standard output
HTML page
msg «name=val+field »
HTTP Server

Extracting and transferring files

Extracting screens from the server

First you must extract the data required for local parameterizing from the referential.

- From the Dialog: application title, options, documentation
- From the Screen: definition, description, documentation
- From the Data Elements: name, code, length, autorized values, documentation
- Dialog help



The file which contains the descriptions of the screens to be revamped is output by the **GPRT** batch procedure, using the **GEO** command, and the **C4** option. The **GPRT** output is a user file: **PAC7GT** (or **GT** for GCOS8 systems) whose records are each 180-character long (maximum).

Transferring the extraction file to a computer

Downloading the **PAC7GT** or **GT** file on a computer is performed by the developer. The transfer utility must be parameterized so as to keep all special characters. The local resulting file of the transfer is what we will name here **FILENAME.EXT.**

> The modification of On-Line Systems Development programs for the implementation of Pacbase Web Connection is documented in the On-Line Systems Development Reference Manual.

Revamping an application

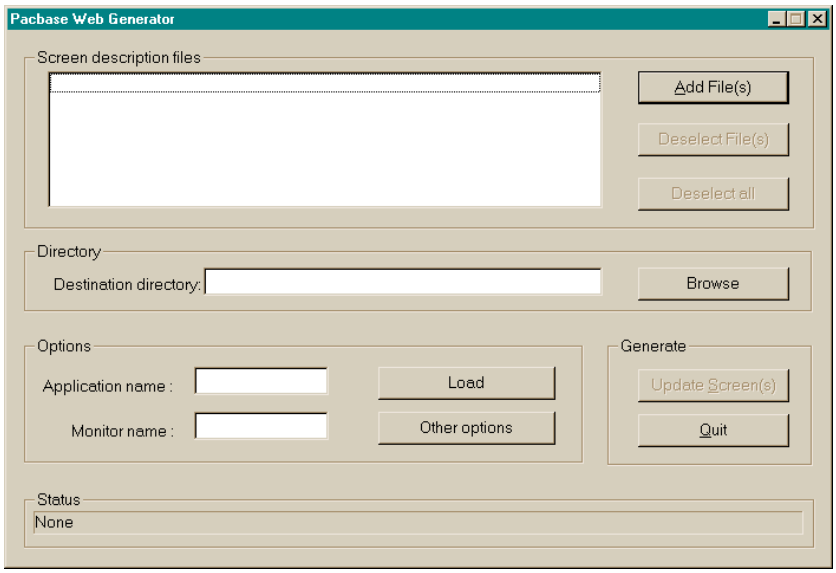
Revamping consists in defining options and parameters which will be used to revamp your VisualAge Pacbase application. To do that, you can use Pacbase Web Generator's graphic interface or a textual interface. The generator will change automatically the application's screens into HTML pages and the parameters selected are saved in files.

Pacbase Web Generator graphic interface

The Parameters files which enable the application revamping are automatically generated by Pacbase Web Generator, which handle data extracted from the VA Pac Database. You will be asked to enter application-specific data that will be automatically included in the generated Parameters files.

To start the Generator, you must execute **`Pacwebgen.exe`**, either directly from the file manager, or via a shortcut.

The following window opens.



Screen description files panel:

Select the extraction file(s) from which you will generate your HTML pages.

With the **`Add File(s)`** button, you can add one or more files in the list.

With the **`Deselect File(s)`** button, you can deselect one or more files.

With the **`Deselect all`** button, you can deselect all the files.

Directory panel:

Enter the directory in which the file structure tree will be generated. For example, if the destination directory is **c:\pacweb2** and the application short title is **WE**, you will have:

- **c:\pacweb2\WE_app:** directory which contains the HTML templates of the application.
- **c:\pacweb2\WE_help:** directory which contains the HTML pages of the application help.
- **c:\pacweb2\icon:** directory which contains the icons displayed in HTML templates.
- **c:\pacweb2\WE\js :** directory containing JavaScripts function files.

The **browse** button enables you to select the destination directory. If the directory does not exist, a message appears.

Options panel:

Application name field:

Enter the name of the file **<Application name>.ini** in which the parameters of generation will be saved as for instance, the parameters of the Middleware used for the application.

Monitor name field:

Specify the name of the monitor activated by the application server.

Load button:

Click on this button to reload the <Application name>.ini file specified in the **Application name** field; all the options of generation defined previously will be kept.

Other options

This button opens a notebook in which you can select generation options (see next section).

Generate panel:

Update Screen(s) button:

With this button, you can generate the extracted file.

Quit button:

With this button, you can quit Pacbase Web Generator.

The data displayed in the **Status** bar at the time of generation is stored in the **GEOC4report.txt** file.

If you generate your files directly, without any generation option, you will obtain a basic revamping.

Generation Options

A click on the **Other options** button of the Pacbase Web Generator window opens a dialog box which contains generation options that you can use to further customize your revamped application.

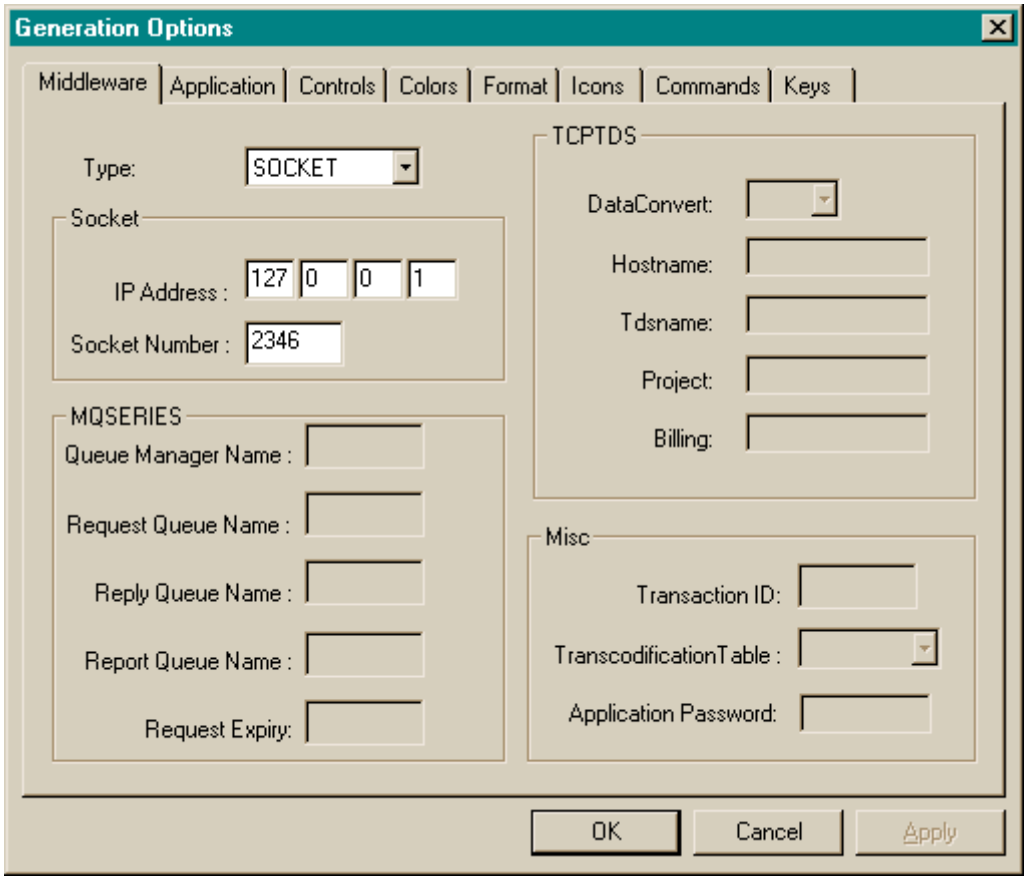
All of these generation options are saved in a file named **<application name>.ini.**

Overview of available options

- Suffix of the files which contain the HTML pages (htm or html)
- Type of Middleware and its parameters.
- General parameters: background color of the page (or background bitmap), color of links, visited links, and active links.
- Formats of labels associated with variable fields : use of the **.libvar** style or keep the initial attributes (color, intensity).
- Format of fixed labels: use of the **.libfix** style or keep the initial attributes (color, intensity).
- Error management: via the display of an 'Alert' box (JavaScript) or via an error message field, or both as it is advised.
- Controls to be generated (via JavaScript functions): numeric or date, presence.
- Generation of fields containing values: maximum number of values for radio buttons (fields with a higher number of possible values are generated as listboxes).
- Name of icons in menu bars. These icons are to be found in the Pacwebgn/icon directory.
- Labels associated with action and operation codes, associated display options.
- Location of function keys (on the bottom or the top of the page).
- Type of page branching: via a drop-down list, a radio button or an edit box.

Full details are given in the following sections.

Middleware options



Type of middleware:

Select the type of middleware in use for the communication between the server of contexts and the application server (the server of contexts is designed to manage several applications using different communication protocols or diifferent computers. The differents choices in the Type field are:

- **TCP/IP socket protocol**

If the context server and the application server communicate via sockets, you must indicate the **IP address** of the application server and **the Socket Number.**

- **CICS: CICS-ECI protocol**

- **CPIC : CICS-CPIC protocol**

If the type of communication is **CICS-ECI** or **CICS-CPIC,** you must specify the transcoding table used to transfer messages between the two systems with the true page code. This table is supplied as a file named **819-297.nls.** The complete access path to the file can be allocated to the VAPTCP environment variable. By default, the context server uses the **819-297.nls** file which is located under the current working directory.

Example: ***SET VAPTCP=c:\pacweb\819-297.nls***

Furthermore, with CICS-ECI you can enter via the Transaction ID field, the code of the CICS transaction which is activated for the communication.

- **CPICXCP2: CPICXCP2 protocol**

This protocol is used for connections between a context server running under AIX and an application server on GCOS8. You must specify the complete access path to the file containig the transcoding table on GCOS7.

- **TCPMVS: TCP/IP** towards **MVS protocol**

This protocol is used for communication between a context server running on AIX or Windows and an application server running under MVS and communicating via sockets. You must specify the complete access path to the file containing the transcoding table.

- **TCIS: TCP/IP protocol** towards **Unisys 2200**

This protocol is used between a context server running on AIX or Windows and an application server running on Unisys 2200 which both communicate via sockets.

- **MQSERIES: MQSERIES protocol**

If the communication is made through the use of **MQSERIES** protocol, you must specify the transcoding table in the **Transcoding Table** field as well as the parameters specific to this type of communication: name of the Queue Manager in the **Queue Manager Name** field, name of the Queues (report, request et reply) in the **Report, Request** and **Reply Queue Name** fields and Request Expiry in seconds.

- **TUXEDO/T** - **TUXEDO/WS**:

This protocol is used to enable a context server running on AIX or Windows to communicate with an application server running onWindows or AIX. When you select this protocol type, you must fill in the **Application password** field.

- **TCPTDS** :

This protocol is used to enable a context server running on AIX or Windows to communicate with an application server running on GCOS7. For this type of protocol, the **Hostname**, **Tdsname** parameters and the **Transaction ID** field must be filled in.

TCPTDS Panel:

If you have selected the TCPTDS middleware, you need to fill in the following options:

DataConvertfield:

If you want an automatic conversion of data , ASCII/EBCDIC, select **YES**. If you choose **NO**, enter the communication monitor name in the **Transaction ID** field.

Hostname field:

Enter the name of DPS 7000 host. IP addresses are rejected.

Tdsname field:

Enter the TDS name.

Project field:

Enter the project name. Once connected, the client application can start all transactions authorized for this project according to the PROJECT/TDS code specified in the GCOS7 catalog. If these parameters only contain spaces, the GCOS7 default project is taken into account.

Billing field:

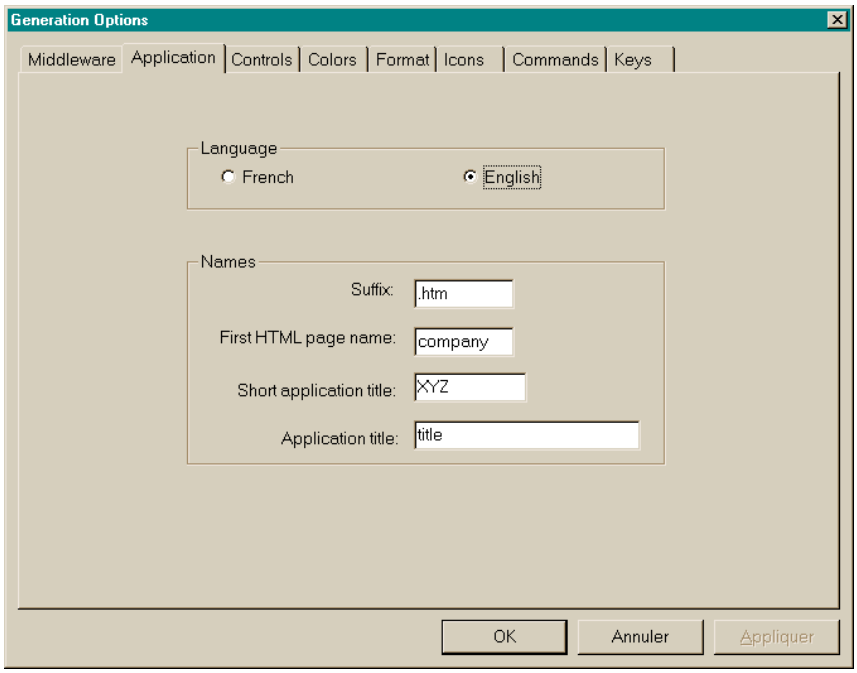
The billing is checked in the GCOS7 catalog. If these parameters only contain spaces, it is the default GCOS7 billing assigned to the project which is taken into account.

You will find in the following chart, the set of middlewares that may be used for connections between the context server and the application server:

Application server	Context server	
	AIX	WINDOWS
	TCPMVS	ECI CPIC TCPMVS
MVS/IMS		MQSERIES
UNIX*	SOCKET/TUXEDO	SOCKET/TUXEDO
WINDOWS*	SOCKET/TUXEDO	SOCKET/TUXEDO
GCOS7	TCPTDS	TCPTDS
GCOS8	CPICXCP2	
TANDEM	SOCKET	SOCKET
UNISYS 2200	TCIS	

UNIX*: represents the SunOS, AIX, HP-UX and DEC-OSF systems
WINDOWS*: represents the Windows system on Intel or Alpha platform.

Application Options



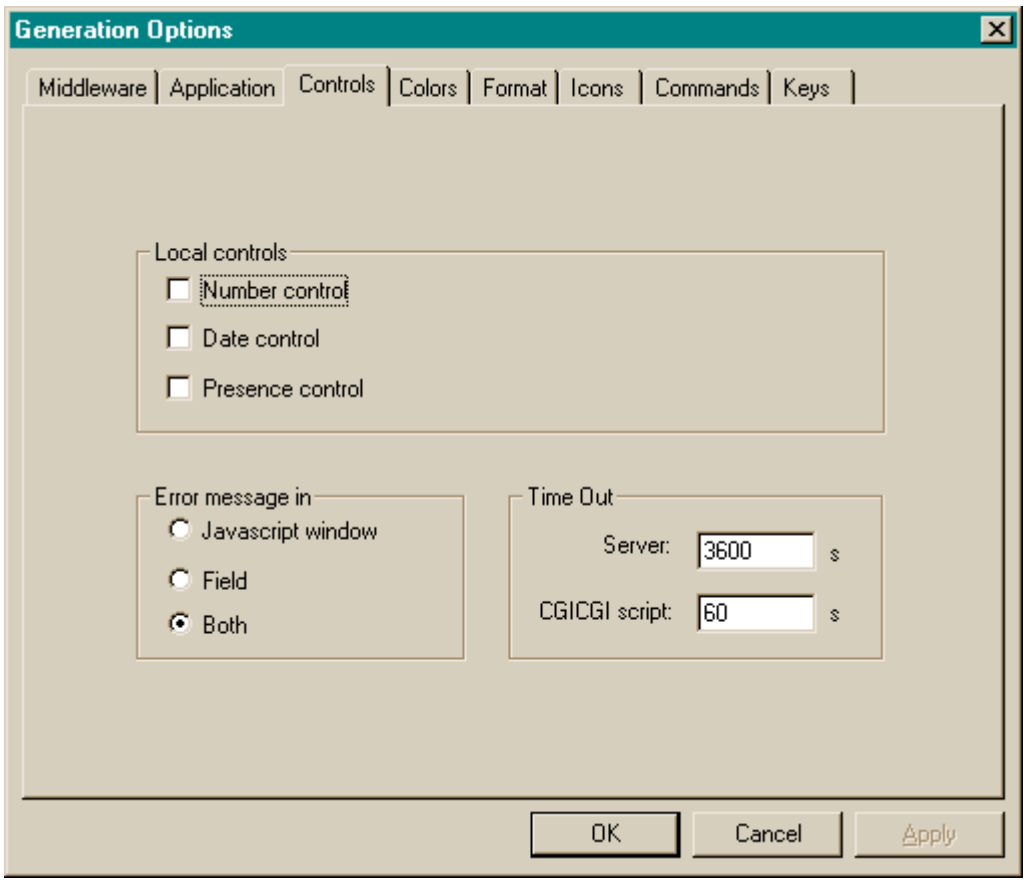
Language panel:

This option enables you to generate the first page and the standard messages of local controls in French or in English. The first page contains a title and the buttons which start the generated applications.

Names panel:

- **`Suffix`** field: suffix used in HTML file names (.htm ou .html)
- **`First HTML page name`** field: name of HTML first page
- **`Short application title`** field: character string which identifies the generated applications. It is used to create the directories which are specific to each application.
- **`Application title`** field: title which will appear in the start button of the application, in the first page.

Controls Options



Local controls panel:

With this option, you specify which local controls are to be performed: control on number, date, presence and cursor management.

In the screen description, you will find control options under the form:

- **ChkNum** for number control.
- **Chkhere** for presence control.
- **ChkDate** for date control.

These functions are described in the file **public.js** located in the js directory.

Error message panel:

With this option, you can parameterize the display of errors.

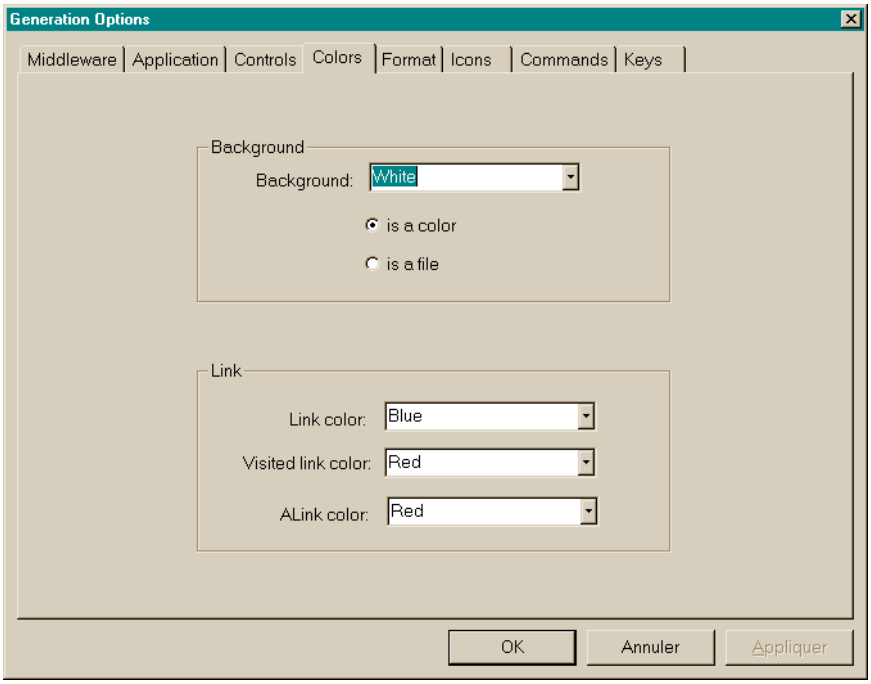
- **Javascript window:** the error message appears in a window directly in the browser.
- **Field:** The message appears at the bottom of the page as indicated in the layout.
- **Both:** Both displays simultaneously.

Time Out panel:

Server: indicates the response time of the server.

CGICGI script: indicates the response time of **CGICGI** program. This program enables the management of the connection between the HTTP server and the context server.

Colors Options



Colors:

With these options, you specify purely graphical parameters for the generation of pages.

Background panel:

Select the background color of the page.

- **Is a color:** select a color in the drop-down list.
- **Is a file:** select a bitmap in the directory:

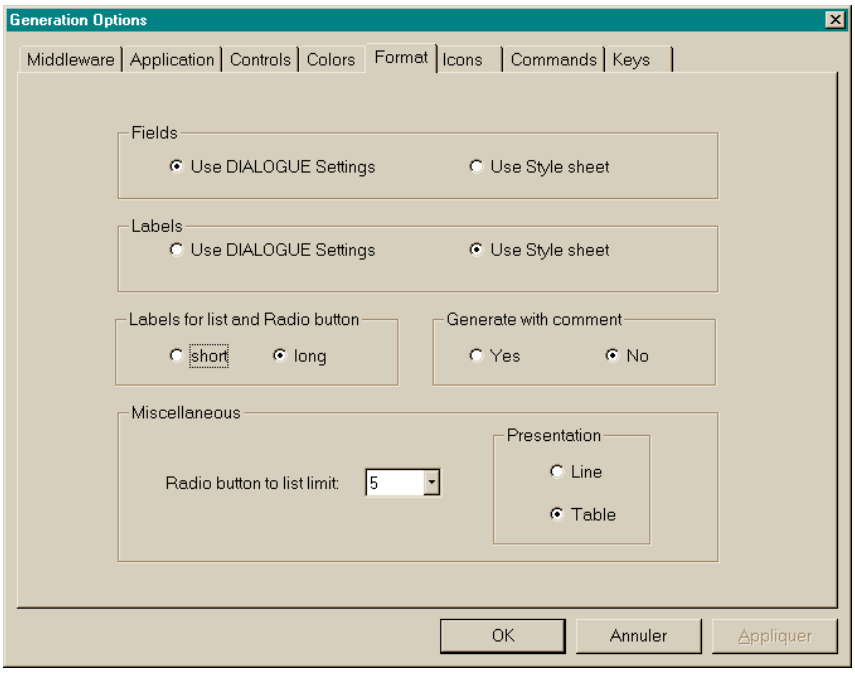
<ShortApplicationTitle>\icon, <ShortApplicationTitle> is the name entered in the **Short Application title** field of the **Application** tab.

Link panel:

Select the color of non-visited links, of visited links and of activacted links. These colors are used only for help pages.

Format Options

These options define the presentation of the displayed fields.



Fields panel:

Variable part of the screen.

- **`Use DIALOGUE settings:`**

Use of the display format as it is described in VisualAge Pacbase.

- **`Use Style sheet:`**

Use of the **`.libvar`** style which is described in the **`style.css`** file.

Labels panel:

Fixed part of the screen.

- **`Use DIALOGUE settings`**:

Use of the display format as it is defined in VisualAge Pacbase.

- **`Use Style sheet`**:

Use of the **`.libfix`** style which is defined in the **`style.css`** file.

Labels for list and Radio button panel:

- **`Short:`** Use the short label as value displayed by a drop-down list or a radio-button.
- **`Long:`** Use the long label as value displayed by a drop-down list or a radio-button.

Generate with comment panel:

Add comments in your HTML pages so that a postprocessor can modify the pages.

Miscellaneous panel:

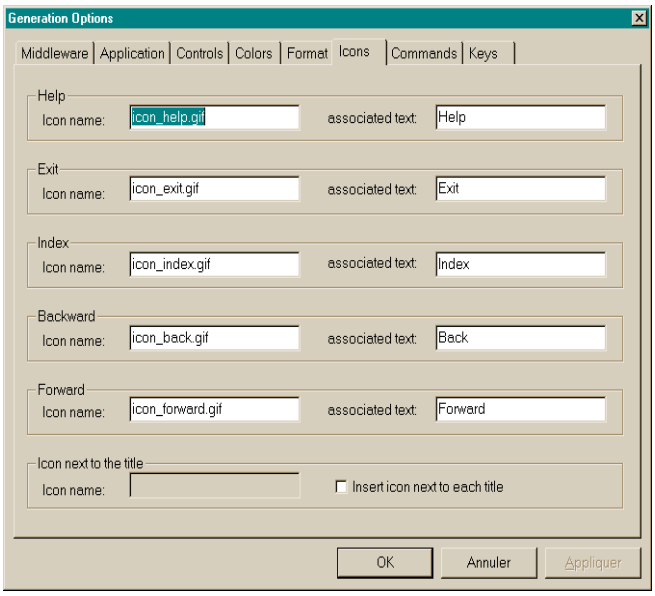
- **Radio button to list limit:** This value is the bound from which a choice list is displayed in a drop-down list rather than in radio-buttons.

Presentation panel:

Used to specify two modes of mapping:

- **Line:** this mapping mode takes into account a line to line layout.
- **Table:** this mapping mode takes into account vertical alignments defined in VisualAge Pacbase.

Icons Options



With these options:

- you name the files which contain the icons of the standard buttons. You can modify all displayed filenames.
- and you can enter the text of the default label associated with the icon in the **Associated name** field.

Help:

Enables the user to view help pages, the help on help and the help index.

For more details, see Chapter *Presentation of generated pages*, Subchapter *Structure of help pages*.

Exit:

Enables the user to quit the application or help pages.

Backward:

Enables the user to navigate in the help pages he/she has viewed already. By clicking on this button, he/she will view the previous page.

Forward:

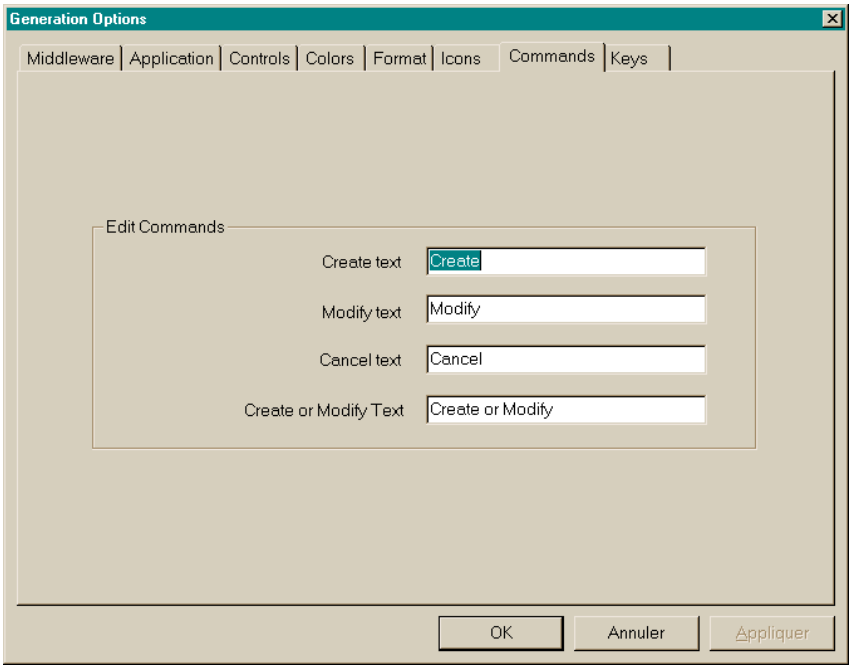
Enables the user to navigate in the help pages he/she has viewed already. By clicking on this button, he/she will view the next page.

Icon next to the title:

Insert icon next to each title: Enables you to insert a logo-type icon before the screen title.

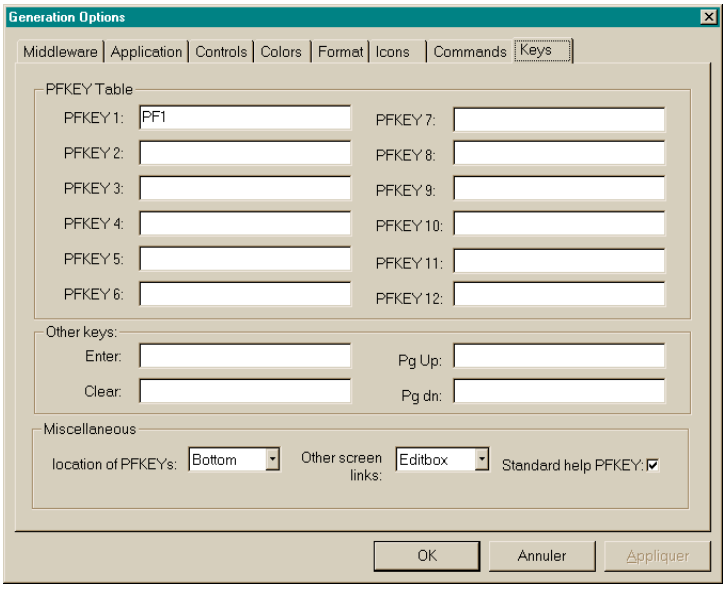
Icon name: Name of the file which contains the icon to be inserted at the top of each page if the box **Insert icon next to each title** is checked.

Commands Options



On this screen, you generate a list of choices corresponding to action codes: Create, Modify, Cancel, Create or Modify.

Keys Options



This screen enables you to manage function keys.

PFKEY Table panel:

You may enter labels which will appear in the buttons associated with function keys on the top or bottom of the screen. You may insert up to 12 keys.

Other keys panel:

- **`Enter:`** Label of the button associated with the Enter key.
- **`Clear:`** Label of the button associated with the Clear key.
- **`Pg Up:`** Label of the button associated with the Page Up key.
- **`Pgdn:`** Label of the button associated with the Page down key.

Miscellaneaous panel:

- **`Location of PFKEYs:`** Enter the location of function keys in the screen: on the top or bottom of the screen or in the footer.
- **`Other screen links:`** Specify the type of link mapping between screens (drop-down list, radio button, edit box).
- **`Standard help PFKEY:`** When this option box is checked, the function keys used to activate the standard help and the standard contextual help, are mapped with the button format.

The standard help is directly generated by the application server unlike the HTML pages help which is generated from the extracted file.

Textual interface

Activating the generator

To start the generator on a textual interface, you need to enter the following commands:

```
PWCbatch.exe [-d destination_directory] [-i parameters_file] geofile
```

Destination_directory refers to the directory where the whole HTML pages are generated.

Parameters_file refers to the file containing the different generation parameters. The default file used is **PacWebgen.ini**.

Géofile refers to the file resulting from the extraction of the VisualAge Pacbase files containing the screen description (GEO C4).

Modifying the generation parameters

The file **Pacwebgen.ini** is used by default to generate an HTML application. You must fill some of the fields in the file before starting the generation.

If you want to start the generation without a Parameters file, do not forget to enter the application name, the monitor name, the type of middleware and the different parameters relating to the selected middleware.

The generation parameters are:

Default value	Other value possible	Description
Suffix= **.htm**	**.html**	Suffix used to name HTML pages.
Error message= **BOTH**	**WINDOW or FIELD**	Parameterize the error messages display: displayed in a window or at the bottom of a page or both.
CGICGI timeout= **60**		HTTP server timeout: indicates the response time of the server
Monitor=		Monitor name
Instal_Dir=		Name of the directory containing the pages which are accessed by the context server
App_CodePage=		Server page code
MWARE=	**SOCKET/CICS/ CPICXCP2/TCPMVS/ MQSERIES/TUXEDO/ TCPTDS**	Type of middleware in use for the communication between the context server and the application server.
MWTIMEOUT= **3600**		Indicates the response time of the server
MWADDRESS=		IXO adress
MWCODEPAGE=	819	Client page code
MWQUEUEMANAGER=		

MWREQUESTQUEUE= MWREPLYQUEUE= MWREPORTQUEUE= MWREQUESTEXPIRY=		Name of the MQSeries queues (report, request and reply) and request expiry in seconds.
Language= **ENGLISH**	**FRENCH**	Generation language
First page name= **company**		First page name
Short application title= **APPN**		Character string used to create the different directories specific to each application.
Application title= **title**		Title displayed in the start button in the first page
Number control= **NO**	**YES**	Number control
Date control= **NO**	**YES**	Date control
Presence control= **NO**	**YES**	Presence control
Background= **White**		Background color of the page
Link color= **Blue**		Color of hypertext links
Visited link color= **Red**		Color of visited links
A Link color= **Red**		Color of activated links
Field format= **DIALOGUE**	**STYLE**	**DIALOGUE:** Use of the display format as it is described in VisualAge Pacbase **STYLE:** use of a style sheet
Label format= **STYLE**	**DIALOGUE**	**DIALOGUE**: use of display as it is described in VisualAge Pacbase **STYLE**: use of a style sheet
Comment= **NOCOMMENT** |	**COMMENT**	Generate with or without comments
Radio to list= **2**		
SLlabel= **short**	**long**	Label type: short or long as value displayed in a drop-down list or a radio button.
presentation= **table**	**Line**	Présentation type: a line to line layout (Line) or vertical alignment defined in VisualAge Pacbase.
		Name of the icon to be inserted on the left of the title
Help icon= **icon_help.gif**		Name of the icon associated with the button used to display the help
Help text= **Help**		Text associated with the button used to display the help
Exit icon= **icon_exit.gif**		Name of the icon associated with the button used to quit the application.
Exit text= **Exit**		Label associated with the button used to quit the application.
Index icon= **icon_index.gif**		Name of the icon used for the Index button.
Index text= **Index**		Text associated with the index button

Backward icon= **icon_back.gif**		Name of the icon used to access the previous page
Backward text= **Back**		Text associated with the button backward page
Forward icon= **icon_forward.gif**		
Forward text= **Forward**		Text associated to the forward page button.
C CMVT string= **Create**		Label of the create transaction code
M CMVT string= **Modify**		Label of the modify transaction code
A CMVT string= **Cancel**		Label of the cancel transaction code
X CMVT string= **Create or Modify**		Label of the create/Modify transaction code.
Function buttons= **TOP**	**BOTTOM ou FOOTER**	Location of the key functions: top of the screen, Bottom of the screen or in the footer.
Other screens= **EDITBOX**	**BUTTONS ou LISTBOX**	Used to specify the type of link mapping between screens: editbox, radio bouton or listbox .
Common Enter Key=		Label associated with the Enter key
Common Clear Key=		Label associated with the Clear key
Common PageUp Key=		Label associated with the PageUp key
Common PageDown Key=		Label associated with the PageDown key
Common PFKEY Table= **PF1,,,,,,,,,,,**		
Standard help PFKEY= **NO**	**YES**	Mapping of the standard help keys.

Generation Output

The generation of the interface outputs the following components:

- Under the **Destination Directory**:
  - ✓ an **<ApplicationName>.ini** file containing the application's parameters.

    These files are then installed on the workstation where the context server is running.

    <ApplicationName> corresponds to the name entered in the Application Name field in the generator.
  - ✓ a first page, required for initializing the Dialog whose name is given by the field **First HTML Page**.

    <FirstHTMLPage>.htm

- Under the **app** directory which is located in the **Destination Directory/Short application title**:
  - ✓ a utility page which contains JavaScript utility functions or the HTML screen descriptions in *frames* (=fields), in a file whose name is given by the field **Short application title**.
    <ShortApplicationTitle>app.htm
  - ✓ a page for each screen, stored in a file named **<screenCode>** (on 6 characters) and suffixed **htm** or **html** (generation option):
    <ScreenCode>.htm
  - ✓ the following files:
    editpl.htm, error.htm, fieldtpl.htm, hiddentpl.htm, menubar.htm, notfoundtpl.htm, passwdtpl.htm
- Under the **help** directory which is located in the **Destination Directory/Short application title**:
  - ✓ a help page named **<ShortApplicationTitle>help.htm**
  - ✓ a help page for each screen, contained in a file named **<ShortApplicationTitle>_help_<ScreenCode>** (on 6 characters) and suffixed **htm. <ShortApplicationTitle>** is the name entered in the **Short Application Title option** in the generation window, Application tab.
  - ✓ an index page for each screen, contained in a file named **<ShortApplicationTitle>_index_<ScreenCode>** (on 6 characters) and suffixed **htm.**
  - ✓ **footer.htm**: icons for help screens
  - ✓ **helphelp**: help on help
  - ✓ **<ShortApplicationTitle>_Index**: general index
- Under the **js** directory which is located in the **Destination Directory**:
  - ✓ a **public.js** file for public functions used at General level
  - ✓ a **private.js** file for private functions used at General level
  - ✓ **<ShortApplicationTitle>private.js**: private functions used at Dialogue level
  - ✓ **<ShortApplicationTitle>public.js:** public functions used at Dialogue level
  - ✓ **<ScreenCode>private.js**: private functions used at Screen level
  - ✓ **<ScreenCode>public.js**: public functions used at Screen level
  - ✓ **style.css**: file used for the style
  - ✓ **notfoundtplprivate.js** and **notfoundtplpublic.js**

For more information on the functions used at General, Dialogue and Screen levels, refer to Chapter *Customizing generated pages*, Subchapter *JavaScript functions*.

- Under the icon directory which is located in the **`Destination Directory/Short application title`**:
  - ✓ the bitmaps used in the different pages
- Under the generator's current directory:
  - ✓ A generation report in the **`GEOC4report.txt`** file.

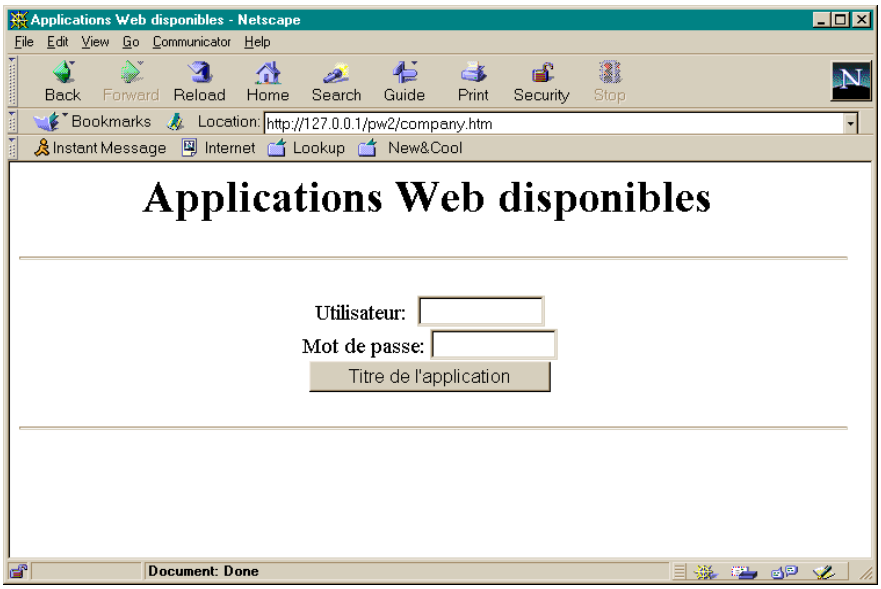
When you re-generate a page, you lose all initial modifications entered in your HTML pages.

Presentation of generated pages

First Page

The First page contains User and Password fields used in an application whose server requires a user identification (CICS for example).

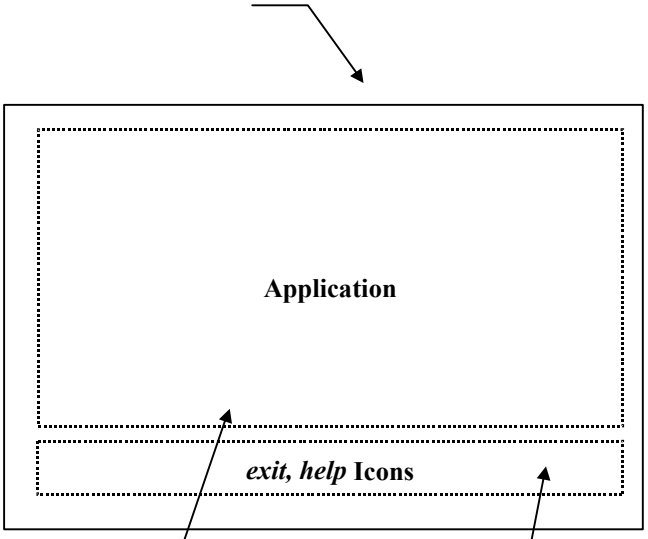
The label of the button used to start the application corresponds to the name you have entered in the **`Application Title`** field of the **`Application`** tab, in the generation options.



Structure of the general page

The HTML page is displayed in a general page which is made of two frames: the first one is used to display the application's screen and the other one to display the buttons which are controlling the application and the help windows.

File name: **<ShortApplicationTitle>App.htm**
Function: defines two frames app and menubar, loads basic JavaScript functions



Frame name: **app**
Filename contained in the frame:
<ScreenName>.htm
Function: displays the requested screen

Frame name: **menubar**
Filename displayed in the frame: menubar.htm
Function: displays the control bar of the application

Screen Description

The <ScreenName>.htm file which is generated contains the description of the screen. The screen is divided into two sections: the head and the body.

<HTML>
<HEAD>
Comments indicating that the screen results from a VisualAge Pacbase generation.
META data which identify the screen (program code, generatio date, etc.)
Screen title
Call of the stylesheet
JavaScript functions loading (at the screen level)
</HEAD>
<BODY>
<FORM METHOD=POST>
HIDDEN field used for the error message, the cursor positioning, the function key value
Screen description:
Literals
INPUT and SELECT fields
etc.
</FORM>
</BODY>
</HTML>

Head

This section contains the following data:

- identification of the program in use (META instructions),
- screen title,
- stylesheet name,
- loading of JavaScript functions.

Example:

```
<TITLE>screen title</TITLE>
<META NAME=“ SESSI” CONTENT=“ 5222 ”>
<META NAME=“ LIBRA ” CONTENT=“ xxx ”>
<META NAME=“ DATGN ” CONTENT=“ 15012000 ”>
<META NAME=“ TIMGN” CONTENT=“ xxxxxxx”>
<META NAME=“ PROGE ” CONTENT=“ xxxxxx ”>
<META NAME=“ COBASE ” CONTENT=“ xxxx ”>
<LINK REL=STYLESHEET HREF="/zz/app/style.css" type="text/css">
<SCRIPT LANGUAGE="JavaScript" SRC="/zz/js/zzxxxxPrivate.js">
</SCRIPT>
<SCRIPT LANGUAGE="JavaScript" SRC="/zz/js/zzxxxxPublic.js">
</SCRIPT>
```

Body

This section contains the screen description. It contains a FORM instruction which specify the **'POST'action** (parameter transfert option) and may include the list of controls to be performed (**onSubmit**).

The Body section has the following characteristics:

- Each new line creates a new line break (**
** instruction) in a **« Line »** type generation.

- Each label or protected field is retrieved as a text, adapted to the generation parameters.

- Variable fields (protected or not):
  - ✓ F-type data element (displayed or protected on the screen but received by the program): a hidden field whose value is the label text must be added besides the label which is in a text form.
  - ✓ P-type data elements (displayed or protected on the screen but not received by the program): the data element is displayed as a simple text.
  - ✓ V-type data element (variable): each variable field is described with an **<INPUT>** or **<SELECT> instruction,** followed by the name of the corresponding data element (**NAME attribute**), the field length (**MAXLENGTH** attribute) and for input fields, the displayed length (equal to the maximum length at generation time, but it can be modified by the developer). This instruction does not contain the call to the appropriate control function; this is done in the submit (onSubmit-type Event Handler).

Example:

```
<INPUT TYPE='TEXT' NAME='COPOS'
CONTENT='#COPOS000101' MAXLENGTH=5 LENGTH=5>
```

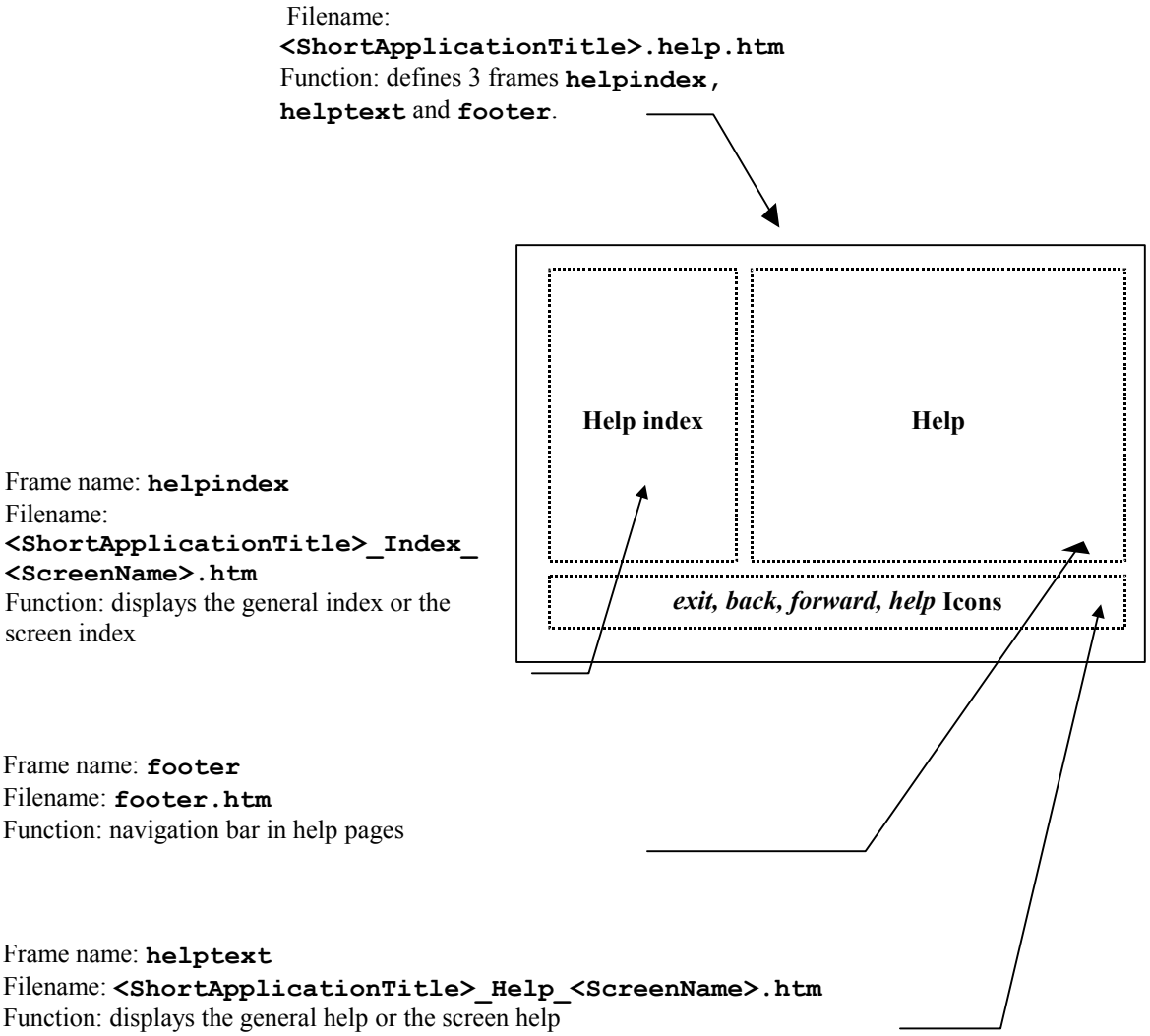
- The fields whose values are unknown (such as **TEXT** or **PASSWORD**) include a CONTENT (or VALUE) attribute whose value is the name of the field prefixed by # (see example above).

- Each field which has a default value is initialized with this value (**SELECTED** attribute of **BUTTON**-type **INPUT** fields, **SELECTED** instruction of **SELECT** fields, etc.).

Structure of help pages

Help pages are generated from VisualAge Pacbase extracted file.

Help is generated directly in HTML and saved on the HTTP server. Help is structured as follow:



Structure of help in frames

The file **<ShortApplicationTitle>_Help.htm** (<ShortApplicationTitle> was defined in the **Short Application Title** option) defines the size and location of the 3 frames. The 3 frames are described in 3 different files: (**footer.htm, <ShortApplicationTitle>_Index_<NomEcran>.htm** and **<ShortApplicationTitle>_Help_<ScreenName>.htm)**

Contextual Help

A help page is generated for each screen. This page contains:

- the general help associated with the screen
  <ShortApplicationTitle>_Help_<ScreenName>.htm
  It is displayed in the **helptext** frame.
- the list of the screen's data elements with their long label, authorized values and associated documentation
  ShortApplicationTitle>_Index_<ScreenName>.htm.
  It is displayed in the **helpindex** frame.

General Help (summary)

General help is designed to be displayed in the **helpindex** frame, i.e. the frame which can also contain the list of a screen's Data Elements.

General help contains the list of screens, each screen corresponding to a link on the help associated with this screen. The user then chooses the screen whose help he/she wants to view: the corresponding help page is displayed in the **helptext** frame and the screen index overwrites general help in the **helpindex** frame.

The Navigation Bar

The navigation bar is displayed in the **footer** frame, and described in the **footer.htm** file. It contains the following icons:

- Back and forward: to view help pages already viewed. These icons call the Javascript functions **history.back()** and **history.forward()** which are located in standard Javascript functions.
- Help : to display general help in the **helpindex** frame.
- Exit: to quit help.

Customizing generated pages

Generated HTLM pages can be entirely customized. You may modify the aspect of a page (add colors, icons, modify labels, layout or insert fields) by using an HTML page editor or by modifying the HTML page directly via a text editor.

However you must be careful not to delete the character strings which begin with #.

Example: Customizing the First Page

The first page is always generated. By default it contains a title field and a button to start the application. But this can be modified. For example, you may add buttons to start several applications from the same first page. To do this, you must copy the following HTML paragraph, located in the description of the generated first page, in the file whose name you have entered in the field **First HTML page name** of the **Application** tab.

```
 <INPUT TYPE = « button » NAME = « WE » VALUE =
« Application title »
                onClick = « openNewWin('/WE/app/','WE AppApp') »>
```

Styles

The CSS1 (Cascading stylesheet) architecture is used for the presentation (size, font, colors…) of different fields in a page. These styles are described in the **style.css** file under the **js** directory.

Styles can be modified, they are not overwritten once regenerated.

- There is one style for each type of field:
  - ✓ **.text** for a text type field,
  - ✓ **.passwd** for secret type field,
  - ✓ **.libfix** for fixed labels,
  - ✓ **.libvar** for variable labels,
  - ✓ **.radio** for radio buttons,
  - ✓ **.screentitle** style used for the title of a screen,
  - ✓ **.dropdown** style used for drop-down lists.

- 3 styles when mapping repetitive fields:
  - ✓ **.libtbl** for the label of the table,
  - ✓ **.rowodd** for odd rows,
  - ✓ **.roweven** for even rows.

- and n styles are available to simulate the display of the Dialogues.

  The names of these n styles are related to the names of the presentation attributes in VisualAge Pacbase. There are 56 different styles of **.libXYC** name, where X can take the N or B values (Normal, Bold), Y can take the N, B, U, R values (Normal, Bold, Underline, Reverse), and C can take W, R, P, Y, G, T B values (White, Red, Purple, Yellow, Green, Turquoise, Blue).

Automatic Modification of HTML Pages

Additional comment tags can be implemented at the generation (Generate with comment, Format options 0). These tags are added so as to enable automatic changes to be operated in the generated pages. They are used by the tool for a better reading of the generated page. In this way, it is possible to apply the same modification on a set of pages or to apply again the same changes after re-generation.

The generated tags are the following:

At the beginning of the screen line **<!--!PWC!LINEx-->** with x = Line number

Each field is framed either by :

<!--!PWC!field name-->

or :

<!--!PWC!xCy--> (for the labels)

Example: Automatic deletion of a line and modification of the SIZE attribut in some of the fields.

This example is written in Perl.

```
# print current line until arg1
sub printuntil
{
  unless (/$_[0]/)
  {
        print OUT;
        while (<IN>) {
              last if (/$_[0]/);
              print OUT;
        }
  }
}

# replace value of size if greater than 10
sub replacesize {
  &printuntil("SIZE=");
  s/SIZE=[0-9]{2,}/SIZE=15/;
}

# suppress a line
sub supline()
{
  local($wrk) = "<!--!PWC!LINE";
  local($search) = $wrk . $_[0] . "-->";
  if (/$search/)
  {
        while (<IN>) {
              last if (/$wrk/);
        }
  }
}

($filein) = @ARGV;
open (IN, $filein) || die "Cannot open file $filein
  :$filein \n";
open (OUT, ">result.htm");
while(<IN>) {
```

```
        &supline("19");
        if (/<!--!PWC!NOMRUE/)
        {
              &replacesize();
              &printuntil("<!--!PWC!NOMRUE");
        }
        print OUT;
};
```

JavaScript Functions

Presentation of Javascript functions

You can cutomize your HTML pages, add controls by example or modify the aspect of each page through the writing of Javascript programs. To manage HTML pages, the generated Javascript functions are divided into two categories: private and public functions.

Private functions are essential for the running of the whole application, they should not be modified.

Public functions are empty generated functions so that you user can enter your own processing. Their name should not be modified. Unlike public functions, private functions are systematically regenerated when generation is requested and all modifications are overwritten.

Furthermore, these functions are divided into 3 levels: General, Dialogue and Screen. The General level functions are common to all dialogues, the Dialogue functions are related to a particular Dialogue and the Screen level functions are related to a particular screen.

If we take **WX** as short application name as an example, the name of the files in which will be stored the functions at generation is:

	General	Dialogue	Screen
Public	**Public.js**	**WXPublic.js**	**xxxxxxPublic.js**
Private	**Private.js**	**WXPrivate.js**	**xxxxxxPrivate.js**

Functions used at General Level

Private

- **openNewWin(strPath, strAppName)**

This function opens a new navigator window. It is used to display the help.

- **Go(str)**

Function activated when a page is validated either by activating a key function or a « Enter » key

- **ReadCook()**

This function reads the name and the password the user entered in the FrontPage.

- **UpdatePosCursor (aObjectText)**

Store the field where the entry point is located.

- **SetPosCursor ()**

This function sets the entry point.

- **DisplayError ()**

Displays an error message in a JavaScript window if you asked this type of display at generation time.

- **PWCLoad()**

Function activated when a page is loaded.

- **PWCUnload()**

Function activated when a page is unloaded.

- **PWCChange(objet)**

Function activated when a field is modified.

- **PWCKeyPressed(e)**

This function is executed each time a keyboard is pressed. It is used to submit a page by pressing the Enter key.

- **PWCSubmit()**

Function activated when a page is submitted.

- **Traces functions**

The **TraceInit**, **Trace** and **TraceView** functions are used to add a trace mechanism. **TraceInit** initializes a variable in which additional messages are stored by use of the **TraceView** function. The display of the data contained in this variable and thus the display of traces, is triggered by the call of **TraceView**. These functions can be used in all public or private functions.

Public

- **chkDate(type,field)**

This function is used for date control of a field

- **chkNum(field)**

This function is used for numeric control of a field

- **chkHere(field)**

This function is used for presence control of a field

- **displayErr(str,field)**

This function displays the errors detected by the control functions above.

- **initArray()**, **isEmpty(str)**, **isANumber(str)**

Functions used to implement control functions.

Functions used at Dialogue Level

Private

It does not include any functions but a file is available to receive possible improvements.

Public

All these functions are empty. They can be used as entry points to enter additional processing. They are called through the use of General level functions.

- **`PWCLoadA()`**

Function called when a HTML page is loaded and **after** the specific processing defined at General level (see **`PWCLoad()`**).

- **`PWCUnloadB()`**

Function called when a HTML page is unloaded and **before** the specific processing defined at General level (see **`PWCUnload()`**).

- **`PWCChangeA(object)`**

Function called when a field is modified in a HTML page, **after** the specific processing defined at General level (see **`PWCChange(objet)`**).

- **`PWCChangeB(object)`**

Function called when a field is modified in a HTML page, **before** the specific processing defined at General level (see **`PWCChange`**).

- **`PWCKeyPressedB()`**

Function called when a keyboard is pressed, **before** the specific processing defined at General level (see **`PWCKeyPressed(e)`**).

- **`PWCSubmitB()`**

Function called when a HTML page is submitted, **before** the specific processing defined at General level (see **`PWCSubmit()`**).

Functions used at Screen Level

Private

- **`check()`**

This function is launched before submitting a page. It performs numeric, presence and date controls which could have been requested at generation.

Public

- **`PWCSCRLoadA()`**

Function called when the HTML page is loaded, **after** the specific processing defined at Général level (see **`PWCUnload()`**) and **after** **`PWCLoadA()`**.

- **`PWCSCRUnloadB()`**

Function called when a HTML page is unloaded and **before** the specific processing defined at General level (see **PWCUnload()**) and **before** the call of **PWCUnloadB**.

- **PWCSCRChangeA(object)**

Function called when a field is changed in a HTML page and **after** the specific processing defined at General level (see **PWCChange(objet)**) and after **PWCChangeA**.

- **PWCSCRChangeB(object)**

Function called when a field is changed in a HTML page and **before** the specific processing defined at General level (see **PWCChange(objet)**) and before **PWCChangeA**.

- **PWCSCRKeyPressedB()**

Function changed when a keyboard is pressed and **before** the specific processing defined at General level (see **PWCKeyPressed(e)**) and before **PWCKeyPressedB**.

- **PWCSCRSubmitB()**

Function called when a HTML page is submitted and **before** the specific processing defined at General level (See **PWCSubmitB()**) and before **PWCSubmitB**.

Using JavaScript Functions as entry points

Public JavaScript functions described above can be used to insert JavaScript code. This code is aimed at modifying the standard behavior of pages.

Depending the level at which they are used (General, Dialogue or Screen), these modifications will impact one or several pages.

Example: Additional function changing lower cases into upper cases characters

If you want to change systematically lower cases values in the NOVOL1000101 field into upper cases for a given screen, you just need to change the code of PWCSCRChangeA function defined in the **<Screen_Name>Public.js** file.

```
function PWCSCRChangeA(objet) {
  var wfrom = objet.name;

  if (wfrom == "NOVOL1000101")
        objet.value = objet.value.toUpperCase()
}
```

Templates

Presentation of templates

The HTML template files generated by Pacbase Web Generator are built from generation template files.

The HTML template files used for the generation of pages contain labels which are dynamically replaced with the content of .tpl files.

Frontpage (First Page)

HTML page which contains the application starting button.

Template used	Template label	Replaced with the content of the following file
Firsthtmlpage.tpl	#firsthtmlpagetitle	firsthtmlpagetitle_xx.tpl
	#errorcompat	errorcompat_xx.tpl
	#user	User_xx.tpl
	#password	Password_xx.tpl
	#suffix	The suffix given at generation

xx corresponds to the choice of language: **fr** for French and **us** for English.

Common Page

Page which contains the partition in frames and the local controls common to all screen pages.

Template used	Label	Replaced by
	#shortapp	Short apllication name given at generation
	#applicationtitle	Application name given at generation
	#cgicgifirstcall	Cgicgifirstcall.tpl
	#menubar	Menubar.tpl
	#suffix	Suffix given at generation
	#msgnoframes	Msgnoframes_xx.tpl

xx corresponds to the choice of language : **fr** for French and **us** for English.

Screen Page

Page which contains the description specific to a screen. There are two types of templates depending on the format choosen for the HTML page generation.

Template used	Label	Replaced by
Screen.tpl	#title	Short application name given at generation
	#screenname	VisualAge Pacbase screen name
	#body	Outfield.tpl Label.tpl Radio.tpl Editbox.tpl Hiddenfield.tpl Dropdownbox.tpl Table.tpl
	#enter	Submit button Submit_xx.tpl

Help Page

Page which contains the partition in frames of help pages.

Template used	Tag	Replaced with
help.tpl	#msgnoframes	Msgnoframes_xx.tpl

Page which contains the help text.

Template used	Tag	Replaced with
helpscreen.tpl	#label	helplabel_us
	#label	helplabel_fr
	#bodytag	bodytag.tpl
	#bg	backgroundfile.tpl

Templates Description

Templates	Description
app.tpl	Common page which contains the partition in frames and the functions used for local controls
backgroundfile.tpl	Used to generate the filename which contains the background in the case of a file-type background
begindex.tpl	Beginning of index pages
bodytag.tpl	Beginning of the BODY part of a page
bodytbl.tpl	Body of the generated page with option « Table »
	Title of a screen page
Ccontrol.tpl	Used to create a cell when using a Table format
button.tpl	Submit button
cell.tpl	Table cell
cgicgi.tpl	Activation of Perl script (called by `cgicgicall`)

cgicgicall.tpl	Call of Perl script with parameters
cgifirstcall.tpl	Call of Perl script for a first connection
chkdate.tpl	Date control function
chkhere.tpl	Presence control function
chknum.tpl	Numericity control function
drop-downbox.tpl	Drop-down list
editbox.tpl	Edit box
edittpl.tpl	Input field for a particular generation
endindex.tpl	End of index pages
errorcompat_fr.tpl	Error compatibility with the navigator in French
errorcompat_us.tpl	Error compatibility with the navigator in English
fieldtpl.tpl	Labelled field for a particular generation
firsthtmlpage.tpl	First page
firsthtmlpagetitle_fr.tpl	Title of first page in French
firsthtmlpagetitle_us.tpl	Title of first page in English
footer.tpl	help page footer
help.tpl	Partition of help pages in frames
helphelp_fr.tpl	French help on help
helphelp_us.tpl	English help on help
helpindex.tpl	Index page
helplabel_fr.tpl	French help title
helplabel_us.tpl	English help title
helplineindex.tpl	Link used by help indexes
helprub.tpl	Anchor for data element help
helpscreen.tpl	Page which contains all of a screen help
hiddenfield.tpl	Invisible field
hiddentpl.tpl	Template when generating particular hidden fields
indextitle_fr.tpl	French Title for index
	English Title for index
itemradio.tpl	Generation of a radio button item
Templates	**Description**
javacontrol.tpl	Check function
label.tpl	Label of a data element on a screen
lineindex.tpl	Link used by general index
menubar.tpl	Screen footer
Menubarpf.tpl	Screen footer with function keys.
msgdate_fr.tpl	Message for date control in French
msgdate_us.tpl	Message for date control in English
msgday_fr.tpl	Message for date control (day) in French
msgday_us.tpl	Message for date control (day) in English
msgfield_fr.tpl	Champ
msgfield_us.tpl	Field
msglongdate_fr.tpl	Long message for date controls in French
msglongdate_us.tpl	Long message for date controls in English
msgmonth_fr.tpl	Message for date controls (month) in French
msgmonth_us.tpl	Message for date controls (month) in English
msgnoframes_fr.tpl	Message to signal the use of frames in French
msgnoframes_us.tpl	Message to signal the use of frames in English
msgnumber_fr.tpl	Message for numericity controls in French
msgnumber_us.tpl	Message for numericity controls in English
msgvalue_fr.tpl	Message for a value request in French
msgvalue_us.tpl	Message for a value request in English
notfound.tpl	
Onclick.tpl	Contextual help for radio buttons
Onfocus.tpl	Contextual help for texts lists etc…
option.tpl	Choice for a drop-down list

outfield.tpl	Field
pacwebgenini.tpl	Generation of the pacwebgen.ini file
pacwebini.tpl	Generation of the pacweb.ini file
passwdtpl.tpl	Secret field when generating a particular page
password_fr.tpl	Password label for frontpage in French
password_us.tpl	Password label for frontpage in English
pfkey.tpl	Function key in menubarpf
private.tpl	Private JavaScript functions
privatedia.tpl	Private JavaScript functions at dialogue level
privatescr.tpl	Private JavaScript functions at screen level
public.tpl	Public JavaScript functions
	Public JavaScript functions at dialogue level
publicscr.tpl	Public JavaScript functions at screen level
radio.tpl	Radio button
row.tpl	Table row
screen.tpl	General skeleton for a screen
	button
screenbuttonbar.tpl	Button bar when branching on screens by pressing buttons
style.tpl	Stylesheet
submit_fr.tpl	French label of submit button
submit_us.tpl	English label of submit button
table.tpl	Table
titleicon.tpl	Addition of a title icon
user_fr.tpl	User label for first screen in French
user_us.tpl	User label for first screen in English

Modifying the Templates

Pacbase Web Generator generates HTML pages from HTML template files.

The HTML template files used in the generation processing contain HTML syntax blocks which can be modified. These blocks enable you to build an elementary paragraph.

It is possible to modify these templates so as to apply these modifications to all generated pages.

To modify a template, we advise you to duplicate the files located in the **`template.htm`** directory in the **`template.usr`** directory rather than modify them directly.

Such is the case for the **`cgicgicall.tmp`** template which describes the code related to the starting of the **`cgcgi.tpl`** script. You must modify this template if the context server and the HTTP server are not on the same machine. Replace the **`localhost`** string with the name (or IP address) of the machine on which the context server runs.