LAUNCHER Office

The **LAUNCHER Office** solution is dedicated to managing document flows from your systems, applications and RDBMS, developed by AURA Equipements.

AURA

Smart Software for Powerful Business

It allows you to define, design, and manage entire document flow processes.

LAUNCHER Office controls:

- Microsoft Office software (Word and Excel)
- sending mail (from Oulook, Lotus or SMTP)
- generation of files (PDF, for instance)
- any other Windows program

This is some features available :

- Launch a PC program, send parameters, wait for the result or the program termination,
- Transfer Database file, or SQL result set to the PC,
- Launch Word or Excel, open documents or templates,
- Show customized menus,
- Modify, edit, merge documents and data,
- Save modified documents, print, show,
- Convert documents into PDF,
- Compose and send e-mail messages, with attachments,
- Transfer database files or SQL result set to EXCEL or ACCESS.

These features are provided as:

- a set of Java functions for Windows and Linux
- REST webservices callable by any programming language for Windows and Linux
- CL commands for AS400

Here is a non exhaustive list:

- Create Word documents, in real time, (LNCPRTDOC),
- Create Excel workbooks (CPYTOXLS),
- Display documents in Word, PDF, Excel, HTML formats (LNCSHELL),
- Send documents by Fax or e-Mail (LNCSNDMAIL) (Sending mail),
- Transfer Data from queries on databases to PCs (DBFXFER),
- Replace your Printer-Output (Sending a spool),

See : Commands List included in LAUNCHER Office product.

Product package

LAUNCHER Office is made of:

- **The LAUNCHER Server** dedicated to LAUNCHER clients requests.
  This server includes input (eg AS/400 and/or DBMS) and output (eg Word and/or PDF) connectors, depending on the license you have acquired.

- **The LAUNCHER clients**, used by applications or programs on Windows, Linux or AS/400, that submit requests to the LAUNCHER server.

  Depending on the license you purchased, the LAUNCHER clients contain:

  - A set of Java classes for Windows clients,
  - A set of CL commands for AS/400,
  - A REST webservices gateway. Webservices should be called by any programming language.

Installation Files and Directories

The server part of LAUNCHER Office must be installed on each Windows machine to receive commands from programs on Windows, Linux or AS/400.

The installation directory is named by default:
"C:\Program Files\LAUNCHER<X>"

(eg C:\Program Files\LAUNCHER400 or
C:\Program Files (x86)\LAUNCHER XPRESS).

This directory has no dependency with any other file or directory on the system.

The « LAUNCHER » program must be launched on the PC, to start the demon process that will wait for the requests coming from Windows, Linux or AS/400.

This launch can be automated through the Startup group, or by using the Windows service.

Once the process is started, an icon appears on the taskbar (if you don’t use Windows service).

System Requirements

Windows PC/Server

- Operating Systems: all Windows OS supported by Microsoft.
- TCP / IP protocol.

Logiciels:

  - Microsoft Office version supported by Microsoft.
  - Optional: MAPI compliant email (Outlook, Exchange), Lotus Notes or SMTP server connection.
  - Optional : JDBC drivers and SAP JCO

If using AS/400 client:

- All models from the AS/400 series B.
- OS/400 since V5R2 version.
- TCP/IP connection between PC and AS/400.

If using webservices :

- JRE8 (Java SE Environment 8) installed on Windows PC/Server.

Installation

LAUNCHER Office installation

The installation of **LAUNCHER Office** consists of:

- Installation of the server part, the examples, and eventually the Java classes and the Webservices gateway on the Windows side.
- Eventually the installation of the client part on the AS/400 (LAUNCHER library).
  The client part must only be installed once on the AS/400.

Run the installation program "as an administrator" is recommended.

Prerequisites

On Windows PC:

TCP/IP protocol is part of the Windows installation.

No other software is required on the PC.

On AS/400:

The PC must be connected to the AS/400 through a TCP/IP link.

The FTP server is used only for the installation of the AS/400 side.
For the normal runtime, it can be stopped.

During the installation, the user profile of the AS/400 to be used must be **'QSECOFR'** type (it is possible to enter this during the installation).

Before installing the client software, be sure that you know the QSECOFR profile and the name or IP address of the AS/400.

Some security-management applications may prevent a correct installation of LAUNCHER Office, by preventing the importation of external data.

In such a case, desactivate the concerned application during the LAUNCHER Office installation.

Installation Steps

Run the installation program "as an administrator" is recommended.

❶ Fill in the dialog box and follow the instructions.

❷ **If you wish to install the AS/400 programs,** fill in the dialog box for this installation as well.

The following steps will be performed during the installation:

On the PC:

- Creation of a « Program Files \LAUNCHER<X> » directory (for example C:\Program Files\LAUNCHER400 ou C:\Program Files (x86)\LAUNCHER XPRESS)**,**
- Creation of a specific sub-directory for the examples,
- Copying all the required files.

On the AS/400:

- Creation of a **LAUNCHER** library,
- Creation of various objects in this library.

After confirmation, the installation software starts transferring programs to the AS/400.

In case of failure, you can run the transfer of AS/400 programs again using the "Installation AS400" module from "LAUNCHER Office", the program group which has been created.

The time required for the installation of AS/400 programs can vary considerably (from 5 to 30 minutes) depending on the type and the configuration of your AS/400.

Installation

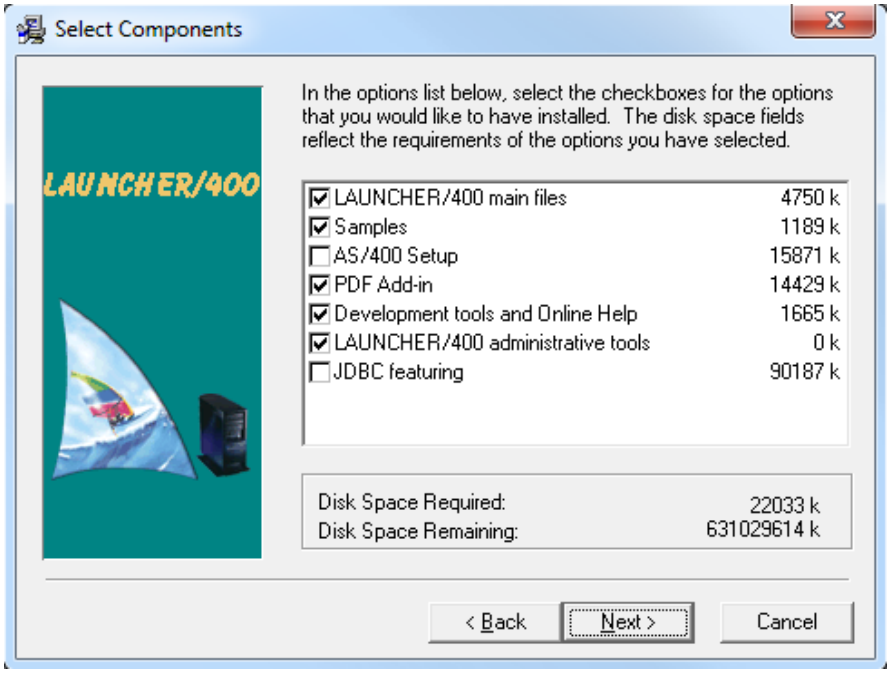
According to the license, Launcher Office comes in different products:

- **Launcher/400** : specially dedicated to AS400 environment
- **Launcher XPRESS** : deticated for all client application/program on Windows or Linux side, written in any programming language.

Launcher/400

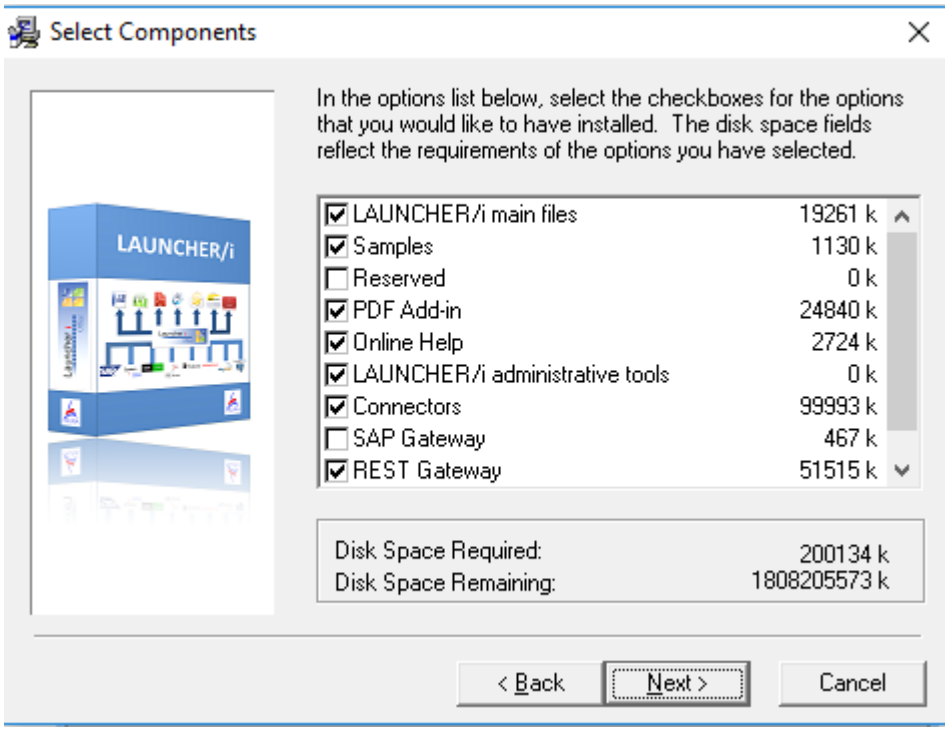
If you install the LAUNCHER400 product, by default you install only the server part on Windows side.

With the same installation executable you can choose to install the client part on AS400 (by checking "AS/400 Installation") .



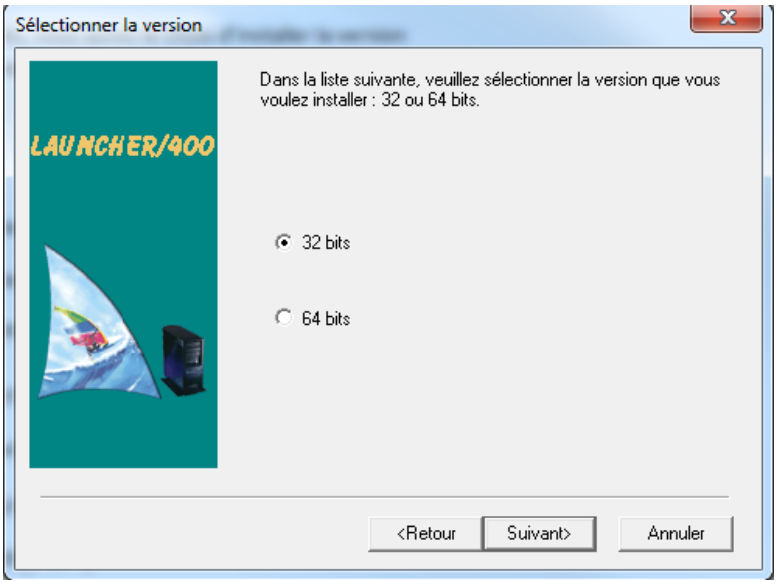
Launcher XPRESS

If you install the Launcher XPRESS product, by default you will install the Launcher server and the webservices Gateway server on Windows side.



32-bit or 64-bit version

During the installation on a 64-bit Windows PC/Server, you have the choice between the 32-bit or the 64-bit version of Launcher Office.



Choose the same version than the Microsoft Office one.
If you use the 32-bit Microsoft Office product, therefore install the 32-bit version of Launcher Office.

If you use the 64-bit Microsoft Office product, therefore install the 64-bit version of Launcher Office.

Otherwise, if you use the mail option with Lotus Notes, therefore it is mandatory to install the 32-bit version of Launcher Office.

Product license

Launcher 400

If you use LAUNCHER400, the licenses have to be installed on the AS400.

These licences will be sent by email from the commercial department.

The license is specific to each AS400 and is independent of Windows PC/Servers.

AS400 Activation Key

The AS400 activation key is necessary if you use AS400 client.

The activation key is calculated from the *AS/400's serial number* and *the processor group of the AS/400*.

To find out this number, on a terminal session, type the command :

DSPSYSVAL SYSVAL(QSRLNBR)

When you receive your activation code, on an AS/400 terminal, enter :

CHGCURLIB LAUNCHER

EASYREG <F4>

Enter the key parameters provided by Aura Equipements :

License	LNC or LNCE
Special Development License	*YES
Company name	Name_of_your_company
Activation Key	Your_key
Number of connections	0
Authorized partition ID (1->n)	0
EASYCOM'S Version	3
EASYCOM'S Options	0
Expiration date (ddmmyyyy) . . .	00000000
Authorized Proc. Group. . .	P05 or P10 or P20 or P30 or P40 or P50
Extended license	*NONE

Warning: Distinguish between O (letter) and 0 (digit) and between I (letter) and 1 (digit).

Launcher XPRESS

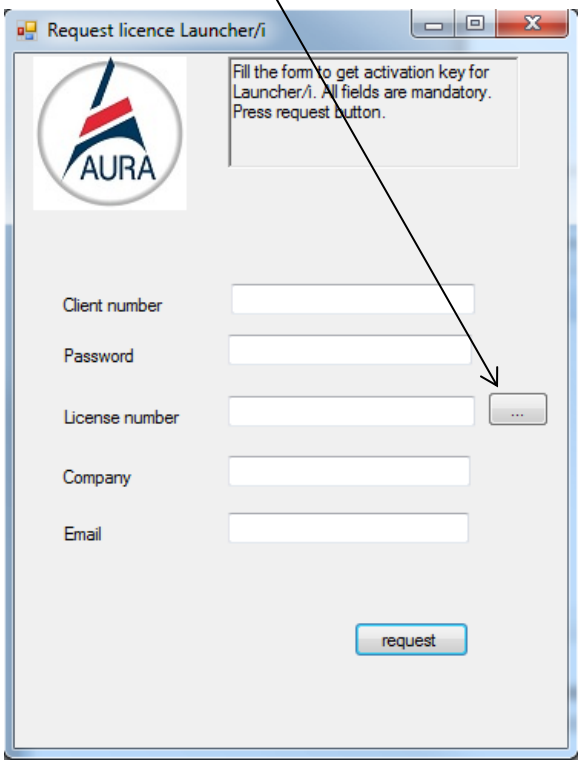
The license is specific to each Windows PC/Server on which the product is installed.

Please note:
1 serial number of the Launcher Office product = 1 single activation key for one single Launcher server

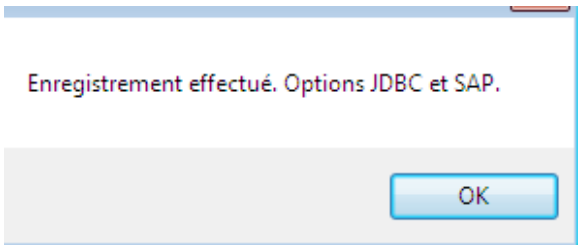
To use Launcher XPRESS, you must register an activation key on the PC hosting the Launcher server.

To obtain the activation key you must run the following program with administrator rights :
C:\Program Files (x86)\ LAUNCHER<X>\Licence**registerKey_EN.exe**

Use this button to know serial number of Launcher Office product you have. Firstly fill in your client number.



Fill all the fields and press the "request" button. A window will appear:



You will receive a summary email :

Your reference :
Client number : 1111111
Company : company

Mrs/Mr,

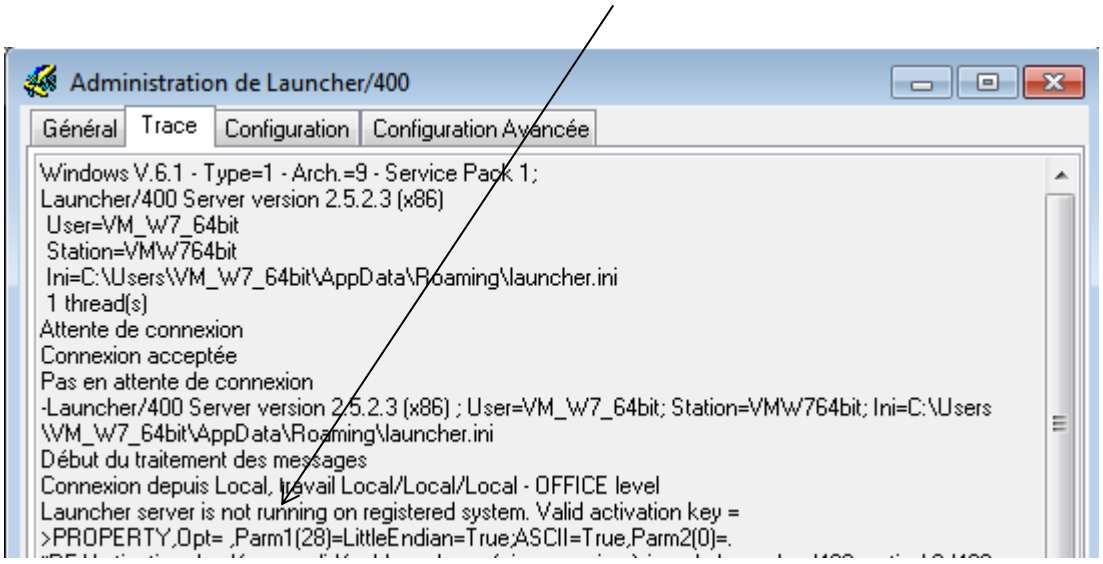
For the license Launcher/i N° AURA-KPNPGG58-NS, the activation key is : 5510975337514AFF03728968.

This key has been registered on your PC hosting the server Launcher/i.

Otherwise, you have subscribed to the options SAP and JDBC.

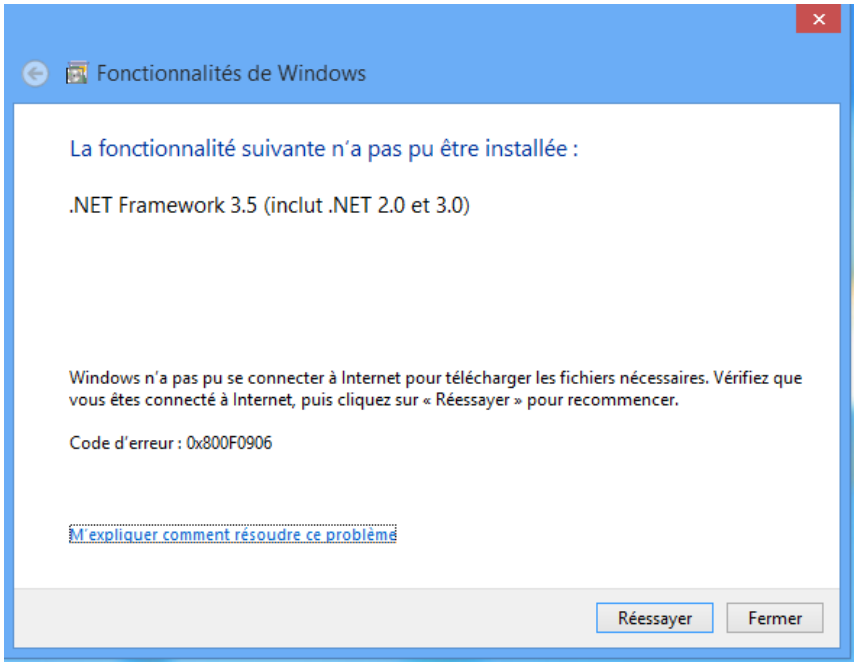
Best regards,
AURA Equipements

If you use the Windows Launcher Client with a Launcher server that does not have an activation key, this is the message you will get :



<u>Prerequisites for the execution of the program C:\Program Files (x86)\LAUNCHER<X>\Licence\registerKey_EN.exe :</u>

On some systems such as Windows 8, Framework 3.5 (included. NET2.0 and 3.0) may not be present, and you get this message when starting the program:



Proceed to download and installation using the following link :
http://www.microsoft.com/fr-fr/download/details.aspx?id=21

Batch installation in silent mode

It is possible to have an automatic installation program (LNC_xxxxAUTO.EXE).

Ask to the AURA technical service if you have a valid maintenance contract.

This program installs the Windows side of LAUNCHER Office, without any user action.

The command line options are:

/s: Set the "silent" mode. No message is shown.

/invisible: LAUNCHER Office will start in "Invisible" mode. Without any icon on the screen.

/nopdf: Do not install the PDF driver.

/withadmin: Install the administration tool.

/noautostart: Do not include LAUNCHER Office in the Auto Start.

/appdir="Installation directory": Set the directory path for LAUNCHER Office files.

/64 : Install the 64-bit version of Launcher.

/nostart : Launcher server does not start at the end of the installation.

If only the **/s** option is given, the installation is processed with the following options:

- Installation of the base components and LAUNCHER_PDF (are excluded: the samples, the administration tool…)
- The shortcut is added into the auto start.

Installation as "Windows Service"

Generality

LAUNCHER Office set up as "Windows Service" allows it to run on a dedicated server without any user sessions opened. A running machine is sufficient.

All LAUNCHER Office operations will then be executed in a hidden mode.
If an application tries to make a document visible through LAUNCHER Office in Service mode, the system generates an error.

LAUNCHER Office in service mode can be installed on any user PC or more generally on a Windows server.

Install/Uninstall the service

1. Set Up

All LAUNCHER Office operations must be closed before starting set up.

Launching **LNCsrv.exe** program with "/installservice" option on the command line will set up LAUNCHER Office as Windows service.

Use a Windows "cmd" session and enter the command:

"c:\program files\launcher<X>\lncsrv.exe" /installservice

A confirmation message must appear.

The service will be installed but not launched.
Configuration will take place in **automatic** launching mode.

Open the services manager to launch the service for the first time.
In few seconds the service will be ready to operate.

2. Uninstalling

To uninstall service, stop it manually, then enter the following command on command line:

"c:\program files\launcher<X>\lncsrv.exe" /uninstallservice

Service Mode Restrictions

The Service mode introduces some restrictions.

Service only

It is impossible to use LAUNCHER Office as a Windows Service and Normal mode simultaneously on the same machine (except on a TSE machine).

Trying to do so will result in the following error:

"This application fell to initialise correctly (0xc0000142). Click on OK to close application."

No Display

No display is allowed to the application driving LAUNCHER, any attempts will stuck the service. As example, the WORDSHOW function will send an error message.

It may be useful to allow interaction with the station desk (See : Service configuration from Windows services manager).

Machine Name

With 'classical' LAUNCHER Office the use of '*DEV' as ID for LNCOPEN connection may be common use.
In service mode, as LAUNCHER Office programs will not be run from an emulated terminal located in the machine using the service, the server should be directly named either with its name or its IP address.

'SYSTEM' User

Default system user will be the 'LocalSystem' or 'SYSTEM' user where service is running.
This may be a nuisance when using Lotus Notes messaging, MAPI or either to print on a network printer.
It may be necessary to change user in service configuration with Windows services manager.

Trace and tricky configuration

Service has no graphic interface, then using ‘LAUNCHER Administration’ becomes necessary to configure the service and/or refer to the trace.
Directly modifying the Launcher.ini file is another way to configure.

"LAUNCHER Administration" program allows LAUNCHER Office options to be set up in Service.
Modified .ini file name appears on the first program tab.

Trace is also displayed with this tool but you can also decide to use a **'file' trace**.

To consult trace or configuration, use the tab:
« Configuration on Network », then, select the machine or service line.

At the first tab bottom check if Service is concerned with the message :

"LAUNCHER Office Server version x.x.x.x – running as Service"

Now configuration and trace can be referred to.
Warning, some configuration option modifications reinitialise the connection with AS/400.

Stopping, Starting up or Restarting Service can only be performed with Windows services manager !

Service Mode – start up script

In view of using LAUNCHER Office as a Windows service, it may be useful to launch commands with LAUNCHER Office Service initialisation.

Modified launcher.ini file as follow in order to set on a start-up script :

[service]
StartupScript=c:\program files\launcher<X>\startup.cmd
(assuming a startup.cmd file in c:\program files\launcher<X> directory)

Any commands can be included in this file (other scripts launch, executable, network reading device, mapping, ..). Being line processed, this script must not contain script language (if, goto, …). Using such script types requires to call them from the main script.

If this option is activated a remainder appears in the LAUNCHER Office trace.

Example :

Processing the service Start up Script c:\program files\launcher400\startup.cmd

If one line fails, the line text will appear in the trace together with an error message.

If the script stacks, service will not start. It is very important that the script be tested separately before being put into place (example : using the LAUNCHER Office SHELL command).

N.B. : The script will only be run if Launcher is in Service Mode (Otherwise the option will be ignored).

Multiple instances installation

You can install multiple instances of the LAUNCHER Office server on a single Windows server.

Each instance must have:

- Its own service named, configured, and started.
- Its own independent directory with all LAUNCHER Office program and DLL files.
- Its own TCP/IP communication port. (LAUNCHER defaults port : 6078)
- His own file "launcher.ini" in his own directory.

You must install each of the services with the following command:

"<instance directory path>**lncsrv.exe**" **/installservice** ServiceName **/multiInstance /port** number

Example:

Assuming we have two LAUNCH_PROD and LAUNCH_DEV directories, each containing a full copy of the initial "c:\Program Files\Launcher400" directory.

"LAUNCHER_PROD\lncsrv.exe" /installservice Edit_Prod /MultiInstance /Port 6091
"LAUNCHER_DEV\lncsrv.exe" /installservice Edit_Dev /MultiInstance /Port 6092

The two commands above have installed 2 services "Edit_Prod" and "Edit_Dev" respectively on ports 6091 and 6092.

To address one of the instances, the client programs must specify the port number of the instance.
The port is given by the client program:

- With the address or the name of the server, separated by the ":" character :
ServerName: Number
Example: Edit: 6091

Examples:

Connection to LAUNCHER_PROD service:

LNCOPEN SVRADDR ('Editing: 6091')

Connection to Launcher service by default:

LNCOPEN SVRADDR ('Editing')

Dialog boxes with "Windows Service" mode

The different dialog boxes

In service mode (without interaction with the desktop), dialog boxes (for example Word or Excel) may appear during processing and block the LAUNCHER Office commands.

However, LAUNCHER Office triggers the closing of the dialog box automatically, which prevents blocking.

LAUNCHER Office regularly reviews the dialog boxes present in the session. For each "visible" dialog box (which would be visible if you were not in Windows service mode), indications appear in the LAUNCHER Office trace.

Then appears a text describing the action performed (or nothing if no action is performed).

LAUNCHER Office distinguishes two types of dialogs:

- **Dialog boxes** that are information boxes.
- **Message boxes** that are question or information boxes.

The dialogue boxes

In the case of a **dialog box**, LAUNCHER Office displays only the title of the dialog box and its class.

Example:

> *Dialog Info: **DialogTitle**=Save As, **DialogClass**=bosa_sdm_Microsoft Word 9.0, Style=94c80000, StyleEx=501, Id=0.*

A popup window with the title 'Save As' is present.

This can block LAUNCHER Office in 'service' mode.

LAUNCHER Office will automatically close this window, and this will probably generate an error while processing a current LAUNCHER Office command.

The LAUNCHER Office configuration allows you to change the default behavior (here closing), depending on the title or the class of the dialog box.

The message boxes

In the case of a **message box**, LAUNCHER Office displays the same information as for a "classic" dialog box, but with the **text** of the message as well as the text of the **buttons** normally displayed to the user.

Example:

> *Dialog Info: **DialogTitle**=Microsoft Word, **DialogClass**=#32770, Style= 94c801c5, StyleEx=10101, Id=0. **MsgBoxText**=Do you want to save changes made to Document1?*
>
> *Button n.1 ID 6: '& Yes'*
>
> *Button n.2 ID 7: '& No'*
>
> *Button n.3 ID 2: 'Cancel'*

A popup window with the title 'Microsoft Word' is present. This can block LAUNCHER Office in 'service' mode. LAUNCHER Office will automatically close this window, and this will probably generate an error while processing a current LAUNCHER Office command.

LAUNCHER Office has therefore automatically closed the message box which corresponds to the action "cancel", as if a user had pressed the "cross" of the message box.

The configuration of LAUNCHER Office allows to change the behavior and to choose a button to automatically activate or the default button.

Configuration for dialog boxes

For non-message **dialog boxes**, LAUNCHER Office can not retrieve the text that is present.

Possible actions are "keep the box", "**close the box**", or "validate the box".

"**Close box**" is the default behavior in LAUNCHER Office (except for "print in progress" boxes).

Only the **title** and **class** can give an indication on the dialog box.

Thus, two kinds of settings are possible:

- **selection by title**
- **selection by class name**

Note:

The following options also work also for message boxes.

Selection by title

To set up an automatic action based on the title of the dialog box, use the following options in the launcher.ini file:

[ServiceDialogs]
DialogTitle_<n> = <title text>
*DialogTitleAction_<n> = **Close**|Validate>*

<n> is an increment that starts at 1

<title text> is the exact title (case respected) of the title of the dialog box, corresponding to DialogTitle=xxx in the LAUNCHER Office trace.

DialogTitleAction_ <n> can have as value:

- "Keep" to leave the dialog box
- "Close" (default) to close it
- "Validate" to validate

Example:

[ServiceDialogs]
DialogTitle_1=Save As
DialogTitleAction_1=Validate

This will validate (not close) any dialog box titled "save as".

After modifying launcher.ini (and **restarting** the service), the trace will show for the same case:

Dialog Info: DialogTitle=Save As, DialogClass=bosa_sdm_Microsoft Word 9.0, Style=94c80000, StyleEx=501, Id=0.

The default button in the 'Save As' dialog box has been automatically selected.

Selection by the class

To set up an automatic action based on the class of the dialog box, use the following options:

[ServiceDialogs]
DialogClass_<n> = <class of the box>
*DialogClassAction_<n> = <Keep|**Close**|Validate>*

<n> is an increment that starts at 1 (DialogTitle_ <n> separate numbering).

<class of the box> is the exact title of the class of the dialog box.

DialogClassAction_ <n> can have the value "Keep" to leave the dialog box, "Close" (default) to close it or "Validate" to validate.

Note:

The DialogClass_ <n> and DialogTitle_ <n> numberings are independent (each starting at 1).

Configuration for the « Message » boxes.

To change the default behavior of LAUNCHER Office against message boxes, the possible options in Launcher.ini are:

[ServiceDialogs]
MsgBoxText_<n>=<start_of_text>
MsgBoxActionNum_<n>=<button_number>

<n> is a number that starts at 1

<start_of_text> is the beginning of the text in the dialog box (see the value of "MsgBoxText =" in the trace).

<button_number> is the number of the button to press (starts at 1) or 0 (at the default) so that the default button is selected.

Example:

Dialog Info: ***DialogTitle****=Microsoft Word,* ***DialogClass****=# 32770, Style= 94c801c5, StyleEx=10101, Id=0.* ***MsgBoxText****=Do you want to save changes made to Document1?*

*Button n.****1*** *ID 6: '& Yes'*

*Button n.****2*** *ID 7: '& No'*

*Button n.****3*** *ID 2: 'Cancel'*

To handle this case, we can insert:

[ServiceDialogs]
MsgBoxText_1 = Do you want to save the changes
MsgBoxActionNum_1 = 2

After modifying launcher.ini (and **restarting** the service), the trace will show for the same case:

Dialog info: DialogTitle = Microsoft Word, DialogClass = # 32770, Style = 94c801c5, StyleEx = 10101, Id =
MsgBoxText = Do you want to save changes made to Document1?
Button n.1 ID 6: '& Yes'
Button n.2 ID 7: '& No'
Button n.3 ID 2: 'Cancel'

The button number #2 (labeled: '& No') into the message box the 'Do you want to save changes to Document1?' was automatically selected.

To make multiple rules, simply increment <n>. For example:

[ServiceDialogs]
MsgBoxText_1 = Do you want to save the changes
MsgBoxActionNum_1 = 2
MsgBoxText_2 = Text Second Box
MsgBoxActionNum_2 = 0

Combining DialogClass_<n>, DialogTitle_<n> and MsgBoxText_<n>

A class name is often the same for several boxes, therefore we may want to define a default behavior for a given class and refine (handle "exceptions") by the title.

To handle this possibility LAUNCHER Office manages the options with the following order of priority:

1) MsgBoxText_<n>: the highest priority

2) DialogTitle_<n>

3) DialogClass_<n>: the lowest priority

For example, if you put in the launcher.ini file:

[ServiceDialogs]
DialogClass_1=bosa_sdm_Microsoft Word 9.0
DialogClassAction_=Keep

DialogTitle_1=Save As
DialogTitleAction_1=Validate

MsgBoxText_1=Do you want to save the changes
MsgBoxActionNum_1=2

This means that if a dialog box has the class "bosa_sdm_Microsoft Word 9.0", it will be kept unless the title is "Save As". In which case it is validated.

Unless this is a message box whose text begins with "Do you want to save the changes" in which case the 2 button is selected.

General options about dialog boxes configuration

The additional options available are:

[ServiceDialogs]
FilterMode=<0|**1**|2>

The "FilterMode" option allows you to enable or disable the automatic management of message boxes. The possible options are:

0: The dialog box filtering system is totaly disabled.
All the dialog boxes will be kept, all the other entries in [ServiceDialogs] section will be ignored.

1 (default) : Normal management.
All the entries in [ServiceDialogs] section are managed.

Several internal rules are handled.
For example, the message "Printing …" is never closed, a "Field not found" during a mail merge is closed.

2: All the dialog boxes are closed by default.
All the others entries in [ServiceDialogs] section are still managed.

Using LAUNCHER Office

Starting LAUNCHER server on a Windows machine

To establish a communication with a PC hosting LAUNCHER server, from a client application, the LAUNCHER server must be running on the PC.

Choose **« Add a shortcut to the auto start group »** during the installation process to make LAUNCHER start when a Windows session is started.

Or, launch LAUNCHER Office on the PC from your "Start" menu.
It is the "normal" mode.

On a Windows sever, you can launch LAUNCHER Office in "service mode".

By default, within "normal" mode, Launcher server launches also the administration screen.

An icon appears on the Windows task bar.
When you double click on the icon, the administration screen is shown.

The LAUNCHER Office server is running as the process "LNCSrv.exe".
The administration tool is running as "LNCAdm.Exe".

Use the AS/400 client

On the **AS/400**, the **client** side is made of a set of programs, commands and files, in the single library named "**LAUNCHER**" by default.

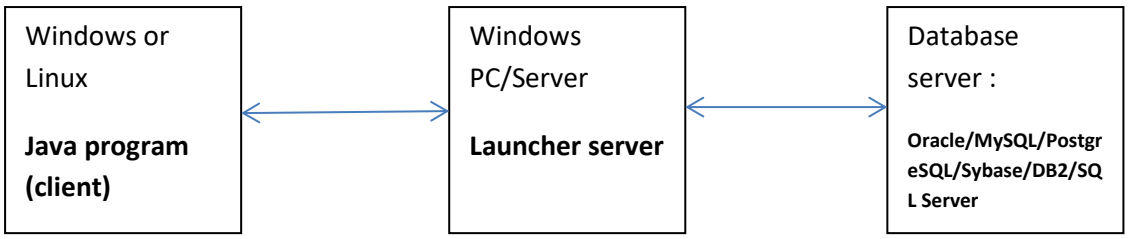
This library must be on line for the jobs using LAUNCHER Office :

ADDLIBLE LAUNCHER

This library can be copied or renamed.

There is no special configuration to do on the AS/400, except for the TSE or CITRIX environment.

Use the Java client



The Java client used into the client application allows interaction with the Launcher server.

It allows to use :

- the traditional commands of LAUNCHER Office with LNCCMD (see command list)
- or the provided Java classes such as **LNCTOXLS, LNCPRTDOC** and **Datasource**.

The different options for LNCTOXLS and LNCPRTDOC are the same as those usually available in an OS400/System i client environment.

The data source for commands LNCTOXLS and LNCPRTDOC :

- can be a CSV file
- or can come from database : Oracle, MySQL, PostgreSQL, Sybase, DB2 for AS400 and Microsoft SQL Server.
- or can come from SAP sytem

Configuring the Java application on client side:

The properties of the Java project, created through an IDE (Integrated Development Environment) such as Eclipse or Netbeans ... must be changed in order to the following library appears in the Java Build Path:

- lnciclient.jar

You have to import the following JAR :

C:\Program Files (x86)\LAUNCHER XPRESS\LAUNCHER_CLIENT\lnciclient.jar

Installation of the connector SAP JCO on the PC/Server hosting the Launcher server:

In order to access the SAP system, it is imperative that the sapjco.jar JAR (SAP JCo 2.1) and the sapjco3.jar JAR (SAP JCo 3.0) are present in the installation directory of the Launcher server.
For instance:

C:\Program Files (x86)\LAUNCHERi\LAUNCHER_CONNECTORS\Inciconnector_lib

Moreover, DLLs associated to the connector SAP JCO 2.1 (sapjcorfc.dll and librfc32.dll) or SAP JCo 3.0 (sapjco3.dll) have to be present into the following directory:

C:\Program Files (x86)\LAUNCHERi\LAUNCHER_CONNECTORS\Inciconnector_lib

Installation of the JDBC JARs on the PC/Server hosting the Launcher server:

In order to access the Oracle, MySQL, PostgreSQL, Sybase, DB2 for AS400 or Microsoft SQL Server database, it is imperative that the corresponding JDBC JAR is present in the installation directory of the Launcher server.

For instance:

C:\Program Files (x86)\LAUNCHERi\LAUNCHER_CONNECTORS\Inciconnector_lib.

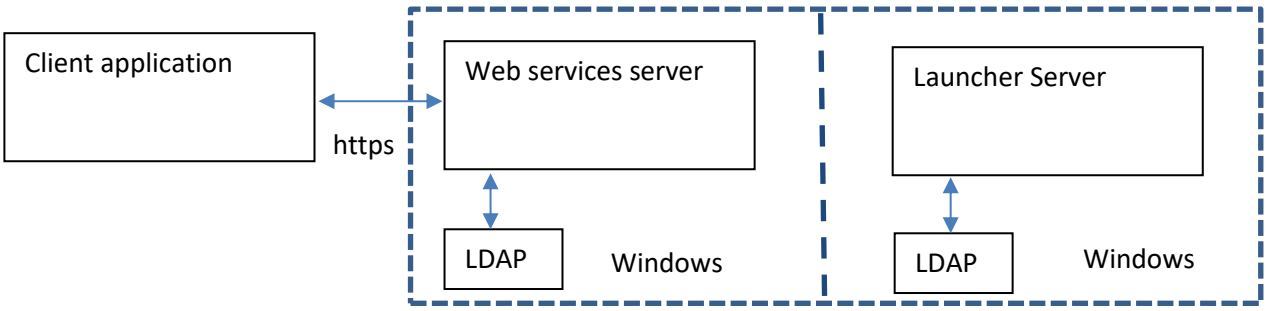
List of official JDBC JARs to use inevitably (no guarantee of operation for any other used JAR):

- For Oracle databases, you must use the Thin driver provided by Oracle.
  For instance, for Oracle Database 10g Release 2, the JDBC Thin driver is named **ojdbc14.jar**, and can be downloaded using the following link:
  http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-10201-088211.html
- For MySQL databases, the JDBC driver **mysql-connector-java-5.1.21-bin.jar** has to be used :
  http://www.mysql.com/downloads/connector/j/
- For PostgreSQL databases (≥ version 7.2), the JDBC driver **postgresql-9.2-1000.jdbc4.jar** can be downloaded from :
  http://jdbc.postgresql.org/download.html
- For Sybase databases (all Sybase products using JConnect-7_0: Adaptive Server Enterprise, SQL Anywhere, Sybase IQ, and Replication Server), the JDBC driver **jconn4.jar** has to be used :
  http://www.sybase.fr/products/allproductsa-z/softwaredeveloperkit/jconnect
- For Microsoft SQL Server (SQL Server2012, SQL Server 2008 R2, SQL Server 2008, SQL Server 2005, and SQL Azure), the JDBC driver **sqljdbc4.jar** has to be used :
  http://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=11774
- For databases IBM DB2 for iSeries, you must use the driver **jt400.jar** supplied with IBM System i Access for Windows. After installing IBM Access on your PC, you can find the JDBC driver, for example here:
  C:\Program Files (x86)\IBM\Client Access\jt400\lib\jt400.jar

After having installed the JDBC JARs, it is mandatory to relaunch Launcher server.

Use the webservices

With the Launcher webservices functionality, the client part can be an application running on any platform. This application calls webservices to place orders at Launcher server. **A webservices server is the gateway between the client application and the server Launcher:**



Webservices can be used with any language, on any platform (Windows, Linux).

Configuration

A LDAP authentication (Active Directory) is necessary for the connection between the client application and the Webservices server. It is done by using the open method.

Into the **"C:\Program Files (x86)\LAUNCHER XPRESS\REST_GATEWAY\configs\authorizedUsers.xml" file,** it is necessary to specify the domain name on which are running the Webservices server and the LDAP server (Active Directory) :

```
<?xml version="1.0" encoding="UTF-8"
standalone="no"?><authentication>
    <!-- authentication_mode : "ldap" for Active Directory  -->
    <authentication_mode>ldap</authentication_mode>

    <ldap>
        <domain>aura.fr</domain>
    </ldap>

</authentication>
```

The LDAP authentification be used:

- for communication between the client application and the webservices server

Put in place the LDAP authentication for the Launcher Server:

If the Launcher Server is not on the same Windows machine and eventually not on the same network, there is the possibility to put in place a LDAP authentication between the Webservices server and the Launcher server.

Into the **launcher.ini** file(*C:\Program Files (x86)\LAUNCHER XPRESS*\launcher.ini), it is necessary to put « UseLdap=Yes » and to specify the domain name:

[LDAP]
*UseLdap=**Yes***
*Hostname=**auraeq.local***

The LDAP authentification be used:

- to establish communication between the webservices server and the Launcher.server. This authentication is established with the **LDAPAUTH** command and must always follow the **LNCOPEN** command. It shall not be taken into account at the end of the communication caused by the **LNCCLOSE** command.

Use a Windows terminal "cmd" as administrator and run the following commands:

cd C:\Program Files (x86)\LAUNCHER XPRESS
notepad launcher.ini

After restart the Launcher server.

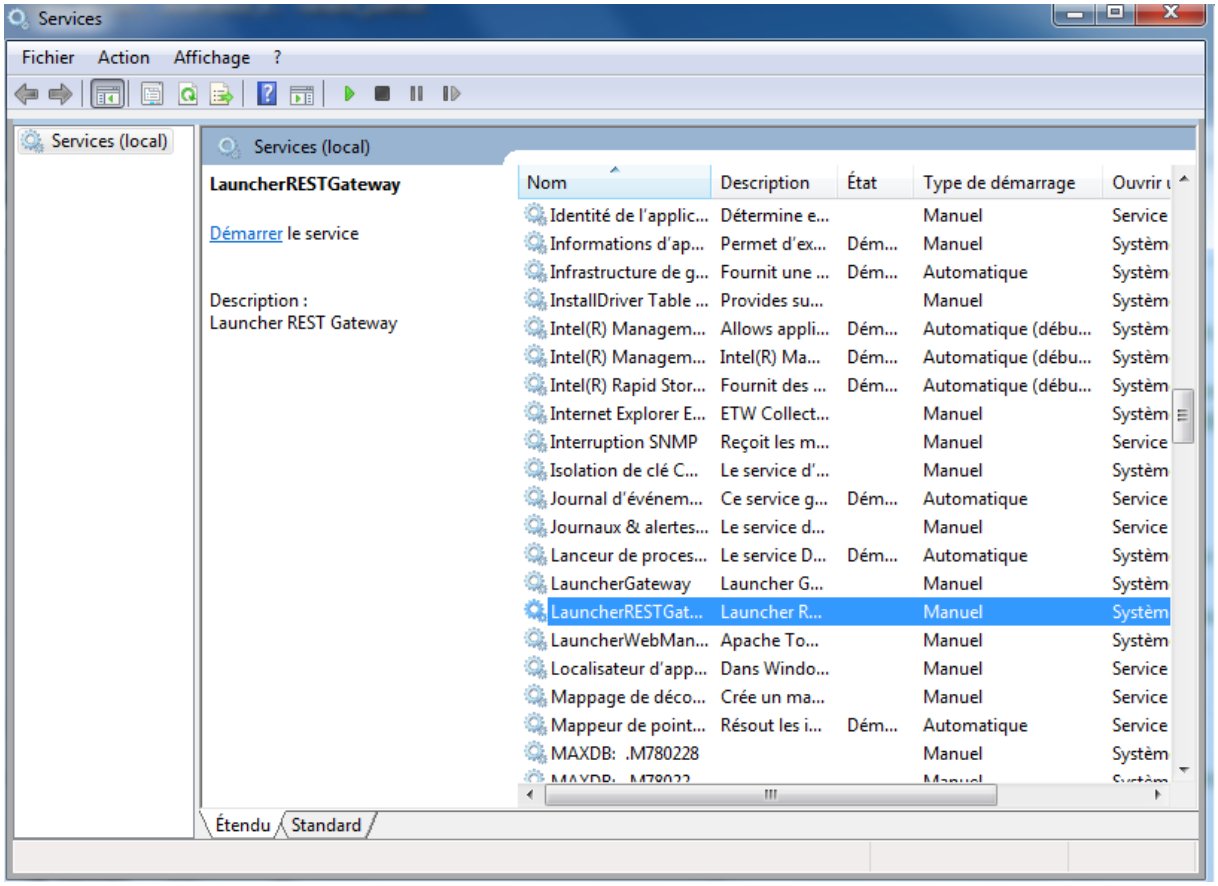
To disable LDAP authentication, it is necessary to put « UseLdap=No » :

[LDAP]
*UseLdap=**No***

Start the webservices server : Launcher REST Gateway

« **Launcher REST Gateway** » has been installed as **Windows service**.

Into the Control Panel :



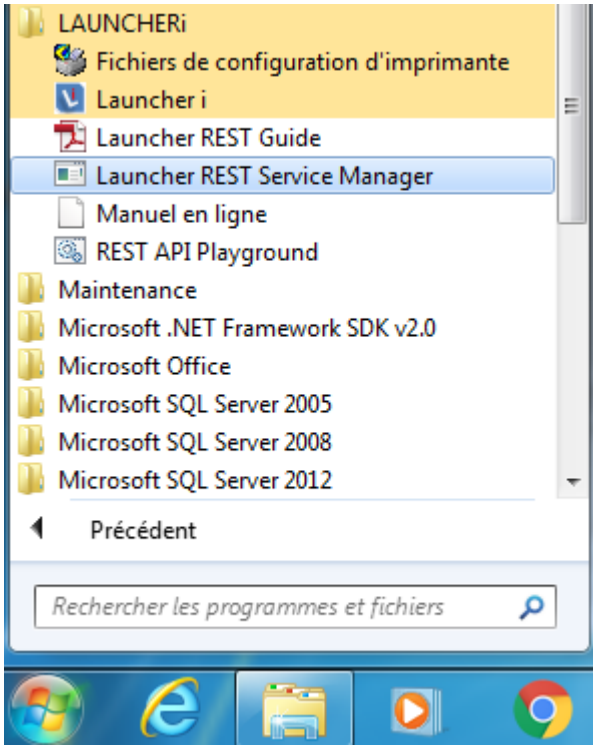
Start the Webservices server : « Launcher REST Gateway ».

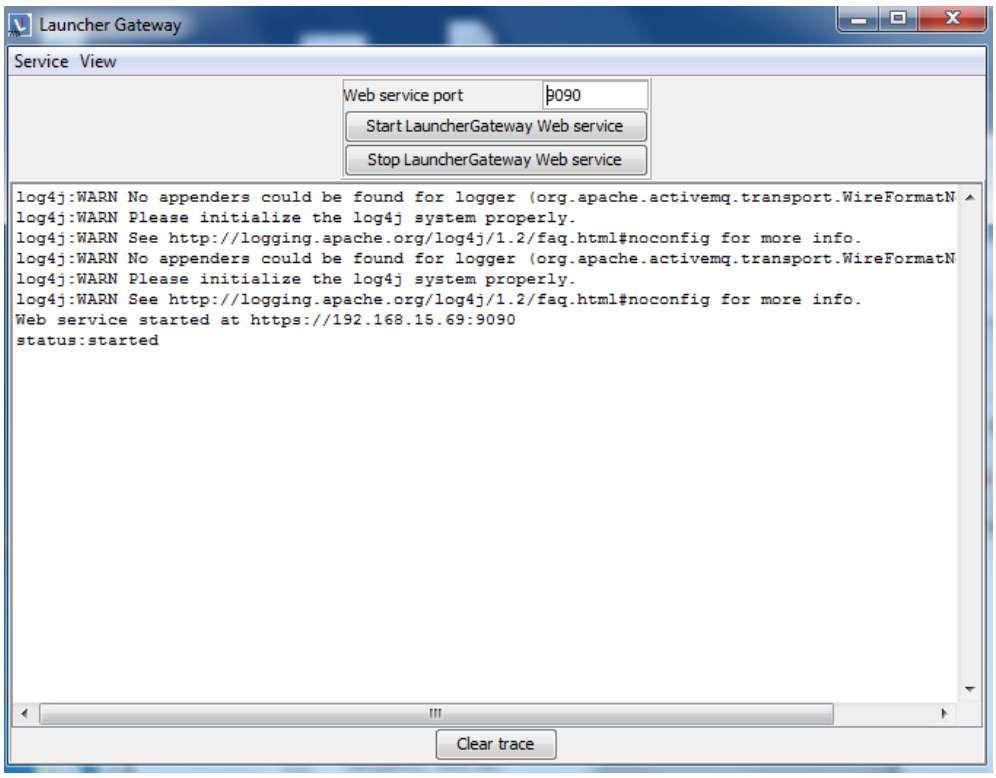
Note:

The webservices server has to be stopped by using the « Launcher REST Service Manager » application.

Launcher REST Service Manager

From the LAUNCHER XPRESS start menu, launch « **Launcher REST Service Manager** » :





You can see that the webservices server is started.
The Launcher server can receive commands to run from the client application, through the web services server.

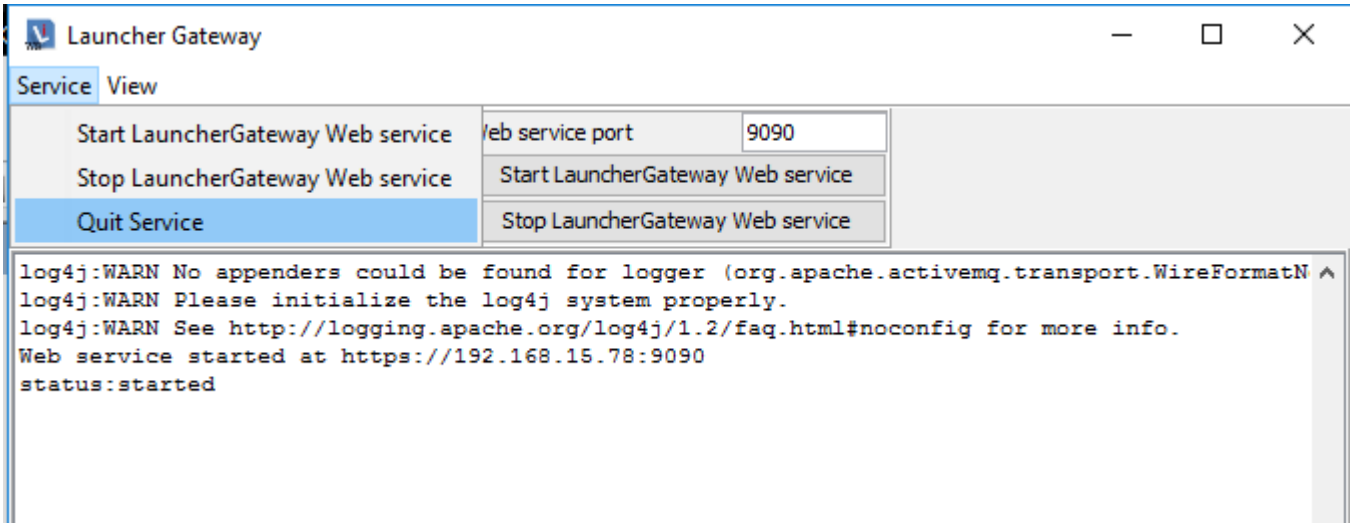
This terminal allows :

- to have a trace of the webservices server activities.
- to stop the webservices server

Stop the webservices server : Launcher REST Gateway

The only way to correctly stop the gateway, is to use the Launcher REST Service Manager from the start menu.

You click on "Quit Service" into the Service menu:



LAUNCHER Administration

LAUNCHER Office program must be run on each PC able to receive a client request, and then becoming a server.

LAUNCHER Office contains two sub programs :

- Executable server (LNVsrv.exe), receiving the client requests.
- Executable controller (LNCadm.exe), allowing to control the server configuration, and to monitor the server activity with the trace.

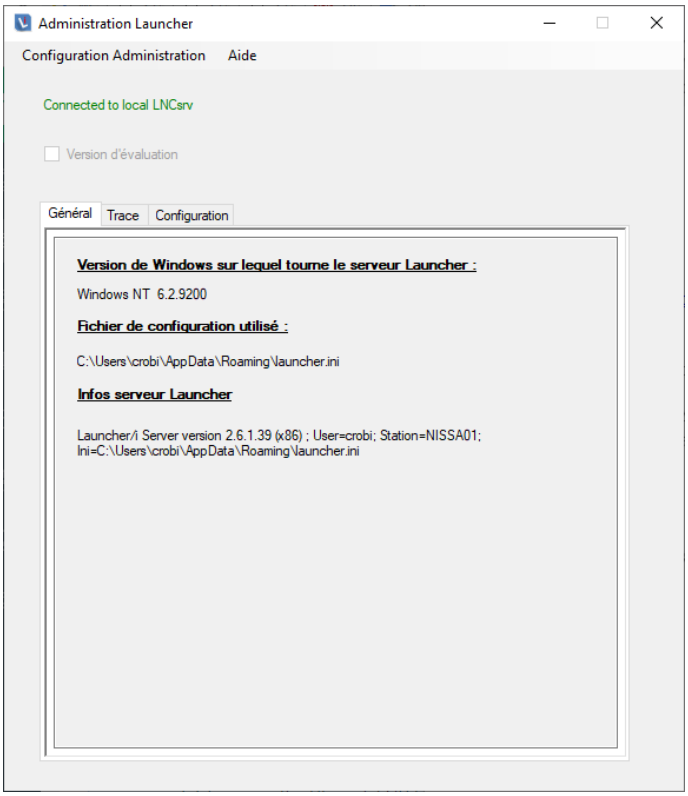
Click on "**LAUNCHER Office**" shortcut at LAUNCHER Office entry in Start Menu. This will launch LNCsrv.exe which in turn will launch LNCadm.exe.

This will have no action if LNCsrv.exe is already running on the station, it will not be launched again. The control window (LNCadm.exe) shows up in local configuration mode.

Those programs run as background task. LNVsrv.exe program stand by waiting for connection with client.

So that **LAUNCHER Office** starts automatically when opening Windows, "LAUNCHER Office" shortcut of LAUNCHER Office entry from the Start Menu must be copied in 'Launching' entry.

LAUNCHER Office server window is divided in three tabs "**General**", "**Trace**" and **"Configuration"**.



Trace tab

This tab contains all instructions sent by clients: connection requests, commands and their parameters and sometime errors description.

A message tells when the server is ready to receive a connection request from the client ("*Waiting for connection*").

When connection is made between the two parts ("*Connection accepted*"), the messages transferred can take place ("*Messages processing started*").

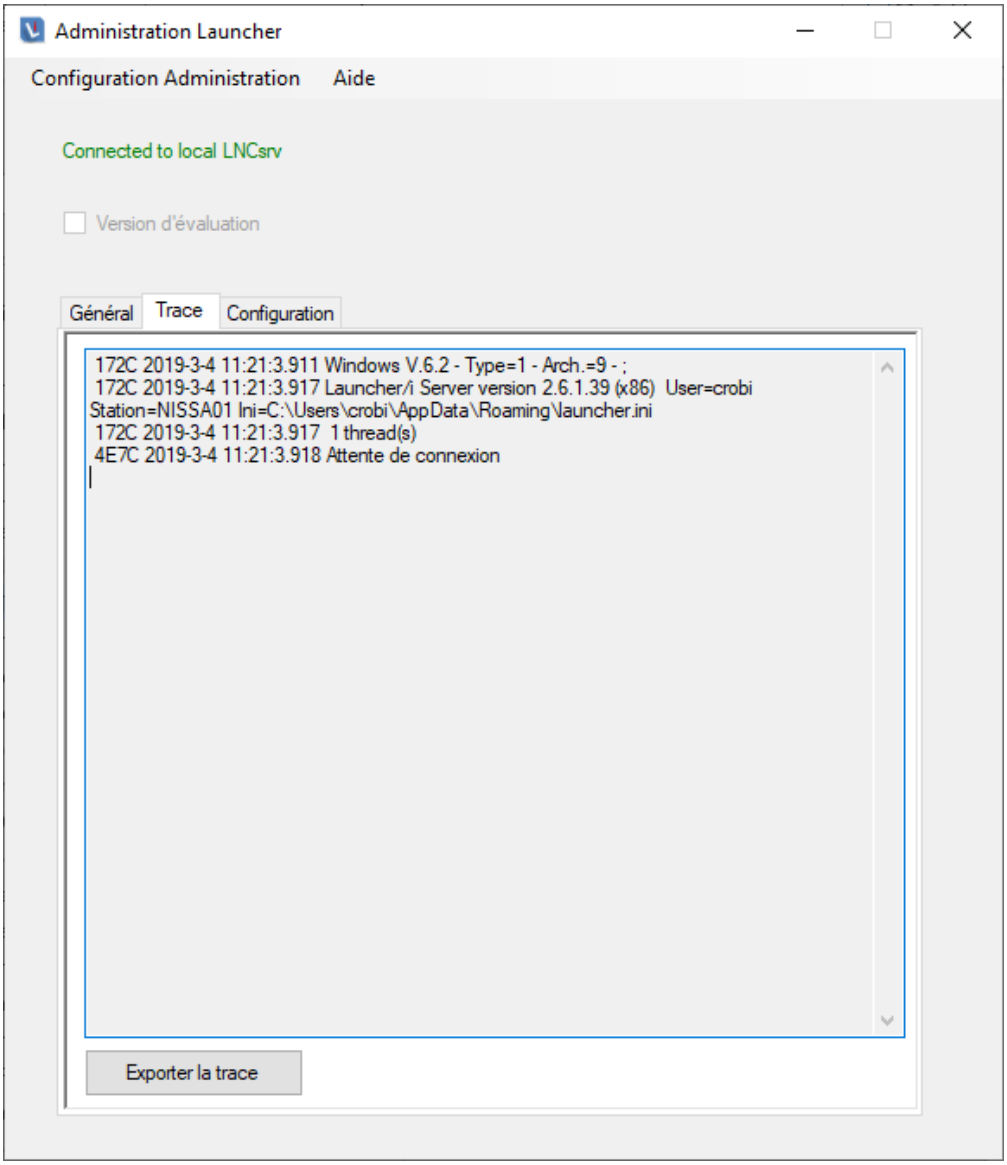
For all received commands, the PC will display : command name, option, the two parameters (Parm1 and Parm2) as well as each element size. For instance : Parm1(33) and Parm2(74).
Errors if any will also be shown.

On receipt of END command sent by the client, the PC closes the connection ("*Transmission Ended*") and returns to connection waiting mode ("*Waiting for connection*").

There is no limit on the displayed trace volume (machine memory only). By cons only the last 500 lines appear.

It may therefore be wise to specify a PC file to save the trace.

"Configuration" Tab allows to display time and/or thread number on each trace line, and eventually to specify a PC file for the trace.

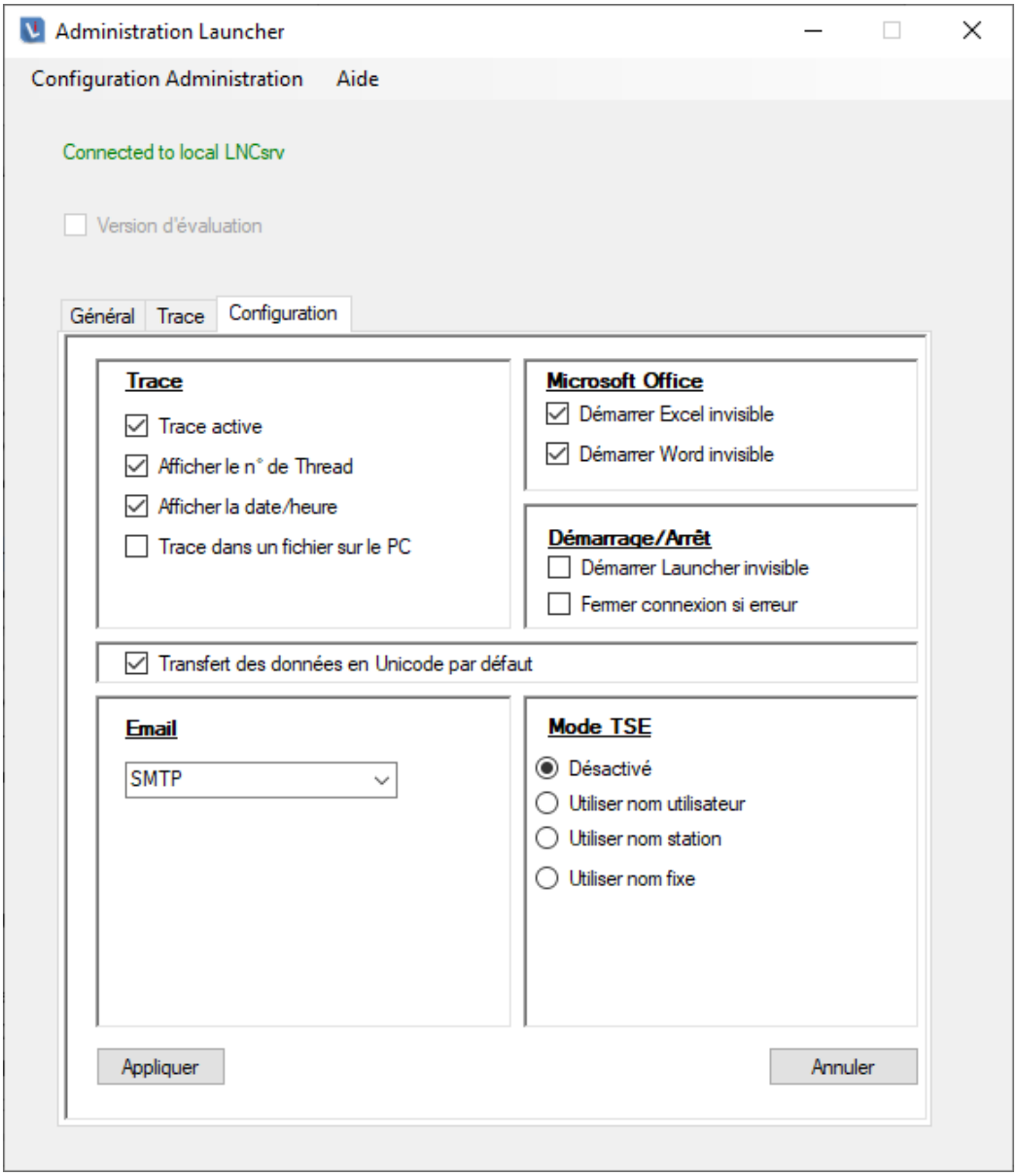


Configuration tab

This tab allows you to define the overall operating conditions of the application program.

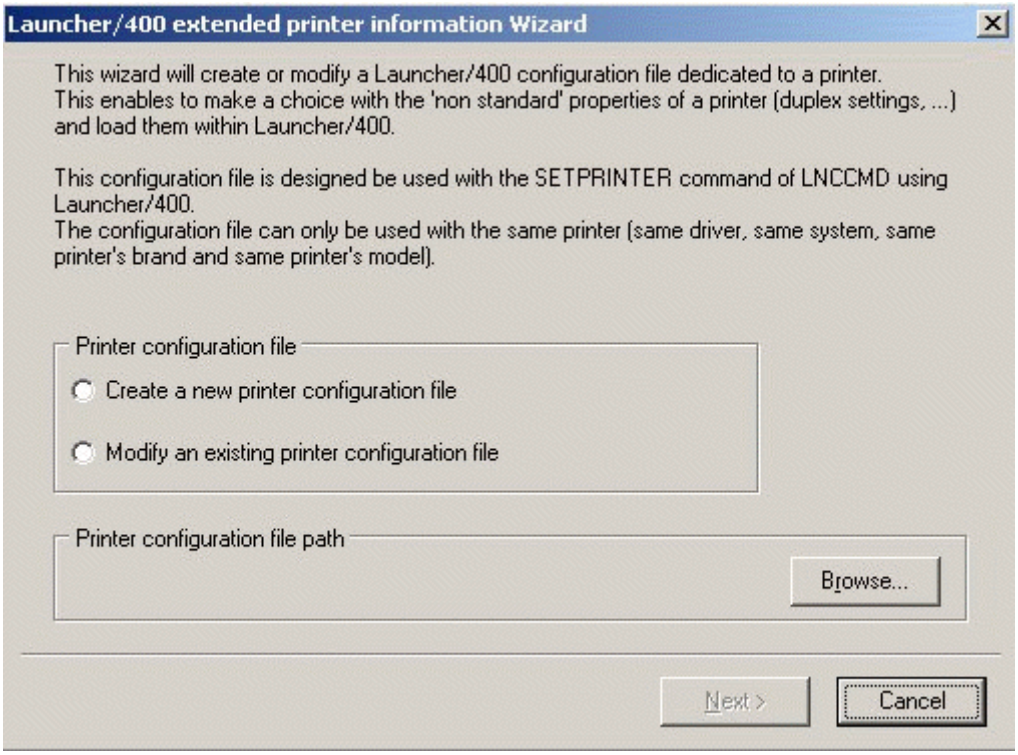
The various options present allow you to make the following settings:

- **Launching Word invisible** : enables you to work without disturbing any other running application, Word will stay hidden until WORDSHOW function is called.
- **Launching Excel invisible** : enables you to work without disturbing any other running application, Excel will stay hidden until EXCELSHOW function is called.
- **Messaging System** : enables you to use the messaging commands to send electronic mails (For more informations see "Advanced programming/Mails").
- **End connection in case of error** : when this option is checked, the connection is automatically closed if any errors occur.
- **Trace Configuration** : enables you to disable the entire trace system, choose a trace toward a PC file or to add thread number and/or date and time for each trace operation.
- **Terminal Services Mode**: allows you to switch to Terminal Services Mode. An AS/400 IP address and a station name have to be selected if this option is activated. See "Configuring LAUNCHER Office on Windows Terminal Server".
- **Launch LAUNCHER Office in hidden mode** : if this option is checked, launching LAUNCHER Office using the usual icon will not launch LAUNCHER Office Control. In this mode, LAUNCHER Office Control must be launched manually in order to change the configuration or access to the memory trace.



Printer configuration file

This wizard will create or modify a configuration file dedicated to a printer.



See also

- Command SETPRINTER
- Command RSTPRINTER

Remote Administration

During set-up you can select "LAUNCHER Office Administration Tools", if you are entitled with **Administration Rights** at that time.

Use 'Start' Menu and then "LAUNCHER Office – Administration" in order to launch Administration from a remote station.

The User interface looks like Administration Tool interface launched with "LAUNCHER Office" icon but with more tabs :

- "**Local Configuration**" : allows to control the local LAUNCHER Office instance, hidden or not.
- "**Network Configuration**" : allows to remote control any LAUNCHER Office version 2 within the local network.

Remote control introduces several limitations :

- The user must be entitled with Network Administration Rights to see the supplementary tabs.
- The LAUNCHER Office instance to control must be running and 'iconised' in order to be remote controlled.
- The LAUNCHER Office instance to control shall not be already under control.

The list of LAUNCHER Office instances that can be remote controlled will be displayed (according to the above mentioned criteria's) if the Remote Control Tab is selected. A click on the chosen machine allows access to its configuration.

This list only contains the local network machine mask. To control others machines, it is necessary to modify the launcher.ini file of the machine to modify (see section 'Configuration file').

The General Tab displays : system version, LAUNCHER Office version, and Launcher.ini file path.

General Tab also displays the current operating mode : connected or unconnected. Unconnected mode means that no LAUNCHER Office instance were found in the current operation (example : no LAUNCHER Office running on the local machine). In this case the administrator version appears.

The configuration may be read and modified from remote. Warning! Modifying some options induces **automatic** LAUNCHER Office **restart**, **ending the connection** with the client. This may have consequences if LAUNCHER Office is currently used.

The properties inducing automatic restart if applying modifications are as follow :

- Terminal Services connection mode
- Terminal Services local name (in Terminal Services mode)
- AS/400 name or IP address (in Terminal Services mode)
- Messaging system type
- Lotus Notes password
- MAPI profile name

A LAUNCHER Office instance can be controlled in NT Service from a local station or a local network station.

If you operate from a Terminal Services server, an additional tab appears : "Users Configuration". It allows to modify the local launcher.ini files for each users who have been once connected to the machine (in this case LAUNCHER Office does not need to be running).

In unconnected mode, (example if LAUNCHER Office local instance could not be found), configuration can anyway be modified, writing in this case, directly in launcher.ini. file. This is the operating mode on "Users configuration" tab.

Configuration File

All settings that can be modified by LAUNCHER Office administrator, are present into the **launcher.ini** file**.**

LAUNCHER Office uses **launcher.ini** file to know the configuration.

LAUNCHER Office Control Program indicates on the main Tab the exact path to **launcher.ini** file currently used.

(See : Configuration Tab)

When Launcher is started the first time (from the Start/LAUNCHER<x>/Launcher<x> menu), the **launcher.ini** file used is:

C:\Users\<user>\AppData\Roaming\launcher.ini.

It will be created here if changes to the Launcher configuration are made, and no **launcher.ini** file exists.

Otherwise in descending order of priority, **launcher.ini** is searched in the following locations:
- C:\Program Files (x86)\LAUNCHER<x> (or C:\Program Files\LAUNCHER<x>)
- C:\Windows
- C:\Users\<user>\AppData\Roaming

If the profile used to open the Windows session does not have the rights to modify the **launcher.ini** file, then the boxes of the configuration are grayed out: no modification possible. You have to run Launcher "as administrator" to change the configuration.

If Launcher is started as a **Windows service**, **launcher.ini** is searched in the following locations in descending order of priority:
- C:\Program Files (x86)\LAUNCHER<x> (or C:\Program Files\LAUNCHER<x>)
- C:\Windows

It will be created in "C:\Windows" if changes in the Launcher configuration are made, and no **launcher.ini** file exists.

If the configuration of a Launcher server is changed **remotely** with "Launcher<x> - Administration" (from the Start/LAUNCHER<x> menu), **launcher.ini** is searched in the following locations:
- C:\Program Files (x86)\LAUNCHER<x> (or C:\Program Files\LAUNCHER<x>)
- C:\Windows
- C:\Users\<user>\AppData\Roaming

Since the 2.6.1.22 version, it is possible to put the launcher.ini file as parameter.
Example: lncsrv.exe /ini "C:\temp\launcher.ini"

LAUNCHER Office Control Program modifies **launcher.ini** content but some options are available only if the file is directly modified.

Those options are as follow :

- **Section [CONFIG]**
  - **MaxThreads**=n

    Define the simultaneous connections maximum number. Default Value : 1 (see : Multiple connection mode.)

- **Section [NETWORK]**
  - **PortNumber**=n

    LAUNCHER Office TCP/IP port number. Default value : 6078
  - **LNCD_PortNumber**=n

    LAUNCHERD TCP/IP port number. Default value : 5681
  - **Admin_PortNumber**=n

    TCP/IP (UDP) port number used for LAUNCHER Office remote control. Default value : 6079
  - **Admin_PortNumber2**=n

    TCP/IP (UDP) port number used for LAUNCHER Office remote control. Default value : 6080
  - **Admin_PortNumber2End**

    TCP/IP (UDP) maximum port number used for LAUNCHER Office remote control. Default value : 6085. Used in case of multiple control on TSE server. This number can be increased to allow more simultaneous controls on the server.
  - **RetriesDelay**

    Delay in second between two connections tries to LAUNCHERD. Default Value : 30. Used in TSE mode only.
  - **AdminFromIPAllowxx**=ipaddress

    Indicates one or several IP addresses enabling LAUNCHER Office station control. This is useful in order to control a machine from different network mask. Several addresses or full network can be listed, in this case the prefix next to option must be incremented.

    Example:
    [NetWork]
    AdminFromIpAllow1=192.206.160.5
    AdminFromIpAllow2=192.206.161.255

    In the above example, the '192.206.160.5' machine and all machines beginning from '192.206.161.' are allowed to remotely control this particular machine.

    N.B. In any cases, all machines from the local network branch are allowed to operate remote control.
    AdminToIpxx option can also be used on the control machine.
  - **AdminToIPxx**=ipaddress

    Indicates one or several IP machines addresses to be remotely controlled.
    By default, LAUNCHER Office administrator can access to LAUNCHER

Office instances located on the same network mask as the local machine. This option allows control of other machines **if they accept connection from the machine where LAUNCHER Office controller run** (see option 'AdminFromIPAllowxx').

Example :
[Network]
AdminToIP1=192.206.162.4
AdminToIP2=192.206.162.5

In the above example, we want to be able to control the 192.206.162.4 and 192.206.162.5 machines.
Notice: No generic address are allowed in this case (example 192.206.162.255)

Windows Terminal Server and AS400

This configuration is not only for Windows terminal Server or Citrix environments.

When the user PCs don't have a unique IP address, or, when the 5250 emulation is handled through a front end, then, the "TSE mode" configuration can be used.

In standard configuration (TSE not active), LAUNCHER server on the PC is waiting for a call comming from an AS/400 job.

The AS/400 program needs to know the IP address of the PC or its DNS name, to be able to contactd it via LNCOPEN.

With special value "*DEV", LAUNCHER retreive the PC IP address from the display device description.
If the IP address cannot be known from the AS/400 program, or, if multiple PCs have the same IP address (like when TSE or CITRIX is used), then, the way of the conversation must be changed by configuring the LAUNCHER Office TSE mode.

This configuration is set on the PC by the "Configuration" tab on the administration screen.

Note :The two configurations can be used together with the same AS/400.

Special features of AS/400 installation:

- Launch LAUNCHERD deamon on AS/400:

ADDLIBLE LAUNCHER

STRLNCD MSGLVL(1) <F4>

This program must be launched under a profile with priority rights on other profiles, otherwise the LAUNCHERD program from LAUNCHER must be modified so that it will run with QSECOFR :

CHGPGM LAUNCHER/LAUNCHERD USRPRF(*OWNER)

A deamon LAUNCHERD program will be launched.

If the default JOBQ value is used, LAUNCHERD will be launched in QSYSWRK sub system.

The **DSPLNCD** command is used to view connected workstations

Special features on the TSE server :

Running LAUNCHER Office on a all users "Terminal Services" Server requires a LAUNCHER Office instance that will operate for each user.

These instances may come in addition to another LAUNCHER Office instance as NT service.

To manage those many LAUNCHER Office instances a station cannot be identified anymore only with its IP address.

Each LAUNCHER Office TSE user must have a **unique name**, and indicate its **AS/400 IP address**.

So, on configuration screen, one of the "Terminal Services" modes must be selected (Configuration tab).

Name choice can be made either from :

- **Using the user name** option: WINDOWS user name will be used as name.
- **Using the station name** option: the PC name connected to TSE (user station) will be used as name.
- **Using a fixed name** option : allows choice of any name (in this case you have to check if this name does not already exist in the TSE server).

Your application will have to manage the names, for example, using the user name as unique local name assuming that the user connects from only one station.

When the TSE mode is on (LAUNCHERD is running on the AS/400), when passing a name to LNCOPEN, LAUNCHER server looks for the PC in the following order:

- Look for a PC having the requested "TSE" name.
- Look for a PC having the requested "DNS" name.

Configuring 5250 emulator in the same way (example station name), allows to use *DEV name in LNCOPEN command.
Otherwise, use the corresponding name.

When **"*DEV"** is used, and LAUNCHERD is running, LAUNCHER Server looks for the PC as follow:

- Looks for a PC having a "TSE" name equal to the current job name (see job name filtering here after).
- Connect to the display device IP address.

Job name filtering:

If you configure your Client Access emulator, to build the terminal name from the station name, you will be able to use "*DEV" on LNCOPEN calls.

Change the content of the data area LNCNET, position 1 to 100:

CHGDTAARA DTAARA(LNCNET (1 100)
VALUE(`AUTO;STNNAME=<*filter*>**;')**

The value of <filter> tells LAUNCHER how the terminal name is built.

<filter> can have the following values :

- **+&COMPN** No prefix or suffix is added.
- **%+&COMPN** One prefix character is added to identify a display or printer.
- **+&COMPN*** One suffix character is added to have unique names on the station.
- **+&COMPN=** One suffix character is added to have unique names on the network.
- **+&COMPN*=** Two suffix characters are added.

- **%+&COMPN*=** One prefix and two suffix added.

Note : Those values are used by Client access in the display definition files (Files .WS).

Example :

The station name is **PAUL**.

The LAUNCHER server on the PC is configured in TSE mode, using the station name as Launcher name.

Client access is configured to add the prefix and two suffix. The current job name is : **SPAULA1**

The data area value must be :

AUTO;STNNAME==%+&COMPN*= ;

Note :

Using a generic name (user name or station name) makes it possible to configure LAUNCHER Office in unique way valid for ALL users. To do so, set up the configuration on one station and then make a copy of the user launcher.ini file in LAUNCHER Office directory.

See section : "**Launcher.ini**".

If any problems occur, display the PC trace and see AS/400 message file :

DSPMSG LAUNCHER/LNCMSGQ

Multi Connections Mode

LAUNCHER Office can handle **simultaneous** requests.

Those requests may come from several client programs currently running.

You have to set a maximum number of simultaneous connections using **launcher.ini** following options :

[Config]
MaxThreads=nn

nn being the maximum simultaneous connections number.

Default value is 1. If there is a current connection, this means that any new connection will wait until the current connection stops.

Warning ! Operating with simultaneous connections requires programs handling common objects sharing (files, clipboard, etc.).

Concurrent accesses

To do safe Launcher Office programs concerning concurrent accesses, it is necessary to control shared resources accesses and making sure they will not be used simultaneously.

For instance, assuming a program using the "a.txt" file with DBXFER command. This program is safe in terms of competitive access to a program B that would use the "b.txt" file, but not in relation to **itself**.

Therefore, we have to make sure that program A is never run simultaneously, or solve the **critical resource** problem.

There are different ways to solve the above problem :

- Using a variable file name (to be set by program). This seems to be the best solution as it enables several program A instances to run fully simultaneously.

  %CONID% variable assures this unicity to a particular LAUNCHER Office instance (care must be taken to files used by several stations being on the network).
- Using a '**critical section**' with LAUNCHER **'CRITSECT'** command. This command will protect a common resource.

  In this case, just before using the a.txt file, enter the following command

  LNCCMD CMD(CRITSECT) PARM1('Name="ProgramA";Enter')

  Then, just after :

  LNCCMD CMD(CRITSECT) PARM1('Name="ProgramA";Leave')

It will always be better to avoid conflict (when possible) instead of using critical sections.

Critical sections are automatically released with LNCCLOSE command.

Clipboard

Clipboard when used with **WCOPY** and **WPASTE**, **XLCOPY** and **XLPASTE** commands is already secured.

Warning the following rules must be observed :

- Always do a 'Paste' after a 'Copy' (another program using the clipboard may be stacked).
- Do not do several 'Paste' and only one 'Copy' (there will be not content safety for the next 'Paste' as the first 'Paste' as already released the critical resource').

Clipboard Critical section is named **'CLIPBOARD'**. It must be used if the clipboard is used in different ways than with WCOPY/WPASTE/XLCOPY /XLPASTE functions.

Trace Analysis

The ' thread number' column must be displayed in order to correctly read the trace knowing which 'connection' is working and finding out how each program runs.

Each session has a constant thread number.

Example:

```
00000354 Waiting connection
00000354 Waiting connection
00000414 LAUNCHER/400 V:Launcher server version 2.0.0.0 - running as
Service
00000414 Connection accepted
00000414 Messages processing started
00000354 Waiting connection
0000034C LAUNCHER/400 V:Launcher server version 2.0.0.0 - running as
Service
0000034C Connection accepted
0000034C Messages processing started
0000034C >CRITSECT,Opt=
0000034C *ACK
00000414 >WORDOPEN,Opt=
00000414 LNC/Word Module=C:\progra~1\launcher400\LNCWORD.dll;
V.2,0,0,0
00000414 MS Word V.9.0
00000414 *ACK
```

In this case two sessions are opened: threads '414' and '34C'.

Errors management

Word and Excel OLE interfaces allow to generate very clear error messages (this file does not exist, …).

LNCCMD program generates an error on AS/400 side. Refer to JOBLOG to find out the reason.

Errors messages displayed in Trace tab are also sent to the client.

&RESULT return variable allows to display the messages (See : Section **LAUNCHER Office** Programming).

Programming with LAUNCHER Office

LAUNCHER Office Programming

LAUNCHER Office provides a programming interface dedicated to :

- any programming language developers, using **REST webservices**, on Windows and Linux
- Java developer using **Java** classes on Windows/Linux
- RPG, COBOL and CL developers on **AS/400**.

The following tools are provided to the programmer :

1) LAUNCHER Office **API base Programs** set for every client (Windows, Linux or AS/400):

- LNCOPEN, to open a conversation with a PC.
- LNCCMD, to send commands to a PC.
- LNCLOSE, to close communication with a PC.

2) High level **CL command** kit for AS/400 client:

- LNCPRTDOC, to lay out and print Word document.
- LNCXFER, to transfer a file or a request to a PC.
- LNCTOXLS, to transfer data to an Excel folder.
- LNCSHELL, to launch Windows commands or programs
- CPYTOMDB, to transfer data to MS Access.
- LNCSNDMAIL, to lay out and send an electronic message.

3) Some Java classes:

- LNCPRTDOC
- LNCTOXLS
- DataSource

4) Webservices providing equivalent methods than the API (Open, Cmd, Close) and others like Filetransfer and Ldapauth.

Notes for AS/400 client:

The LAUNCHER Office (LNCOPEN, LNCCMD and LNCCLOSE) programs can be called by using the CALL instruction.

LAUNCHER Office will work with ILE as well as non ILE environment.

LAUNCHER Office programs run in batch as well as in interactive mode.

Operating Principle

Any program whatever the language and platform used (Windows or Linux) can use the provided webservices

Another possibility for the Java client (Windows and Linux) and AS400: use the API (LNCOPEN, LNCCMD and LNCCLOSE).

A client program (Java or CL) using the API has to follow the following steps :

- Open a conversation with API program "LNCOPEN".
- Send several commands to be run on the PC using "LNCCMD".
- Close the conversation with API program "LNCCLOSE".

Windows applications (Word, Excel) are starting and closing by client program are processed through API program "LNCCMD".

A Java client can also use the provided Java classes.

An AS400 client can use the provided CL programs.

Notes for AS/400 client:

LAUNCHER Office (LNCPRTDOC, LNCXFER, LNCTOXLS, …) high level CL commands, support communications opening and closing as well as starting and stopping Windows applications.

If only CL commands are used, the developer shall step over the operation description above.

LNCCMD commands

LNCCMD key words

The **LNCCMD** command requires the followings parameters:

CMD, PARM1, PARM2 and RESULT.

The **CMD** parameter value indicates which PC action is requested by client program.

The values given to the parameters **PARM1** and **PARM2** of the LNCCMD command depend on the action requested on the PC by the key word CMD.

Description of those two parameters for each key word can be found in the alphabetical list.

RESULT parameter contains a return value for LNCCMD program call.

Use of some keywords for CMD parameter induces a return result.

You can find the list of keywords for the **CMD** parameter, by consulting the following chapter: "Alphabetical list".

For this parameter, possible values of keywords may be classed in categories :

- **Wxxxx** : Words beginning with W concern a MS Word action.
- **XLxxxx & EXCELxxx** : Words beginning with XL or EXCEL concern a MS EXCEL action.
- **MAILxxxx** : Words beginning with MAIL concern the messaging system.
- **OLExxxx** : Words beginning with OLE concern OLE object links.
- **PDFxxxx** : Words beginning with PDF concern PDF action.
- ***(Others)*** : Others words concern general actions concerning Windows or others applications.

Call LNCCMD command from AS/400 client in CL program :

```
CALL PGM(LNCCMD) PARM(&HANDLE &OPT &CMD &PARM1 &PARM2 &RESULT)

Or

LNCCMD      CMD(EXCELOPEN)
```

Call LNCCMD command into Java program :

```
test.LNCCMD("EXCELOPEN ");
```

Use LNCCMD in Java : example

```
package test;

import fr.aura.launcher.wrapper.LNCClientWrapper;
import fr.aura.launcher.wrapper.LNCSrvAckMsg;
import fr.aura.launcher.wrapper.LauncherException;

public class TestCMD
{

	public static void main(String[] args)
	{

		String pSvrAddr = "127.0.0.1";
		int pSvrPort = 6078;
		LNCSrvAckMsg iRetLNCCMD = null;

		LNCClientWrapper test = new LNCClientWrapper();

		test.LNCOPEN(pSvrAddr,pSvrPort);

		try
		{
			iRetLNCCMD = test.LNCCMD("WORDOPEN",null,null,null);
		}
		catch (LauncherException e)
		{
			e.printStackTrace();
		}
		System.out.println(iRetLNCCMD.getMsg());

		Parm1 = "C:\\temp\\essai2.docx";
		String Parm2 = "visible";


		try
		{
			iRetLNCCMD = test.LNCCMD("WOPENFILE",null,Parm1,Parm2);
		}
		catch (LauncherException e)
		{
			e.printStackTrace();
		}
		System.out.println(iRetLNCCMD.getMsg());

		Parm1 = "File=C:\\temp\\testPdf_9b6_result.pdf;EmbedFonts=True";

		try
		{
			iRetLNCCMD = test.LNCCMD("PDFPRINTER",null,Parm1,null);

		}
		catch (LauncherException e)
		{
			e.printStackTrace();
		}
		System.out.println(iRetLNCCMD.getMsg());
```

```
                Parm1 = "Printer=\"LAUNCHER_PDF\"";


                try
                {
                        iRetLNCCMD = test.LNCCMD("WPRINT",null,Parm1,null);
                }
                catch (LauncherException e)
                {
                        e.printStackTrace();
                }
                System.out.println(iRetLNCCMD.getMsg());

                Parm1 = "Save=false";


                try
                {
                        iRetLNCCMD = test.LNCCMD("WORDCLOSE",null,Parm1,null);
                }
                catch (LauncherException e)
                {
                        e.printStackTrace();
                }
                System.out.println(iRetLNCCMD.getMsg());

                test.LNCCLOSE();

        }


}
```

Commands list

Examples of using commands are given in CL, but the commands can also be used in a Java program.

You must refer to the example on the previous page.

FTPLISTDIR
FTPMKDIR
FTPPUTDIR
FTPPUTFILE
GETREGE
GETSYSINFO
GRAPHSEND
IFSPUT
LISTSCAN
LNCDBGET
MAILATT
MAILBODYF
MAILCC
MAILEND
MAILLATT
MAILLMSG
MAILMCHG
MAILMDLT
MAILPREP
MAILPRTY
MAILREPORT
MAILSEND
MAILSMTP
MAILSUBJ
MAILTEXT
MAILTO
MAIXATT
MENU
MENUWAIT
MIN
MOVEFILE
MSGBOX
NCL
NLN
NOP
OLECALL

OLECREATE
OLERELEASE
PARSEXML
PDFALLXL
PDFCONCAT
PDFCOUNT
PDFENCRYPT
PDFEXTRACT
PDFFACTX
PDFFRIM
PDFFRXL
PDFMERGE
PDFORDERX
PDFOVER
PDFPRINT
PDFPRINTER
PDFSEARCH
PDFSIGN
PDFSPLIT
PDFTOIM
PRINTPRNF
PROPERTY
RCL
REMOVEDIR
RSTPRINTER
SAISI
SAISIWAIT
SCAN
SELECTFILE
SETPRINTER
SHELL
STO
VAL
WADDINS
WBOOKMADD
WBOOKMARK

WBOOKMDEL
WCHDIR
WCLOSEFILE
WCOPY
WDOCUMENT
WDPROTECT
WEXEMACRO
WFIELDS
WFINDTEXT
WFORMFIELD
WGETHPOS
WGETPAGE
WGETPROP
WGETTEXT
WGETVPOS
WGOTOTBL
WINSERTOBJ
WINSERTBRK
WINSERTCOL
WINSERTF
WINSERTIMG
WINSERTROW
WINSERTTBL
WMAILMERGE
WMAXIMIZE
WMENU
WMERGECELL
WMETHOD
WMINIMIZE
WMOVE
WMOVERIGHT
WNEWFILE
WOPENFILE
WORDCLOSE
WORDHIDE
WORDOPEN

WORDSHOW
WORDWAIT
WPAGEBRK
WPASTE
WPRINT
WPROTECT
WREFVAR
WREPLACE
WSAVE
WSAVEAS
WSELECT
WSELROW
WSETLINE
WSETVAR
WSETPASSWD
WSETPROP
WSPLITCELL
WTYPETEXT
XLADDINS
XLADDSHEET
XLBOLD
XLCALCULAT
XLCELLNAME
XLCELLS
XLCLOSFILE
XLCOPY
XLCOPYSH
XLDELSHEET
XLDISADD
XLDPROTECT
XLDRAWGR
XLENAADD
XLEXEMACRO
XLEXISTSH
XLFIND
XLFORMULA

XLGETFILE
XLGETPOS
XLGETPROP
XLGETVALUE
XLGOTOCELL
XLGOTOSH
XLINSERT
XLINSERTCL
XLINSERTRW
XLITALIC
XLMAXIMIZE
XLMENU
XLMETHOD
XLMINIMIZE
XLOPENFILE
XLPAGEBRK
XLPASTE
XLPRINT
XLPROTECT
XLREPLACE
XLRESIZE
XLSAVEAS
XLSETLINE
XLSETPROP
XLSETTEXT
XLSETVALUE
XLSHNAME
XLSORT
XLSUBTOTAL
XLUNDERLN
XLWAIT

CHKDIR command

Allows to verify if a Windows directory exists.
248 characters max for the directory path.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('CHKDIR')
CHGVAR              VAR(&PARM1) VALUE('
                    Dir="Directory path to verify"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Dir** : Complete path of the directory to verify.
RESULT	TRUE : if the directory exists. FALSE : if the directory does not exist.

Example

```
LNCCMDR     CMD(CHKDIR) +
               PARM1('DIR="C:\AURADATA\SUPPORT\TKM"') +
               RESULT(&RES)
LNCCMD      CMD(NOP) PARM1(&RES)
```

CHKFILE command

Allows to check if a file or a directory is present on a PC, and the granted rights on a file.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('CHKFILE')
CHGVAR              VAR(&PARM1) VALUE('
                    "File"
                    [;Read]
                    [;Write]
                    [;Wait=number od seconds]
                    [;GetInfo]
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	Full path to file or directory. **Read** : Specify this option to check file reading right. **Write** : Specified this option to check file writing right. **Wait**=nnn; nnn is a number of seconds. Delay for waiting for the file availability. **GetInfo**=True; To receive into RESULT the informations of the file. Position 7 : **AAAAMMJJ** : Date last modification. Position 16 : **HHMMSS.mmm** : Time last modification. Position 27 : **TTTTTTTTTTT** : Size in bytes (12 characters). Position 40 : **D** if the path corresponds to a directory.
RESULT	**&RESULT** parameter returns one of the following values : Position 1 : TRUE / FALSE / DENIED / SHV Position 7: Information of file if **Getinfo**=True TRUE : The file exists, the requested rights are granted. FALSE : The file does not exist. DENIED : The file exists, but the reading and/or writing requested rights are not granted. SHV: The file exists, but it is used by another application.

Example

```
CHGVAR VAR(&CMD) VALUE('CHKFILE')
CHGVAR VAR(&PARM1) VALUE('"%LNCDIR%\SAMPLES\LNCMENU.DOC";READ;+
WRITE')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

Other example

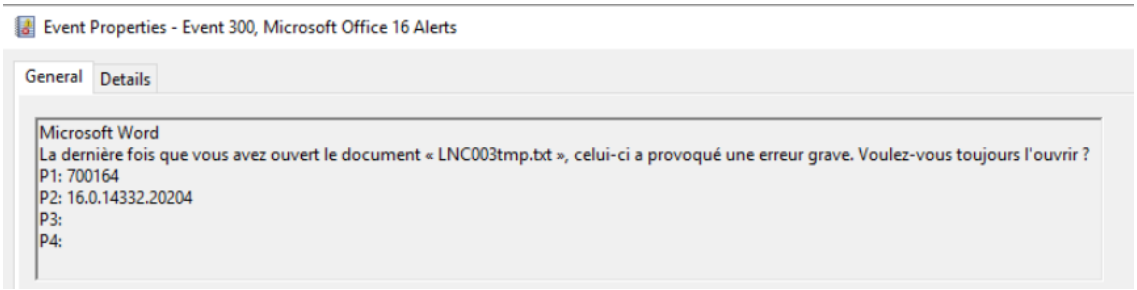
```
LNCCMDR    CMD(CHKFILE) PARM1('"' *TCAT &OUTDIR *TCAT +
              '\' *TCAT &NOMLOT *TCAT +
              '.pdf";Getinfo=true') RESULT(&RESULT)
```

CLEANREG command

When a mail merge (WMAILMERGE command) ends in error with Launcher, Microsoft Word may disable the temporary data file (%TEMP%\ LNC00xtmp.txt). From then on, all subsequent mail merges will not work.

In the Launcher trace, you will have the following message:
*REJ800706be:The remote procedure call failed

In addition, in the Windows Event Viewer, you will have this information:



The CLEANREG command is used to reactivate temporary data files.

Note that this processing is done automatically when the Launcher service starts. This command is therefore only useful when Launcher is launched as an application.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('CLEANREG')
CHGVAR              VAR(&PARM1) VALUE('
                    Appli="Word"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Appli**: Currently the only possible value is: **Word**.

Example

```
CHGVAR VAR(&CMD) VALUE('CLEANREG')
CHGVAR VAR(&PARM1) VALUE('Appli="Word"')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

COPYFILE command

Allows to copy a file from a directory to another one on PC.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('COPYFILE')
CHGVAR              VAR(&PARM1) VALUE('
                    File="File to copy"
                    NewFile="Copied file"
                    [;Replace=true/false]
                    [;ReadOnly=true/false]
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File** : Complete path and name of the file to copy. **NewFile** : Complete path and name of the copied file. **Replace** = true, to automatically replace the copied file if it already exits. Default value = false. **ReadOnly**=true, to specify the READ-ONLY attribute on the copied file. Default value = false.
RESULT	Contains eventually error message.

Example

```
LNCCMD CMD(COPYFILE) PARM1('File="C:\temp\SBOOK_template.xls";
+ NewFile="C:\temp\TEST\copy2.xls";replace=true')
```

CREATEDIR command

Allows to create a Windows directory.

Several directories (intermediate and final) can be created at once, if the FullPath option is equal to true.

It will only create the final directory specified into the path, if the FullPath option is equal to false (or absent).

The security descriptor for the directory is inherited from its parent directory.

248 characters max for the directory path.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('CREATEDIR')
CHGVAR              VAR(&PARM1) VALUE('
                    Dir="Directory path to create"
                    [;FullPath=true/false]
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Dir** : Complete path of the directory to create. **FullPath**: Set to true (default: false) to create the intermediate directories and the final directory, from the path specified via the Dir parameter.
RESULT	Contains eventually error message.

Examples

```
LNCCMD      CMD(CREATEDIR) +
              PARM1('Dir="C:\temp\test\AURA\result"')
```

The final directory "result" will be created if "C:\temp\test\AURA" already exists.

```
LNCCMD CMD(CREATEDIR) PARM1('Dir="C:\A\DV\KAT";FullPath=true')
```

The "DV" and "KAT" directories will be created with FullPath=true, because "C:\A" already exists.

CRF command

The command CRF creates an empty file on PC.

Do not forget to finalize the creation of the file with the ENF command.

This file can then be used as a data file on the PC.

The file can be filled programmatically with the NCL, NLN and VAL commands.

Or we can use the DBFXFER command for the transfer of a database file.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('CRF')
CHGVAR          VAR(&PARM1) VALUE('File')
CHGVAR          VAR(&PARM2) VALUE(' ')

CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Complete path of the file to create. The file is created empty. If the file already exists, it will be overwritten.

Example 1

```
LNCOPEN

LNCCMD      CMD(CRF) PARM1('C:\A\fusion2.txt')

LNCCMD      CMD(NCL) PARM1('Lastname')

LNCCMD      CMD(NCL) PARM1('Firstname')

LNCCMD      CMD(NCL) PARM1('Id')

LNCCMD      CMD(NLN)

LNCCMD      CMD(VAL) PARM1('Lastname') PARM2('Redfall')

LNCCMD      CMD(VAL) PARM1('Firstname') PARM2('Robert')

LNCCMD      CMD(VAL) PARM1('Id') PARM2('1')
```

```
LNCCMD      CMD(ENF)

LNCCLOSE
```

In this example, the file 'fusion2.txt' is created in the directory "C: \ A".

Example 2

```
LNCOPEN

LNCCMD      CMD(CRF) PARM1('C:\A\fusion.txt')

LNCCMD      CMD(ENF)

LNCCMD      CMD(DBFXFER) PARM1('FILE="C:\A\fusion.txt"') +

              PARM2('SQL:select CUST_ID as Identifiant, +

              FIRSTNAME as Prenom, LASTNAME as Nom from +

              SP_CUST')

LNCCLOSE
```

See also

- ENF
- NCL
- NLN
- VAL
- DBFXFER

CRITSECT command

Allows to solve PC resources access conflicts, if LAUNCHER Office server is configured to accept several concurrent connections.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('CRITSECT')
CHGVAR          VAR(&PARM1) VALUE('Name=NAME" [;Enter] | [;Leave]')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Name** : Symbolic resource name. **Enter** : Points out an application critical phase beginning. It will end with a CRITSECT command with the Leave option, on the same resource. **Leave** : Points out an application critical phase ending. Another application can enter into it.
Parm2	

Refer to Multiple connection mode.

When an application must use a shared PC resource, clipboard or a job file, it must enter to a critical phase and leave it when the resource use is completed.

Application will stand-by if another application is already in the same critical phase until it is released.

Resource name is a virtual name sets by the application.

Example

```
CHGVAR VAR(&CMD) VALUE('CRITSECT')
CHGVAR VAR(&PARM1) VALUE('Name="PhaseCrit";Enter')
CALL  PGM(LNCCMD) PARM(&HANDLE &OPT &PARM1 &PARM2 &RESULT)


/* Copy to clipboard */
CHGVAR VAR(&CMD) VALUE('WCOPY')
CALL  PGM(LNCCMD) PARM(&HANDLE &OPT &PARM1 &PARM2 &RESULT)


/* Paste to clipboard */
CHGVAR VAR(&CMD) VALUE('WPASTE')
CALL  PGM(LNCCMD) PARM(&HANDLE &OPT &PARM1 &PARM2 &RESULT)
```

```
/* Exit now */
CHGVAR VAR(&CMD) VALUE('CRITSECT')
CHGVAR VAR(&PARM1) VALUE('Name="PhaseCrit";Leave')
CALL  PGM(LNCCMD) PARM(&HANDLE &OPT &PARM1 &PARM2 &RESULT)
```

CSVTOXML command

Allows you to generate an XML file from a CSV file (separator: semicolon).

Syntax

```
CHGVAR              VAR(&CMD) VALUE('CSVTOXML')
CHGVAR              VAR(&PARM1) VALUE('
                    CSV="Path of the CSV file";
                    XML="Path of the XML file";
                    ElemName="Tag name"')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**CSV**: Contains the full path of the CSV file (semicolon separator). **XML**: Contains the full path of the XML file. **ElemName**: Indicates the name of each tag corresponding to each record of the CSV file.
Parm2	

Example

```
PGM

DCL         VAR(&SVRADDR) TYPE(*CHAR) LEN(30) VALUE(*DEV)
DCL         VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL         VAR(&CCSID) TYPE(*CHAR) LEN(10)

DCL         VAR(&CMD) TYPE(*CHAR) LEN(10)
DCL         VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL         VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL         VAR(&PARM2) TYPE(*CHAR) LEN(1024)
DCL         VAR(&RESULT) TYPE(*CHAR) LEN(512)

CHGVAR      VAR(&HANDLE) VALUE('*ONLY')
CHGVAR      VAR(&CCSID) VALUE('*JOB')

CALL        PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)

CHGVAR      VAR(&CMD) VALUE('CSVTOXML')
```

```
CHGVAR     VAR(&PARM1) +
             VALUE('CSV="C:\A\csv.txt";XML="C:\A\csv_xml+
             .xml";ElemName="SP_CUST"')
CHGVAR     VAR(&PARM2) VALUE(' ')
CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
             &PARM2 &RESULT)

LNCCLOSE

ENDPGM
```

With the following CSV file:

```
"CUST_ID";"COMPANY";"FIRSTNAME";"LASTNAME";"CIVIL";"ADDRESS";"ADDR2";"CITY";"STATE";"ZIP";"COUNTRY";"PHONE";"FAX"
"1231";"Unisco";"George";"Weathers";"1";"PO Box Z-547";"";"Freeport";"";"";"Bahamas";"809-555-3915";"809-555-4958"
"1351";"Sight Diver";"Phyllis";"Spooner";"1";"1 Neptune Lane";"";"Kato Paphos";"";"";"Cyprus";"357-6-876708";"357-6-870943"
```

The generated XML file is the following :

```
<?xml version="1.0" standalone="yes"?>
<DocumentElement>
  <SP_CUST>
    <CUST_ID>"1231"</CUST_ID>
    <COMPANY>"Unisco"</COMPANY>
    <FIRSTNAME>"George"</FIRSTNAME>
    <LASTNAME>"Weathers"</LASTNAME>
    <CIVIL>"1"</CIVIL>
    <ADDRESS>"PO Box Z-547"</ADDRESS>
    <ADDR2>""</ADDR2>
    <CITY>"Freeport"</CITY>
    <STATE>""</STATE>
    <ZIP>""</ZIP>
    <COUNTRY>"Bahamas"</COUNTRY>
    <PHONE>"809-555-3915"</PHONE>
    <FAX>"809-555-4958"</FAX>
  </SP_CUST>
  <SP_CUST>
    <CUST_ID>"1351"</CUST_ID>
    <COMPANY>"Sight Diver"</COMPANY>
    <FIRSTNAME>"Phyllis"</FIRSTNAME>
    <LASTNAME>"Spooner"</LASTNAME>
    <CIVIL>"1"</CIVIL>
    <ADDRESS>"1 Neptune Lane"</ADDRESS>
    <ADDR2>""</ADDR2>
    <CITY>"Kato Paphos"</CITY>
    <STATE>""</STATE>
    <ZIP>""</ZIP>
    <COUNTRY>"Cyprus"</COUNTRY>
    <PHONE>"357-6-876708"</PHONE>
    <FAX>"357-6-870943"</FAX>
  </SP_CUST>
</DocumentElement>
```

CSVUTF8 command

Used to convert a text file from the DBFXFER command, to UTF-8 format.

By default, the data transfer mode is Unicode with Launcher (UTF-16LE).

Syntax

```
CHGVAR              VAR(&CMD) VALUE('CSVUTF8')
CHGVAR              VAR(&PARM1) VALUE('
                    CSVIn="Path of the text file in input"
                    ;CSVUTF8="Path of the result test file"
                    [;CodeIn="Encoding of the input text file"]')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**CSVIn**: Contains the full path of the text file from the DBFXFER command. **CSVUTF8**: Contains the full path of the text file converted to UTF-8. **CodeIn**: Optional. Can take the values "UTF-16LE" or "WINDOWS-1252". If this parameter is omitted, its default value is: "UTF-16LE". Indeed, the default data transfer mode is Unicode ("UTF-16LE") with Launcher.
Parm2	

Example

```
PGM

DCL         VAR(&SVRADDR) TYPE(*CHAR) LEN(30) VALUE(*DEV)
DCL         VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL         VAR(&CCSID) TYPE(*CHAR) LEN(10)

DCL         VAR(&CMD) TYPE(*CHAR) LEN(10)
DCL         VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL         VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL         VAR(&PARM2) TYPE(*CHAR) LEN(1024)
DCL         VAR(&RESULT) TYPE(*CHAR) LEN(512)

CHGVAR      VAR(&HANDLE) VALUE('*ONLY')
```

```
CHGVAR     VAR(&CCSID) VALUE('*JOB')

CALL       PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)

CHGVAR     VAR(&CMD) VALUE('CSVUTF8')
CHGVAR     VAR(&PARM1) +
             VALUE('CSVIn="C:\A\csv.txt";CSVUTF8="C:\A\csv_utf8+
             .txt"')
CHGVAR     VAR(&PARM2) VALUE(' ')
CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
             &PARM2 &RESULT)

LNCCLOSE

ENDPGM
```

DBFXFER command

Transfer a database file, SQL result set, or IFS document, from the iSeries to the PC.

The PC file resulting from a database file or SQL result set transfer will be in text format.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('DBFXFER')
CHGVAR          VAR(&PARM1) VALUE('
                File="Destination file on PC"
                [;UNICODE=True/False]
                [;Fixed=True/False]
                [;FixedNL=True/False]
                [;ColHdg=True/False]
                [;Delimiter="Delimiter"]
                [;DecimalPt="Decimal point"]
                [;QuoteText=True/False]
                [;DescFile=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE('
                MAXREC=Maximum number of records ;
                COLHDG=LNAME / COLHDG / SNAME ;
                <Database file name / IFS document / SQL statement>
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2+ &RESULT)
```

Parameters

Parameters	
Parm1	**Parm1 contains options that will be interpreted by LAUNCHER Office on the PC.** **File=**Full path to the resulting file to create on the PC. If the keyword "FILE=" is missing, then, the whole parameter *Parm1* contains the path to the file. **UNICODE**=True / False (*Default=False*). By default, the data are stored in 8-bit characters set in the destination file. With UNICODE format a 16-bit characters set is used. **Fixed**=True / False (*Default =False*). By default, the file will be in CSV format. Columns are separated by a semicolon (;). If **Fixed** is true, the columns will have a fixed size, with

	no column or line separator. **FixedNL** Like for **Fixed**, columns will have a fixed size, but the lines will be separated by a "new line" character. **ColHdg**=True / False (Default =True). By default, the first line in the destination file will contain the columns headings or names. If **ColHdg** is false, the data will start from the first line. **Delimiter** : The default column delimiter is the semicolon (;). This delimiter can be changed here. Example***: Delimiter=","*** The delimiter can be totally deleted by: **Delimiter=""** If the file has only one field, for compatibility with MS Office, a delimiter is added at the end of each line. **DecimalPt** : The default decimal point will be the one defined in the Windows environment. It can be changed here. Example***: DecimalPt=","*** **QuoteText**=True / False (*Default =True*). The column values are between two double quote characters. Set **QuoteText** to false to remove those double quotes. **DescFile**=True / False (*Default =True*). A description file (*.LFD) is generated during the file transfer. This decription file is used by LAUNCHER Office when exporting te data to Excel. To spare resources, set **DescFile** To False.
Parm2	**Parm2 contains options interpreted by LAUNCHER Office on the AS/400.** These options must be at the beginning of Parm2, separated by semicolons (;). The name of the AS/400 data source is at the end of Parm2. **MAXREC**=*Number*; With this option, you can set the maximum number of record to transfer. By default, there is no limit. **COLHDG=** *LNAME / COLHDG / SNAME* The column heading on the PC file will consist of : If COLHDG=LNAME : the long field names or Alias. If COLHDG=COLHDG : the column headings. If COLHDG=SNAME : the short field names (*Default*).

	Database file name : Give here a database file name, with eventually its library and member name. The syntax for a qualified name is : LIBRARY/FILE(MEMBER) **IFS Document** : Enter here the full path to the file in the IFS, starting with keyword « **DOCUMENT:**» Example *: 'DOCUMENT:/qdls/docfolder/info.pdf'* **SQL Result set** : The keyword « **SQL:**» is followed by the SQL statement. Example : *'SQL:select * from customer where dept="75"'*. To qualify a file path, separate the library name from the file name with a point character, as defined by SQL syntax. Example : *'SQL:select * from mylib.myfile'*
RESULT	On return, variable &RESULT contains the number of transferred records (9 digits).

This transfer function doesn't require any other software than LAUNCHER installed on the PC.

To transfer a document from the IFS to the PC, you don't need to connect a virtual drive to the IFS.

Note : .lfd file is also created.

Example

```
CHGVAR        VAR(&CMD) VALUE('DBFXFER')
CHGVAR        VAR(&PARM1) VALUE('file="%LNCDIR%\fusion.txt ')
CHGVAR        VAR(&PARM2) VALUE('COLHDG=LNAME;LAUNCHER/SP_CUST')
CHGVAR        VAR(&CMD) VALUE('DBFXFER')
CHGVAR        VAR(&PARM1) VALUE('%TEMP%\readme.txt')
CHGVAR        VAR(&PARM2) VALUE('DOCUMENT:/home/qsecofr/vim-
              6.3/readme.txt')
CHGVAR        VAR(&CMD) VALUE('DBFXFER')
CHGVAR        VAR(&PARM1) VALUE('%TEMP%\trf.txt ')
CHGVAR        VAR(&PARM2) VALUE('MAXREC=100;SQL:select * from customer')
```

DBXPROP command

Controls options of transfer of AS/400 file on PC.
It is mandatory to use it before using a DBFXFER command.

Default **DBFXFER command** creates an ASCII file with the following properties :

- field separator : semicolon ";"
- record separator (line break)
- all fields in double quotes, column headers on first line

Syntax

```
CHGVAR              VAR(&CMD)VALUE('DBXPROP')
CHGVAR              VAR(&PARM1)VALUE('Property')
CHGVAR              VAR(&PARM2)VALUE('Value')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

Parameters

Property (PARM1)	Value (PARM2)	Default	Description
FLDSEP	character or string	;	Defines fields separator
COLHDG	OFF SNAME LNAME COLHDG	SNAME	**OFF =** no column headers. **SNAME =** short zone name used as column name. **LNAME =** long zone name, or alias, used as column name. **COLHDG =** zone "Column Heading" is used.
NUMPREFSUF	ON/OFF	OFF	No double quotes for digital values.
FLDPREF	character or string	"	Field beginning character.
FLDSUF	character or string	"	Field ending character.
DESCFILE	ON/OFF	ON	Generates a description file (*.lfd).
UNICODE	ON/OFF	OFF	Creates an UNICODE format text file.
DBLQUOTE	ON/OFF	ON	**ON :** Double quotes may be doubled in zone values. **OFF :** no doubling.

Example

```
CHGVAR                VAR(&CMD) VALUE('DBXPROP')
CHGVAR                VAR(&PARM1) VALUE('DESCFILE')
CHGVAR                VAR(&PARM2) VALUE('OFF')
```

The next DBFXFER will only create the data text file and not the description file (*.lfd).

Other example

Disable Unicode transfer:

```
  LNCCMD      CMD(DBXPROP) PARM1('UNICODE') PARM2('OFF')
```

See also

- DBFXFER

DELETEFILE command

Deletes a file from a directory on a PC.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('DELETEFILE')
CHGVAR              VAR(&PARM1) VALUE('
                    File="File to delete"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File** : Complete path and name of the file to delete.
RESULT	Contains eventually error message.

Example

```
LNCCMD CMD(DELETEFILE) PARM1('File="C:\temp\res.pdf"')
```

DIRLIST command

Gets the contain of a Windows directory.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('DIRLIST')
CHGVAR          VAR(&PARM1) VALUE('
                [;Pattern="Files to list"]
                [;Path="Path to directory"]
                [;First= True / False ]
                [;Close= True / False ]
                [;SubDir= True / False ]
                [;Hidden= True / False ]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Pattern=** : Allows to set the type of files we want to list. Example : **Pattern="*.doc"** Keyword « Pattern » can also include the directory path. Example : **Pattern ="\\FileServer\Docs*.doc"** **Path=** : The directory path, if it is not included in **Pattern**. Example : **Path="\\FileServer\Docs"; Pattern="*.doc"** **First=True** to start listing at the beginning of the directory. If **First** is false, command returns the next entry in the directory. **Close=True** to close the directory listing. **SubDir=True** to list only the sub directories contained into the specified directory. The files are not listed. **Hidden=True** to list also the hidden files.
RESULT	On each call, a file name is returned into the **&RESULT** parameter. If the list is empty, or the last file listed, &RESULT is empty. Each file returned in &RESULT has the following format: • Positions 1 to 256 : File name. • Position 257 : **'N'**. Letter 'N' is present if the entry is not empty.

- Position 258 : **'A'.** If the file has 'Archive' attribute.
- Position 259 : **'H'**. If the file name is hidden.
- Position 260 : **'R'**. If the file is read only.
- Position 261 : **'D'**. If the entry is a sub directory.
- Position 262 : **'S'**. If it is a system file.

Example

```
CHGVAR          VAR(&CMD) VALUE('DIRLIST')
CHGVAR          VAR(&PARM1) VALUE('Pattern="*.xls";Path="N:\DOCS"')
CHGVAR          VAR(&PARM2) VALUE('First=True')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

END command

Allows to disconnect (end commands session).

Word and Excel remain active if they are visible.

Syntax

```
CHGVAR                VAR(&CMD) VALUE('END')
CHGVAR                VAR(&PARM1) VALUE(' ')
CHGVAR                VAR(&PARM2) VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                      &PARM1 &PARM2 &RESULT)
```

See also

- LNCOPEN
- LNCCLOSE

ENF command

Terminates the creation of the merge file.

The file will be saved under the name specified while calling the CRF command.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('ENF')
CHGVAR              VAR(&PARM1) VALUE(' ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

See also

- CRF
- NCL
- NLN
- VAL

ENUMPRT command

List the printer drivers installed on the PC.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('ENUMPRT')
CHGVAR          VAR(&PARM1) VALUE('
                  [ First=True / False ]
                  [; NameOnly=True / False ]
                  [; Close=True ; False ] ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**First** : **True**, to initialize the list process, and receive the description for the first printer. If **First** is false, &RESULT will contain the description of the next printer. **NameOnly** : If **NameOnly** is true, variable &RESULT will only contain the name of the printer driver. Else, &RESULT contains the description as follow : **Printer="Printer name";Server="Server name"** **Close** : True to end the list process.
RESULT	On output, parameter &RESULT contains : • *NONE if the last printer was listed. • The description of a printer. Format depends on the value of the **NameOnly** keyword.

Example

```
PGM
DCL         VAR(&RES) TYPE(*CHAR) LEN(2000)
DCL         VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL         VAR(&RES2) TYPE(*CHAR) LEN(2000)
DCL         VAR(&HDL) TYPE(*CHAR) LEN(100) VALUE('*ONLY ')
```

```
DCL         VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL         VAR(&PARM2) TYPE(*CHAR) LEN(1024)

LNCOPEN
CHGVAR      VAR(&PARM1) VALUE('FIRST=TRUE')
CALL        PGM(LNCCMD) PARM(&HDL 'ENUMPRT' &OPT &PARM1 +
              &PARM2 &RES)

CHGVAR      VAR(&PARM1) VALUE('FIRST=FALSE')

DOWHILE     COND(&RES *NE '*NONE')

CALL        PGM(LNCCMD) PARM(&HDL 'ENUMPRT' &OPT &PARM1 +
              &PARM2 &RES)
ENDDO


LNCCLOSE

ENDPGM
```

EXCELCLOSE command

Closes the Microsoft EXCEL application.

The standard EXCEL exit messages will be displayed if a user action is required (printing in progress as a background task , modified file not saved, etc).

Syntax

```
CHGVAR                VAR(&CMD)VALUE('EXCELCLOSE')
CHGVAR                VAR(&PARM1)VALUE(' ')
CHGVAR                VAR(&PARM2)VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                      &PARM1 &PARM2 &RESULT)
```

Example

```
LNCCMD      CMD(EXCELCLOSE)
```

See also

- EXCELOPEN

EXCELHIDE command

EXCELHIDE command hides used instance.
Document is hidden to the user and LAUNCHER Office operates faster.

Syntax

```
CHGVAR                VAR(&CMD)VALUE('EXCELHIDE')
CHGVAR                VAR(&PARM1)VALUE(' ')
CHGVAR                VAR(&PARM2)VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                      &PARM1 &PARM2 &RESULT)
```

Example

```
LNCCMD      CMD(EXCELHIDE)
```

See also

- EXCELSHOW

EXCELOPEN command

Opens Microsoft EXCEL.

If no settings are used, only the Microsoft Excel application is started.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('EXCELOPEN')
CHGVAR              VAR(&PARM1) VALUE(' ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Name of the file to open. It may be empty if a new document has to be created, or includes a template name, if **NEW is specified as parameter 2.**
Parm2	**Visible** = True/False. To show Excel. Default value is False **New** = True/False. It allows to create a new document with value = TRUE, if Parm1 is empty. Default value is False

Example

```
LNCCMD      CMD(EXCELOPEN)
```

See also

- EXCELCLOSE
- WORDOPEN

EXCELSHOW command

EXCELSHOW command shows the used instance. Therefore the user is able to see or to modify the current document.

Excel instance is hidden by default when opening Excel using EXCELOPEN command.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('EXCELSHOW')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Example

```
LNCCMD      CMD(EXCELSHOW)
```

See also

- EXCELHIDE

EXE command

Allows to run a PC application.

Option (0, 1 or 2) follows the command to tell demon what event to wait for, before sending message to the server.

- **Option 0** : demon waits until application job finishes to send message back to the server.
- **Option 1** : demon sends the message back to the server as soon as application is launched.
- **Option 2** : demon sends the message back to the server when an event is triggered by the application.

If you develop the application yourself, you may use LNCMsg library as well as the following functions :

LNCMessage

LNCRcvMessage

LNCSetPostMessage

Syntax

```
CHGVAR              VAR(&CMD) VALUE('EXE')
CHGVAR              VAR(&OPT) VALUE('[0|1|2]')
CHGVAR              VAR(&PARM1) VALUE('Application [Params]')
CHGVAR              VAR(&PARM2) VALUE('[Params]')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Application and its optional parameters command line.
Parm2	May contain parameters to transfer to application.

Example

```
CHGVAR              VAR(&CMD) VALUE('EXE')
CHGVAR              VAR(&OPT) VALUE('0')
CHGVAR              VAR(&PARM1) VALUE(' c:\path\application.exe ')
CHGVAR              VAR(&PARM2) VALUE(' Parameters ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

Other example

```
LNCCMD      CMD(EXE) PARM1('%TEMP%\scan.exe') +
                PARM2('"C:\temp\test2.pdf" /contraste=10 +
                /brillance=10 /resolution=300') OPT('2')
```

See also

- SHELL

FILECLOSE command

Closes the text file previously opened with FILEOPEN.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('FILECLOSE')
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Example

In this example, the « C:\temp\text.txt » file is write enabled and current content is deleted.
The program will, then, write a text into the file and close it.

```
CHGVAR          VAR(&CMD) VALUE('FILEOPEN')
CHGVAR          VAR(&PARM1) VALUE('File="C:\Temp\text.txt";Truncate=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)

CHGVAR          VAR(&CMD) VALUE('FILEWRITE')
CHGVAR          VAR(&PARM1) VALUE('Texte à écrire dans le fichier')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)

CHGVAR          VAR(&CMD) VALUE('FILECLOSE')
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Other example

```
LNCCMD      CMD(FILECLOSE)
```

See also

- Command FILEOPEN
- Command FILEWRITE

FILEOPEN command

Opens a text type file on a PC.

If the file does not exist it will be created.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('FILEOPEN')
CHGVAR          VAR(&PARM1) VALUE('File="name de File" [;UNICODE]
                [ ;Truncate] [ ;InsertNL]')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**File** : Complete path of the file. **UNICODE** : True / False (False= Default) Set "UNICODE=True" to generate an UNICODE format file (16 Bits characters). **Truncate** : True / False (False= Default) If *Truncate* is true, the file is deleted. **InsertNL** : True / False (False= Default) If *InsertNL* is true, each writing in the file using FILEWRITE command generates a break at line end.

Example

In this example, the « C:\temp\text.txt » file is opened for writing, while its current content is deleted. Program will, then, writes a text into the file.

```
CHGVAR          VAR(&CMD) VALUE('FILEOPEN')
CHGVAR          VAR(&PARM1) VALUE('File="C:\Temp\text.txt";Truncate=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)

CHGVAR          VAR(&CMD) VALUE('FILEWRITE')
CHGVAR          VAR(&PARM1) VALUE(&INPUT)
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
```

```
&PARM2 &RESULT)
```

See also

- Command FILEWRITE
- Command FILECLOSE

FILEREAD command

Read the content of a text file (ANSI or UTF8 without BOM) opened by the FILEOPEN command.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('FILEREAD')
CHGVAR        VAR(&PARM1) VALUE('StartBegin=true/false')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**StartBegin** : - True: start reading at the beginning of the file. - False : start reading at the beginning of the line following the line returned by the last FILEREAD command. Each FILEREAD command returns each line ended by CR LF (Carriage Return Line Feed). If %EOF% is returned by the FILEREAD command, it means that you reach the end of the file, or the file is empty.

Example

In this example, the « C:\temp\test.txt » file is opened by the FILEOPEN command, and each line is returned by the FILEREAD command (into &RES).

The content of &RES is displayed by the NOP command.

We know that we went through the entire file as soon as the FILEREAD command returns% EOF%.

```
PGM

DCL         VAR(&RES) TYPE(*CHAR) LEN(2000)
LNCOPEN

LNCCMD      CMD(FILEOPEN) +
              PARM1('File="c:\temp\test.txt"')

LNCCMDR     CMD(FILEREAD) PARM1('StartBegin=true') +
              RESULT(&RES)
LNCCMD      CMD(NOP) PARM1(&RES)

DOWHILE     COND(&RES *NE '%EOF%')
LNCCMDR     CMD(FILEREAD) PARM1('StartBegin=false') +
              RESULT(&RES)
```

```
LNCCMD      CMD(NOP)  PARM1(&RES)
ENDDO


LNCCMD      CMD(FILECLOSE)
LNCCLOSE
ENDPGM
```

See also

- Command FILEWRITE
- Command FILECLOSE

FILEWRITE command

Writes into file, previously opened with FILEOPEN.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('FILEWRITE')
CHGVAR          VAR(&PARM1) VALUE('Text to write in the file')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Text to write into the file. Text is added at file end.

Example

In this example « C:\temp\text.txt » file is opened for writing, while its current content is deleted.
Program will, then, write text into the file.

```
CHGVAR          VAR(&CMD) VALUE('FILEOPEN')
CHGVAR          VAR(&PARM1) VALUE('File="C:\Temp\text.txt";Truncate=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)

CHGVAR          VAR(&CMD) VALUE('FILEWRITE')
CHGVAR          VAR(&PARM1) VALUE('Text... ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Note

If the value to write contains a semicolon, you have to put the value between double quotes :

```
LNCCMD CMD(FILEWRITE) PARM1('"aaa;bbb;ccc;ddd"')
```

See also

- Command FILEOPEN
- Command FILECLOSE

FTPCLOSE command

Close a FTP session opened by the FTPCONNECT command.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPCLOSE')
CHGVAR              VAR(&PARM1) VALUE(' ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Note

Use this command only to close a FTP session opened previously by the FTPCONNECT command.

Example

```
LNCCMD CMD(FTPCLOSE)
```

FTPCONNECT command

Open a FTP session with the FTP server specified into parameters.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('FTPCONNECT')
CHGVAR          VAR(&PARM1) VALUE('
                ServerName="FTP server name"
                ;Username="FTP user name"
                ;Password="FTP user password"
                [;SSLEnable=true/false]
                [;ServerPort="FTP Port"]
                [;Logfile=true/false]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**ServerName** : FTP server name or IP address. **Username** = FTP user login. **Password**=FTP user password. **SSLEnable** : true/false. Optional. By default : false (not secure). If SSLEnable=true, the FTP Secure protocol (FTP over TLS/SSL) is used. **ServerPort** : FTP server port. Optional. If not specified, by default, the port 21 is used for non-secure connections, and the port 990 is used for secure connections. **Logfile** : true/false. Optional. By default : false. If Logfile=true, a logfile is created into %TEMP%.
RESULT	Eventually, contains error messages.

Example

```
LNCCMD CMD(FTPCONNECT)+ PARM1('ServerName="ftp.aura.com";+
username="aura";password="aura"')
```

FTPDELDIR command

Allows to delete a remote directory.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPDELDIR')
CHGVAR              VAR(&PARM1) VALUE('
                    RemoteDir="Remote directory to delete"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**RemoteDir** : Complete path from the FTP server root and name of the directory to delete.
RESULT	Contains eventually error messages.

Example

```
   LNCCMD      CMD(FTPDELDIR)
PARM1('RemoteDir="/www/Data/temp/Suivi5"')
```

FTPDELFILE command

Allows to delete a remote file.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPDELFILE')
CHGVAR              VAR(&PARM1) VALUE('
                    RemoteDir="Remote file to delete"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**RemoteFile** : Complete path from the FTP server root and name of the file to delete.
RESULT	Contains eventually error messages.

Example

```
LNCCMD CMD(FTPDELFILE)
PARM1('RemoteFile="www/Data/temp/lnctoxls_test5.xlsx"')
```

FTPGETDIR command

Allows to get all files of a remote directory.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPGETDIR')
CHGVAR              VAR(&PARM1) VALUE('
                    LocalDestDir="Local directory";
                    RemoteSrcDir="Remote directory"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**LocalDestDir** : Complete path of the local directory where the remote files will be copied. **RemoteSrcDir** : Complete path from the FTP server root and name of the directory containing the files to get.
RESULT	Contains eventually error messages.

Example

```
LNCCMD CMD(FTPGETDIR)
PARM1('RemoteSrcDir="/www/Data/temp/suivi";LocalDestDir="C:\A\suivi"')
```

FTPGETFILE command

Allows to get a remote file present on a FTP server.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPGETFILE')
CHGVAR              VAR(&PARM1) VALUE('
                    NewLocalFile="Local file got from FTP server";
                    RemoteFile="Remote file to get on FTP server"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**NewLocalFile** : complete path and name of the local file got from the FTP server. **RemoteFile** : complete path from the FTP server root and name of the remote file to get on the FTP server.
RESULT	Contains eventually error messages.

Example

```
LNCCMD CMD(FTPGETFILE)+
PARM1('NewLocalFile="C:\temp\gettest.xlsx";+
RemoteFile="/www/Data/temp/lnctoxls_test5.xlsx"')
```

FTPLISTDIR command

Allows to list all files present into a remote directory on a FTP server.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPLISTDIR')
CHGVAR              VAR(&PARM1) VALUE('
                    LocalFile="Local file";
                    RemoteDir="Remote directory"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**LocalFile** : complete path and name of the local file used to get the list of the files contained into the remote directory. **RemoteDir** : complete path from the FTP server root and name of the remote directory on the FTP server.
RESULT	Contains eventually error messages.

Example

```
LNCCMD      CMD(FTPLISTDIR)
PARM1('RemoteDir="/www/Data/temp/mkdir_test";+
LocalFile="c:\temp\mdir_test.csv"')
```

FTPMKDIR command

Allows to create a remote directory.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPMKDIR')
CHGVAR              VAR(&PARM1) VALUE('
                    RemoteDir="Remote directory to create"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**RemoteDir** : Complete path from the FTP server root (‘/’) and name of the directory to create.
RESULT	Contains eventually error messages.

Example

```
LNCCMD      CMD(FTPMKDIR) PARM1('RemoteDir="/www/Data/temp/Suivi5"')
```

FTPPUTDIR command

Allows to put a local directory on a FTP server.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPPUTDIR')
CHGVAR              VAR(&PARM1) VALUE('
                    LocalSrcDir="Local directory"
                    RemoteDstDir="Remote directory"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**LocalSrcDir** : Complete path and name of the local directory containing files to put on the FTP server. **RemoteDstDir** : Complete path from the FTP server root ('/') and name of the new directory containing files put on the FTP server.
RESULT	Contains eventually error messages.

Example

```
LNCCMD      CMD(FTPPUTDIR)
PARM1('LocalSrcDir="C:\temp\suivi5";RemoteDstDir="/www/Data/temp+
/suivi5R"')
```

FTPPUTFILE command

Allows to put a local file on a FTP server.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('FTPPUTFILE')
CHGVAR              VAR(&PARM1) VALUE('
                    LocalFile="Local file to put on the FTP server"
                    NewRemoteFile="New file put on the FTP server"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**LocalFile** : Complete path and name of the local file to put on the FTP server. **NewRemoteFile** : Complete path from the FTP server root ('/') and name of the new file put on the FTP server.
RESULT	Contains eventually error messages.

Example

```
LNCCMD CMD(FTPPUTFILE)
PARM1('LocalFile="C:\temp\lnctoxls_test5.xlsx";+
NewRemoteFile="/www/Data/temp/lnctoxls_test5.xlsx"')
```

GETREGE command

Return the value of a Registry key.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('GETREGE')
CHGVAR          VAR(&PARM1) VALUE('
                Register="Path of the Registry key";
                Value="Value"
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
```

Parameters

Parameters	
Parm1	**Register** : Path of the Registry key. **Value** : Value name.
RESULT	Returned registers basic value, or *REJ with error message.

Example

```
CHGVAR      VAR(&CMD) VALUE('GETREGE')
CHGVAR      VAR(&PARM1) +
              VALUE('Register="HKEY_LOCAL_MACHINE\SOFTWAR+
              E\Microsoft\Windows +
              NT\CurrentVersion";Value="RegisteredOwner"')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RES)
```

GETSYSINFO command

Returns information to PC Windows system.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('GETSYSINFO')
CHGVAR          VAR(&PARM1) VALUE('ComputerName' | 'UserName' | 'OSVersion
                | DefaultPrinter | LNCversion')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	This parameter can take one the following values : **ComputerName** : &RESULT variable returns name of PC on network. **UserName** : &RESULT variable returns name of current user on PC. **OSVersion** : &RESULT variable returns the Windows OS version. **DefaultPrinter** : &RESULT variable returns the name of the default printer. **LNCVersion**: &RESULT variable contains the version number of the Launcher server on Windows side.

Note

If **OSVersion** is used, therefore &RESULT will contain one of the following values:

```
OS                              &RESULT
----------------------------  --------------
Windows 10                      10.0
Windows Server 2019             10.0
Windows Server 2016             10.0
Windows 8.1                     6.3
Windows Server 2012 R2          6.3
Windows 8                       6.2
```

```
Windows Server 2012             6.2
Windows 7                       6.1
Windows Server 2008 R2          6.1
Windows Server 2008             6.0
Windows Vista                   6.0
Windows Server 2003 R2          5.2
Windows Server 2003             5.2
Windows XP 64-Bit Edition       5.2
Windows XP                      5.1
Windows 2000                    5.0
Windows ME                      4.9
Windows 98                      4.10
```

Example

In this example current user name returns in &RESULT variable.

```
CHGVAR          VAR(&CMD) VALUE('GETSYSINFO')
CHGVAR          VAR(&PARM1) VALUE('UserName')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

To know installed Windows version.

```
CHGVAR          VAR(&CMD) VALUE('GETSYSINFO')
CHGVAR          VAR(&PARM1) VALUE('OSVersion')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Returned &RESULT contains :

```
     WINDOWS   W8   00002-00006-00002-000023F0
```

GRAPHSEND command

This command is available for Office 365 users. OAuth authentication (OAuth2 protocol) is used to connect to the SMTP protocol and send emails.

In order to authenticate, you must provide the following elements:

- Microsoft Entra ID tenant ID: **Tenant** parameter

- Caller's application client ID: **ClientIdVal** parameter

- Application's secret value: **ClientSecretVal** parameter

Of course, this information will not appear in the Launcher trace.

For more information, here is a link to the Microsoft documentation.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('GRAPHSEND')
CHGVAR          VAR(&PARM1) VALUE('
                Server="SMTP server name"
                ;UserId="SMTP user – email sender"
                ;Password="password of the SMTP user"
                ;EmailTo="Recipients TO"
                ;ClientIdVal="client ID"
                ;ClientSecretVal="client secret"
                ;Tenant="tenant"
                ;Subject="subject"
                [;Text="Text of the email body"]
                [;Html="complete path of the HTML file, email body"]
                [;Attach="attachments list"]
                [;EmailCc="Recipients CC"]
                [;EmailBcc="Recipients BCC"]
                [;Priority=0|1|2]
                [;Importance=0|1|2]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Server** : SMTP server name. Typically: outlook.office365.com.
Parm2	**UserId** : SMTP account used for OAuth authentication and for sending email (FROM). **Password** : UserId password. **EmailTo** : List of TO recipients. Each email address is separated by

a semicolon.

ClientIdVal : Client ID. Useful for authentication.

ClientSecretVal : Client Secret. Useful for authentication.

Tenant : Tenant. Useful for authentication.

Subject : Subject of the email.

Text : Plain text of the email.

Html : Instead of using the Text parameter, you can specify here the full path to an HTML file that will correspond to the body of the email.

Attach : List of attachments. You must specify the full path for each attachment. Each path will be separated by a semicolon.

EmailCc : List of CC recipients. Each email address is separated by a semicolon.

EmailBcc : List of BCC recipients. Each email address is separated by a semicolon.

Priority : Email priority.

0 = not urgent
1 = normal (by default)
2 = urgent

Importance : Importance of email.

0 = low
1 = normal (by default)
2 = high

Example

The following example sends an email with an HTML file in the body of the email, with attachments.

```
LNCCMD      CMD(GRAPHSEND) +
              PARM1('Server="outlook.office365.com";UserId+
              ="test@aura.com";Password="aura"+
              ;emailTo="contact@aura.com;+
              tech@aura.com";clientIdVal="ddfgthyp-4444-5555-a+
```

```
fgt-asdftp74569b";clientSecretVal="TeV7U~rG+
5555.Y1hPmL9-HG7sC_OPMgbls8156zT";Tenant="4+
745etg9-sese-55e0-5479-5639874df156";Subjec+
t="Test GRAPH";Html="C:\A\ista4.HTML";+
Attach="C:\A\res.pdf;C:\A\F84.pdf;C:\A\image001.jp+
g;c:\A\image002.png"') +
PARM2('Priority=2;Importance=2')
```

IFSPUT command

Transfer a file from the PC to the AS/400 into a library or into the integrated file system (IFS).

Syntax

```
CHGVAR          VAR(&CMD) VALUE('IFSPUT')
CHGVAR          VAR(&PARM1) VALUE('Path to the source file on PC')
CHGVAR          VAR(&PARM2) VALUE('
                <Folder and file name on AS/400>
                [; Mode=*ALL ]
                [; Append ]
                [; Text ]
                [; RecIO[=nombre] ]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Full path of the PC source file, to be transfered to AS/400.
Parm2	Full path to the destination file on the AS/400. **If the destination file is in the IFS**, the path is given from the root. (/documents/prod_docs/doc01.pdf). Use the system command **WRKLNK** to know the folder hierarchy. **If the destination is in shared folders**, the path must begin with "/ QDLS /". Be careful, the IFS and "Shared Folders" do not represent the same thing. The shared folders are in a subdirectory of the IFS: **/QDLS**. **If the destination file is into an AS400 library**, the path to the file can be set as follow : /QSYS.LIB/MYLIB.LIB/MYFILE.FILE Or it can also be set as follow : MYLIB/MYFILE If option **RecIO** is set. The path to the destination file can be followed by options, separated by semicolons. The options are separated from the file path by a semicolon. **Mode=*ALL** : Give the right for Read and Write to all users.

Append : The data will be append to the file. By default, the destination file is cleared before the transfer.
If the destination file doesn't exist, it will be created.

Text : By default, the file is transfered as it is, with no conversion. The option **Text** will do a conversion from Text to text.

RecIO=Number. This options says that the file name has the form "LIBRARY/FILE(MEMBER) ". If a value is given with the **RecIO** option, it is the size of file records if a new file has to be created.
If the **RecIO** word alone is fixed, with no assigned number, then the file must exist.

Example

In the following example a document is printed in PRN File. This file is sent to IFS, and latter transferred to a spool file.

```
LNCPRTDOC     DOC('C:\Models\layout.doc')
              MRGTYPE(*FILE)
              FROMFILE(SP_LAYOUT)
              OUTPUT(*OUTPRN)
              OUTPRN('C:\temp\layout.prn')
              ENDOPT(*APP)

CHGVAR        VAR(&CMD) VALUE('IFSPUT')
CHGVAR        VAR(&PARM1) VALUE('C:\Temp\layout.prn')
CHGVAR        VAR(&PARM2) VALUE('/QDLS/IMAGES/Layout.prn')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)


CRTPRNSPLF    FILE('/QDLS/IMAGES/Layout.prn')
              SPLFNAME(PRNSPLF)
              USRTXT('Document Word')
RMVLNK        OBJLNK('/QDLS/IMAGES/Layout.prn')
CALL          PGM(LNCCLOSE)
```

See also

- LCNPRTDOC - CL Command
- CRTPRNSPLF - CL Command
- DBFXFER

LISTSCAN command

Allows to list the different scanners that will be able to be used by the SCAN command.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('LISTSCAN')
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
RESULT	The RESULT parameter will contain the scanners list, separated by semicolon. At the beginning, RESULT will still contain : "*MSG;".

Example

```
PGM
DCL         VAR(&RESULT) TYPE(*CHAR) LEN(2000)
LNCOPEN

LNCCMDR     CMD(LISTSCAN) RESULT(&RESULT)
LNCCMD      CMD(NOP) PARM1(&RESULT)
LNCCLOSE
ENDPGM
```

For instance, &RESULT : *MSG;HP LJ100 M175 Scan

LNCDBGET command

The command **LNCDBGET** allows to retrieve data from database : Oracle, MySQL, PostgreSQL, Sybase, DB2 for iSeries or Microsoft SQL Server.

The "JDBC" license extension is mandatory.

Paramètres

The command LNCDBGET requires the following parameters :

Database port number	**NbPort**
Database type	**Type**
Database server address	**SvrAddr**
Database user name	**User**
Database user password	**Password**
Database name	**DBName**
SQL query	**Query**
Complete path of CSV result file	**CSVFile**

Prerequisites

Installation of the JDBC JARs on PC/Server hosting Launcher server:

In order to access the Oracle, MySQL, PostgreSQL , Sybase, DB2 for iSeries or Microsoft SQL Server database, it is imperative that the corresponding JDBC JAR is present in the installation directory of the Launcher server.

For instance:

C:\Program Files (x86)\LAUNCHER400\LAUNCHER_JDBC\LNCClientWrapperJDBC_lib.

List of official JDBC JARs mandatory (no guarantee of operation if other JDBC JAR is used):

- For Oracle databases, you must use the Thin driver provided by Oracle. For instance, for Oracle Database 10g Release 2, the JDBC Thin driver is named **ojdbc14.jar**, and can be downloaded using the following link: http://www.oracle.com/technetwork/database/enterprise-edition/jdbc-10201-088211.html
- For MySQL databases, the JDBC driver **mysql-connector-java-5.1.21-bin.jar** has to be used : http://www.mysql.com/downloads/connector/j/

- For PostgreSQL databases (≥ version 7.2), the JDBC driver **postgresql-9.2-1000.jdbc4.jar** can be downloaded from : http://jdbc.postgresql.org/download.html
- For Sybase databases (all Sybase products using JConnect-7_0: Adaptive Server Enterprise, SQL Anywhere, Sybase IQ, and Replication Server), the JDBC driver **jconn4.jar** has to be used : http://www.sybase.fr/products/allproductsa-z/softwaredeveloperkit/jconnect
- For Microsoft SQL Server (SQL Server2012, SQL Server 2008 R2, SQL Server 2008, SQL Server 2005, and SQL Azure), the JDBC driver **sqljdbc4.jar** has to be used : http://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=11774
- For databases IBM DB2 for iSeries, you must use the driver **jt400.jar** supplied with IBM System i Access for Windows. After installing IBM Access on your PC, you can find the JDBC driver, for example here: C:\Program Files (x86)\IBM\Client Access\jt400\lib\jt400.jar

After having installed the JDBC JARs, it is mandatory to relaunch Launcher server.

Example

In this example, the database has the following characteristics :

- Type DB : **PostgreSQL**
- Database server address : 192.168.1.7
- Port number of the database server : 6078
- Database user : postgres
- Database user password : aura
- Database name : newold
- SQL query : select * from marci where title < 'b'

The CSV file containing data from database is still located here : %TEMP%\CSV_result.csv

The **LNCTOXLS** command is used to generate the following Excel document : c:\temp\result.xls

```
PGM

LNCOPEN
LNCCMD      CMD(LNCDBGET) +
              PARM1('Type=postgresql;SvrAddr=192.168.1.7+
              0;NbPort=:6078;User=postgres;Password=aura+
              ;DBName=newold;Query=select * from +
              marci where title < +
              ''b'')+
              CSVFile=C:\Users\crobini\AppData\Local\Temp\+
```

```
              CSV_result.csv')

LNCCMD      CMD(LNCTOXLS) +
              PARM1('ToXls=*new;ToSheet=*NONE;ToName=*NON+
              E;ColPref=*NONE;DataSource=C:\Users\crobini+
              \AppData\Local\Temp\CSV_result.csv;RecCnt=32;A+
              utoFmt=*NONE;AutoFit=true;FmtCells=false;Ad+
              dColH=false;ShowDoc=true;SavDoc=C:\temp\res+
              ult.xls;SavFmt=*XLS')

  LNCCLOSE
  ENDPGM
```

MAILATT command

SMTP, MAPI and Lotus Notes compatible.

Adds an attached file to the currently created message.

Command MAILPREP command must be called prior to create message.
Two different types of syntax exist according to the current interface.

Syntax

1) First Syntax (SMTP, MAPI and Lotus Notes)

```
CHGVAR                  VAR(&CMD) VALUE('MAILATT')
CHGVAR                  VAR(&PARM1) VALUE('File')
CHGVAR                  VAR(&PARM2) VALUE(' ')
CALL                    PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                        &PARM2 &RESULT)
```

Parameters	
Parm1	Full path of the file to be attached.

2) Second Syntax (SMTP)

This command can also be used to embed images in a message in HTML format.

The file must be added as message body, using HTML option with the **MAILBODYF** command.

The tag inserting the image in the HTML page must have the 'src' property with a value of type "src = cid: IMAGE001.GIF":

<img src=cid:IMAGE001.GIF>

The second parameter of the MAILATT command must have a value as `HEADER="CONTENT-ID:<IMAGE001.GIF>`.

NOTICE : Not all e-mail readers support this function !

```
CHGVAR                  VAR(&CMD) VALUE('MAILATT')
CHGVAR                  VAR(&PARM1) VALUE('File')
CHGVAR                  VAR(&PARM2)
                        VALUE('HEADER="CONTENT-ID:<IMAGE001.GIF>"')
CALL                    PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                         &PARM2 &RESULT)
```

Parameters	
Parm1	Full path of the file to be attached.
Parm2	Optional. It allows to embed images corresponding to a message in HTML format. Syntax is : HEADER="CONTENT-ID:<identifiant>"

Examples

```
LNCCMD      CMD(MAILATT) +
               PARM1('\\ADX01\SUPPORT\TEST\LOGOA2.JP+
               G') +
               PARM2('HEADER="CONTENT-ID:<IMAGE001.GIF>"')


LNCCMD      CMD(MAILATT) +
               PARM1('\\ADX01\SUPPORT\TEST\IMAGE001.+
               GIF')
```

See also

- LNCSNDMAIL - CL Command
- MAILPREP
- MAILTEXT
- MAILTO
- MAILPRTY
- MAILSUBJ
- MAILREPORT
- MAILSEND
- MAILEND
- MAILBODYF
- MAILCC
- MAILSMTP

MAILBODYF command

SMTP compatible

Sends a HTML or Text file as message body.

Command MAILPREP command must be called prior to create message.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILBODYF')
CHGVAR          VAR(&PARM1) VALUE(' \\MACHINE_A\SAMPLES\test.html')
CHGVAR          VAR(&PARM2) VALUE('HTML|TXT')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Full path of the HTML file to insert.
Parm2	Name of file format : HTML or TXT.

Example

```
LNCCMD      CMD(MAILBODYF) +
               PARM1('\\ADX01\SUPPORT\XYLEM\TEST\TEST2.HTM+
               L') PARM2('HTML')
```

See also

- LNCSNDMAIL - CL Command
- MAILPREP
- MAILATT
- MAILSUBJ
- MAILSEND
- MAILEND
- MAILSMTP

MAILCC command

SMTP Compatible.

Secondary recipients, hidden or showed, setup.

Command MAILPREP command must be called prior to create message.
If there is more than one secondary recipients, hidden or shown, they must be separated with semicolon.

All address formats can be used and combined (tech<tech@easycom-aura.com> or tech@easycom-aura.com).

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILCC')
CHGVAR          VAR(&PARM1) VALUE('tech<tech@easycom-aura.com>')
CHGVAR          VAR(&PARM2) VALUE('sales@Easycom-aura.com;info@Easycom-
                aura.com')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Shown secondary recipients list with **semicolon separators**.
Parm2	Hidden secondary recipients list with **semicolon separators**.

See also

- LNCSNDMAIL – CL Command
- MAILPREP
- MAILATT
- MAILTEXT
- MAILSUBJ
- MAILSEND
- MAILEND
- MAILBODYF
- MAILSMTP

MAILEND command

SMTP, MAPI and Lotus Notes Compatible

Frees all resources allocated to create the e-mail.

You must have called the MAILPREP command previously.

Syntax

```
CHGVAR                VAR(&CMD) VALUE('MAILEND')
CHGVAR                VAR(&PARM1) VALUE(' ')
CHGVAR                VAR(&PARM2) VALUE(' ')
CALL                  PGM(LNCCMD) +
                      PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)
```

Example

```
     LNCCMD     CMD(MAILEND)
```

See also

- LNCSNDMAIL – CL Command
- MAILPREP
- MAILATT
- MAILTEXT
- MAILTO
- MAILPRTY
- MAILSUBJ
- MAILREPORT
- MAILSEND
- MAILBODYF
- MAILCC
- MAILSMTP

MAILLATT command

Compatible Lotus Notes.

List the attachments of a Lotus message.
The message was listed by a previous call to MAILLMSG.

On return, &RESULT contain FALSE if there is no more attachments in the message.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILLATT')
CHGVAR          VAR(&PARM1) VALUE('
                [Start=True/False;]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Start** : If true, the description of the first attachment is returned in &RESULT.
RESULT	After the call, parameter &RESULT contains : **FALSE** if the last attachment was listed, **Type**="file type"; **Name**="original File name" ; **Path**="original path to the file" ;

Example

```
CHGVAR VAR(&CMD) VALUE('MAILLATT')
CHGVAR VAR(&PARM1) VALUE('Start=True')
CHGVAR VAR(&RESULT) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)

IF COND(&RESULT *EQ 'FALSE') THEN(GOTO CMDLBL(END))
```

```
CHGVAR VAR(&NAME) VALUE(' ')
CALL PGM(LNCGETKW) PARM(&HANDLE &RES 'Name ' &NAME)
```

MAILLMSG command

Compatible Lotus Notes.

List the messages in the inbox.

This command returns in the &RESULT parameter, the requested information on the first message or the next message in a Lotus folder.

If the end of the message list is reached, the command returns FALSE in &RESULT.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILLMSG')
CHGVAR          VAR(&PARM1) VALUE('
                Folder=Folder to list";
                Start=True/False;
                Subject=True/False;
                SenderName=True/False;
                UnReadOnly=True/false;
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Folder** : Name of the folder we want to get the message list. If "Folder" is not set, the inbox will be listed. **Start** : If "Start" is true , the first message of the folder is returned. If false, the next message is returned. **Subject** : If "Subject" is true, the subject of the message is returned in &RESULT, in the form : ***Subject="message subject" ;*** **SenderName** : If true, the sender name is returned in &RESULT, in the form : ***SenderName="Nom de l'expéditeur" ;*** **UnReadOnly** : If true, only the unread messages are listed.
RESULT	After the call, &RESULT contains : **FALSE** if the end of the list was reached. **Subject="message subject";** if the option "Subject=True" was

	set. **SenderName="Sender name",** if the option "SenderName=True" was set. Use program LNCGETKW to parse the contain of &RESULT.

Example

```
CHGVAR VAR(&CMD) VALUE('MAILLMSG')

CHGVAR VAR(&PARM1) VALUE('folder="domino/aura!!mail/box.nsf"; +
UnReadOnly=false')

CHGVAR VAR(&PARM2) VALUE('Subject=True;SenderName=True')

CHGVAR VAR(&RESULT) VALUE('*')

CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PM1 &PARM2 &RESULT)

IF COND(&RESULT *EQ 'FALSE') THEN(GOTO CMDLBL(END))

CHGVAR VAR(&SUBJ) VALUE(' ')
CALL PGM(LNCGETKW) PARM(&HANDLE &RESULT 'Subject ' &SUBJ)
```

See also

- LNCSNDMAIL - CL Command
- LNCGETKW
- MAILLATT
- MAILLMSG
- MAILMCHG
- MAILMDLT
- MAIXATT

MAILMCHG command

Compatible Lotus Notes

Change the properties of a message in a Lotus folder.
The message was listed by a previous call to MAILLMSG.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILMCHG')
CHGVAR          VAR(&PARM1) VALUE('
                [AsRead=True/False]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**AsRead** : If AsRead is true, the message is changed to "Read" state. **The message cannot be changed to "unread" state.**

See also

- LNCSNDMAIL - CL Command
- MAILLATT
- MAILLMSG
- MAILMCHG
- MAILMDLT
- MAIXATT

MAILMDLT command

Compatible Lotus Notes

Delete a message from a Lotus folder.
The message was listed by a previous call to MAILLMSG.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILMDLT')
CHGVAR          VAR(&PARM1) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

See also

- LNCSNDMAIL - CL Command
- MAILLATT
- MAILLMSG
- MAILMCHG
- MAILMDLT
- MAIXATT

MAILPREP command

SMTP, MAPI and Lotus Notes compatible

Creates a message to be sent through the messaging system.

This command must be called prior to message creation.

For this function, two different types of syntax exist according to the messaging interface.

Syntax

1) First Syntax (MAPI and Lotus Notes).

```
CHGVAR                  VAR(&CMD) VALUE('MAILPREP')
CHGVAR                  VAR(&PARM1) VALUE('Recipients')
CHGVAR                  VAR(&PARM2) VALUE('Object')
CALL                    PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                        &PARM2 &RESULT)
```

Parameters	
Parm1	Recipients list separated with **semicolons.**
Parm2	Message object.

2) Second Syntax (SMTP)

NOTICE : With SMTP, MAILSMTP Command must be run prior to MAILPREP Command.

```
CHGVAR          VAR(&CMD) VALUE('MAILPREP')
CHGVAR          VAR(&PARM1) VALUE('Recipients')
CHGVAR          VAR(&PARM2) VALUE('
                [subject="subject";]
                [priority=0 to 5;]
                [charset=encoding type;]
                [report]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parm1	Recipients list separated with **semicolons**.
Parm2	Characters chain including various properties*. **All properties are optional.**

*Properties	
subject	Message subject.
priority	Message priority level (from 0 to 5, less important to most important).
charset	Sets characters encoding type used. Possible values : ASCII, UTF-8, UTF-7, UTF-32, Unicode.
report	If this property is present into Parm2, therefore the delivery report is activated.

Example

```
LNCCMD CMD(MAILPREP) PARM1('tech@easycom-aura.com')
PARM2('subject="test";priority=5;report')
```

See also

- LNCSNDMAIL - CL Command
- MAILLATT
- MAILTEXT
- MAILTO
- MAILPRTY
- MAILSUBJ
- MAILREPORT
- MAILSEND
- MAILEND
- MAILBODYF
- MAILCC
- MAILSMTP

MAILPRTY command

MAPI and Lotus Notes compatible.

Sets created message priority.

Command MAILPREP command must be called prior to create message.

The different priority levels are:
HIGH: high priority,
NORMAL: normal priority (default),
LOW: low priority.

For SMTP, the priority of the message is fixed with the MAILPREP command.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILPRTY')
CHGVAR          VAR(&PARM1) VALUE('[*HIGH|*NORMAL|*LOW]')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The priority level you want to assign to the mail ***HIGH** : high priority ***NORMAL** : normal priority (default) ***LOW** : low priority

See also

- LNCSNDMAIL - CL Command
- MAILPREP
- MAILLATT
- MAILTEXT
- MAILTO
- MAILSUBJ
- MAILREPORT

- MAILSEND
- MAILEND

MAILREPORT command

MAPI and Lotus Notes Compatible

Activates the delivery report.

You must have called the MAILPREP command previously.

For **SMTP**, the delivery report is managed with the command MAILPREP.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILREPORT')
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) + PARM(&HANDLE &CMD &OPT &PARM1
                &PARM2 &RESULT)
```

See also

- LNCSNDMAIL - CL Command
- MAILPREP
- MAILLATT
- MAILTEXT
- MAILTO
- MAILPRTY
- MAILSUBJ
- MAILSEND
- MAILEND

MAILSEND command

SMTP, MAPI and Lotus Notes Compatible

Sends the previously created e-mail.

You must have called the MAILPREP command previously.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('MAILSEND')
CHGVAR        VAR(&PARM1) VALUE('
              [ Retry=Number of trials]
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Retry** : Number of attempts to send the email in case of failure.

See also

- LNCSNDMAIL - CL Command
- MAILPREP
- MAILLATT
- MAILTEXT
- MAILTO
- MAILPRTY
- MAILSUBJ
- MAILREPORT
- MAILEND
- MAILBODYF
- MAILCC
- MAILSMTP

MAILSMTP command

SMTP compatible.

Sets connection to SMTP server.

This command must be called first before sending any SMTP protocol messages.
The first parameter contains the different properties, separated by semicolon.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('MAILSMTP')
CHGVAR        VAR(&PARM1) VALUE('
              Server="SMTP server";
              Sender="email address of sender"
              [ ;UserId="SMTP user"]
              [ ;password=Password of the SMTP user]
              [ ;reply="email address receiving answers"]
              [ ;logfile="path of the logfile"]
              [ ;port= port number of the SMTP server ]
              [ ;enableSSL=true/false ]
              [ ;SSLType=SMTPS/TLS ]
              [ ;Sign=true/false ]
              [ ;Encrypt=true/false ]
              [ ;Certif="path of the certificate .p12"]
              [ ;PwdCertif=certificate password]
              [ ;Timeout=in seconds]
              [ ;SecureHeaders=true/false ]
              [ ;WrapperSubject="Sujet du wrapper RFC 822"]
              [ ;checkMail=true/false]
              [ ;useTestMode=true/false]
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	SMTP connection properties* list.

*Properties	
sender	Message sender address. Mandatory.
Server	SMTP server name or IP address. Mandatory.

UserId	SMTP user profile. Optional.
password	User password. Optional.
reply	Address of the person who receives the responses to the message sent, if different of the sender address. Optional.
logfile	Path and file name to store sended messages traces. It allows checking if message sending was performed without error. This file is formatted with separators (;), this allows it to be imported in database or Excel. Optional.
port	SMTP server port, if port 25 is not used. Optional.
EnableSLL	True/False (default). If true, By defauly, Launcher uses the TLS protocol. You can also use SMTPS (SMTP over SSL) by using the **SSLType** parameter
SSLType	Two possible values: **TLS** (default) and **SMTPS**. **TLS** : Launcher supports Secure SMTP on TLS as defined in RFC 3207. In this mode, the SMTP session starts on an unencrypted channel, and then a STARTTLS command is issued by the client to the server to switch to secure communication via SSL. See RFC 3207 published by the Internet Engineering Task Force (IETF) for more information. Generally port **587** is used. **SMTPS**: Also called SMTP/SSL or SMTP over SSL. Port **465** is generally used. Communication is secured via SSL from the start of the connection.
Sign	True/False (default). If true, Launcher signs the email sent, as defined in RFC 8551. It uses the certificate (.p12) specified with the **Certif** parameter. Complies with S/MIME Version 4.0 and RFC 8551. See RFC 8551 published by the Internet Engineering Task Force (IETF) for more information.
Encrypt	True/False (default). If true, Launcher encrypts the email sent, as defined in RFC 8551. It uses the certificate (.p12) specified with the **Certif** parameter. Complies with S/MIME Version 4.0 and RFC 8551. See RFC 8551 published by the Internet Engineering Task Force (IETF) for more information

Certif	Full path of the certificate (.p12) used to encrypt and/or sign the email.
PwdCertif	Certificate (.p12) password used to encrypt and/or sign the email.
Timeout	Timeout for operation, in seconds.
SecureHeaders	Protection of headers through the use of a RFC 822 message. A message/rfc822 wrapper is used to apply S/MIME security services to headers. See RFC 8551 for more details. The email will be received as an attachment by the recipient.
WrapperSubject	If SecureHeaders=true, by default the recipient will receive an email without subject and an attachment containing the specified email. You can specify a subject here in order to prevent the email from going into spam.
CheckMail	True/False (default). If true, a verification of the recipients' email addresses will be performed according to the RFC 5322 standard, before sending. Verification will be effective for MAILPREP, MAILTO and MAILCC commands.
UseTestMode	True/False (default). If true, a test SMTP server, which uses a self-signed certificate, can be used. Not to be used in production.

Example

```
LNCCMD      CMD(MAILSMTP) +
PARM1('SERVER="SMTP.AURA.FR";SENDER="TECH+
@EASYCOM-AURA.COM";REPLY="TECH@EASYCOM-+
AURA.COM";logfile="c:\temp\logaurasmtp.txt"')
```

See also

- LNCSNDMAIL - CL Command

- MAILPREP
- MAILLATT
- MAILTEXT
- MAILSUBJ
- MAILSEND
- MAILEND
- MAILBODYF
- MAILCC

MAILSUBJ command

SMTP, MAPI and Lotus Notes Compatible

Defines the subject of the message.

You must have called the MAILPREP command previously.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILSUBJ')
CHGVAR          VAR(&PARM1) VALUE('Subject')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) + PARM(&HANDLE &CMD &OPT &PARM1
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Subject of the message

See also

- LNCSNDMAIL - CL Command
- MAILPREP
- MAILLATT
- MAILTEXT
- MAILTO
- MAILPRTY
- MAILREPORT
- MAILSEND
- MAILEND
- MAILBODYF
- MAILCC
- MAILSMTP

MAILTEXT command

Compatible SMTP, MAPI and Lotus Notes.

Adds text to the body of the e-mail.

You must have called the MAILPREP command previously.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAILTEXT')
CHGVAR          VAR(&PARM1) VALUE('Text')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) + PARM(&HANDLE &CMD &OPT &PARM1
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Body text of the e-mail

See also

- LNCSNDMAIL - CL Command
- MAILPREP
- MAILLATT
- MAILTEXT
- MAILTO
- MAILPRTY
- MAILSUBJ
- MAILREPORT
- MAILSEND
- MAILEND
- MAILBODYF
- MAILCC
- MAILSMTP

MAILTO command

Adds one or several recipients to the e-mail.

You can specify what kind of role the recipients have (TO, CC, BCC).
You must have called the MAILPREP command previously.

Syntax

```
CHGVAR                  VAR(&CMD) VALUE('MAILTO')
CHGVAR                  VAR(&PARM1) VALUE('Recipients')
CHGVAR                  VAR(&PARM2) VALUE('[*TO|*CC|*BCC]')
CALL                    PGM(LNCCMD) + PARM(&HANDLE &CMD &OPT &PARM1
&PARM2
&RESULT)
```

Parameters

Parameters	
Parm1	List of recipients separated by semicolons.
Parm2	Role of the recipients : *TO : default value *CC : Carbon Copy *BCC : Blind Carbon Copy

Example

```
LNCCMD CMD(MAILTO) PARM1('"tech@easycom-aura.com";"info@easycom-
aura.com"') PARM2('*TO')

LNCCMD CMD(MAILTO) PARM1('"tech@easycom-aura.com";"info@easycom-
aura.com"') PARM2('*CC')

LNCCMD CMD(MAILTO) PARM1('"tech@easycom-aura.com";"info@easycom-
aura.com"') PARM2('*BCC')
```

See also

- LNCSNDMAIL - CL Command
- MAILPREP
- MAILLATT

- MAILTEXT
- MAILPRTY
- MAILSUBJ
- MAILREPORT
- MAILSEND
- MAILEND

MAIXATT command

Compatible Lotus Notes.
Extract an attachment from the last message listed by MAILLATT.

On return, &RESULT contains information about the attachment.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MAIXATT')
CHGVAR          VAR(&PARM1) VALUE('
                File=Path to the destination file";
                [Note=True/False]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File** : Path to the destination file. This file will contain the attachment. **Note** : If Note is true, the body of the message is extracted.
RESULT	After the call, parameter &RESULT contains : **FALSE** if no attachment was extracted, **Type="File type";** **Name="Original file name" ;** **Path="Original pathe to file name" ;**

See also

- LNCSNDMAIL - CL Command
- MAILLATT
- MAILMCHG
- MAILMDLT
- MAIXATT

MENU command

Creates an independant customized menu bar.

MENU must be used with the **MENUWAIT** command.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('MENU')
CHGVAR        VAR(&PARM1) VALUE('Captions="Names of buttons ;…"
              [;Tips="Help ballon ;…"]
              [;MenuName="Name of the menu"]')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

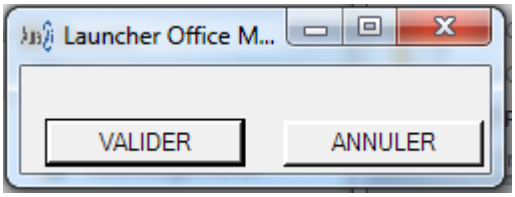
Parameters	
Parm1	**Captions :** Name of each button of the new menu bar. Names separator is semicolon (;). The number of labels determines the number of buttons on the menu bar. **Tips :** Allows to display a help balloon for each button. Tips separator is semicolon (;). **MenuName:** Name of the menu.

Notice

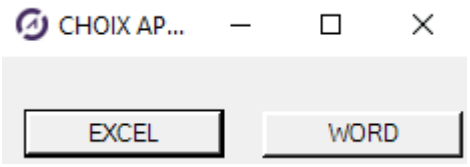
Upon MENUWAIT command call, AS/400 program stands by waiting for an user action. It will take over again when the menu will be closed, or upon one of the new menu button action.
The &RESULT variable of the MENUWAIT command, contains the pressed button number on 5 digits (From 1 to n).

Example

```
CHGVAR      VAR(&CMD) VALUE('MENU')
CHGVAR      VAR(&PARM1) +
              VALUE('CAPTIONS="VALIDER;ANNULER";TIPS="VAL+
              IDER LE DOCUMENT;ANNULER"')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```




```
CHGVAR      VAR(&CMD) VALUE('MENU')
CHGVAR      VAR(&PARM1) +
              VALUE('CAPTIONS="EXCEL;WORD";TIPS="OUVRIR +
              EXCEL;OUVRIR WORD";MENUNAME="CHOIX APPLI +
              OFFICE"')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```



MENUWAIT command

Client program waits for an action on the customized menu created by the MENU command, or for the menu closing.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('MENUWAIT')
CHGVAR        VAR(&PARM1) VALUE(' ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Example

```
      CHGVAR     VAR(&CMD) VALUE('MENU')
      CHGVAR     VAR(&PARM1) +
                   VALUE('CAPTIONS="VALIDER;ANNULER";TIPS="VAL+
                   IDER LE DOCUMENT;ANNULER"')
      CHGVAR     VAR(&PARM2) VALUE(' ')
      CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                   &PARM2 &RESULT)


  /* ATTENTE CLICK SUR BOUTON  */
              CHGVAR     VAR(&CMD) VALUE('MENUWAIT')
              CHGVAR     VAR(&OPT) VALUE(' ')
              CHGVAR     VAR(&PARM1) VALUE(' ')
              CHGVAR     VAR(&PARM2) VALUE(' ')
              CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                           &PARM2 &RESULT)
              IF         COND(&RESULT *NE '00000') THEN(GOTO +
                           CMDLBL(WORDCLICK))

              GOTO       CMDLBL(FINISH)

 WORDCLICK:
              SNDMSG     MSG('CLICK = ' *TCAT &RESULT) TOUSR(QPGMR)
 FINISH:
```

MIN command

Minimizes the current window.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MIN')
CHGVAR          VAR(&PARM1) VALUE(['Window handle'])
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The "*Window handle*" is not mandatory. This handle was returned by a previous call to LNCCMD with command STO. If the handle is given, the windows having the specified handle is minimized, otherwise, the current active window is minimized.
RESULT	On return, this parameter contains the handle (10 digits) of the minimized windows.

See also

- RCL
- STO

MOVEFILE command

Moves a file from one directory to another one, on the PC.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('MOVEFILE')
CHGVAR              VAR(&PARM1) VALUE('
                    File="File to move"
                    ;NewFile="Moved file"
                    [;Replace=true/false]
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File** : Complete path and name of the file to move. **NewFile** : Complete path and name of the moved file. **Replace** = true, to automatically replace the moved file if it already exists. By default, Replace=false.
RESULT	Contains eventually error message.

Note

If the destination file already exists and Replace=false, then an error will be returned. Example in the trace:

```
6540 2019-4-17 10:13:4.808 ERROR: the destination file already
exists, and Replace=false
6540 2019-4-17 10:13:4.811 *REJ
```

Example

```
LNCCMD CMD(MOVEFILE) PARM1('File="C:\temp\SBOOK.csv"; +
NewFile="C:\temp\TEST\SBOOK.csv";replace=true')
```

MSGBOX command

Displays a message box on the Windows station.

Not available when Launcher is running as a Windows service.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('MSGBOX')
CHGVAR          VAR(&PARM1) VALUE('
                Text="Message text";
                Caption="Window title"
                ')
CHGVAR          VAR(&PARM2) VALUE('
                Type=Button type
                [;Default=Button Number]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

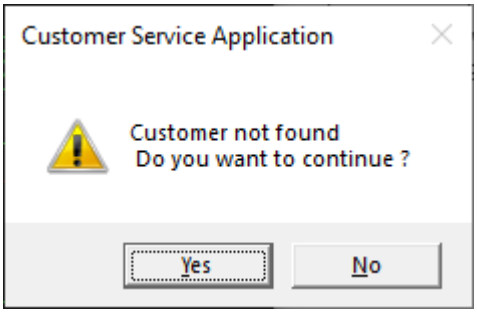
Parameters

Parameters	
Parm1	Parm1 contains the title and the text to display in the message box : **Text** = Text in the message box. **Caption** = Title of the window.
Parm2	Parm2 contains options of the message box. **Type=**type of message box. It can be one of the following values: **MB_ABORTRETRYIGNORE, MB_OK, MB_OKCANCEL, MB_RETRYCANCEL, MB_YESNO, MB_YESNOCANCEL**. One of the following values can be added (with the + sign) to display an icon in the message box: **MB_ICONEXCLAMATION, MB_ICONWARNING, MB_ICONINFORMATION, MB_ICONASTERISK, MB_ICONQUESTION, MB_ICONSTOP, MB_ICONERROR, MB_ICONHAND** **Default**=Button number to make active as default. Optional.
RESULT	Contains the number of the button clicked by the user.

Example

This example shows how to make a message box, with buttons Yes and No, with the Exclamation mark icon.

```
CHGVAR          VAR(&CMD) VALUE('MSGBOX')
CHGVAR          VAR(&PARM1) VALUE('Text="Customer not found %LF%
                Do you want to continue ?";Caption="Customer Service
                Application"')
CHGVAR          VAR(&PARM2) VALUE('Type= MB_ICONEXCLAMATION+
                MB_YESNO')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```



Note

%LF% : Inserts a paragraph break in the text to be displayed.

NCL command

Allows to create a new column in the source file created with the CRF command.

The column will be inserted to the right of the existing columns.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('NCL')
CHGVAR          VAR(&PARM1) VALUE('NameColumn')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Name of the column to create.

Example

```
CHGVAR          VAR(&CMD) VALUE('NCL')
CHGVAR          VAR(&PARM1) VALUE('Column1')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

See also

- CRF
- ENF
- NLN
- VAL

NLN command

Creates a new row in the source file created with the CRF command.

The line will be added at the end of the source file.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('NLN')
CHGVAR              VAR(&PARM1) VALUE(' ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

See also

- CRF
- ENF
- NCL

NOP command

Allows to add a comment in LAUNCHER administration Trace tab.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('NOP')
CHGVAR          VAR(&PARM1) VALUE('Comment')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Comment.

Example

```
CHGVAR      VAR(&CMD) VALUE('NOP')
CHGVAR      VAR(&PARM1) VALUE('Comment')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
            &PARM2 &RESULT)
```

OLECALL command

Gives access to method or property of an object instance created with OLECREATE.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('OLECALL')
CHGVAR          VAR(&PARM1) VALUE('"Identifier of instance"')
CHGVAR          VAR(&PARM2) VALUE('Method or property to execute')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	ID of the object instance. This ID was returned by a previous call of OLECREATE.
Parm2	Text of the method or property to apply. Examples : **ActiveDocument.SaveAs("C:\MyDoc\Doc1.doc")** Saves current document. **ActiveDocument.Words.Count** Returns document words number.
RESULT	At the output, the &RESULT parameter contains a value possibly returned by the called method, or the value of a requested property.

The use of this command requires a good Windows OLE programming knowledge as well as a good knowledge of the object to communicate with.

Example

See OLECREATE.

See also

- OLECREATE

OLECREATE command

Allows to create an OLE object instance.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('OLECREATE')
CHGVAR          VAR(&PARM1) VALUE('"Reference towards the object"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Reference to the application for which you want to create an instance. For instance, to create a Microsoft Access instance, specify : "Access.application". To create a Lotus instance : "Notes.NotesSession". Object references are located in Windows registry, under the HKEY_CLASSES_ROOT key.
RESULT	At return, the **&RESULT** parameter contains the **identifier of the instance of the created object**, on 10 characters. This identifier will be used by the **OLECALL** command.

Use of this command requires a good Windows OLE programming knowledge as well as a good knowledge of the object to communicate with.

Example

```
LNCCMDR     CMD(OLECREATE) PARM1(ACCESS.APPLICATION) +
              RESULT(&RES)

LNCCMD      CMD(OLECALL) PARM1(&RES) +
              PARM2('NewCurrentDatabase("LAUNCH24.MDB")')

LNCCMD      CMD(OLECALL) PARM1(&RES) +
              PARM2('DoCmd.TransferText(0;;"SP_CUST34";"C+
              :\AuraData\ACCESS_2010_test\LNC003_virgule.+
              TXT";true;;1200)')

LNCCMD      CMD(OLERELEASE) PARM1(&RES)
```

See also

- OLECALL

OLERELEASE command

Allows to release an object created with the OLECREATE command.

LNCCLOSE command allows to release all automatically.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('OLERELEASE')
CHGVAR          VAR(&PARM1) VALUE('"Reference towards the object"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Reference to the application whose object we want to release.

Example

See OLECREATE.

See also

- OLECALL
- OLECREATE

PARSEXML command

Allows to get tags values into XML file, or the value of an attribute.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PARSEXML')
CHGVAR          VAR(&PARM1) VALUE('
                XMLFile="XML file name"
                ;Tags="Tags list or only 1 tag"
                [;Attribut="Attribute name"]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**XMLFile** : Path to the XML file. **Tags**: List of tags that you want to retrieve value. Tags are separated by a semicolon (;). **Attribut**: name of the attribute of the tag whose value we want to retrieve. **If you are looking for the value of an attribute, you must specify only one tag.**
RESULT	In return, the list of values corresponding to the list of tags, separated by a semicolon (;). If the **Attribute** parameter is specified, you will have the value of the attribute in return.

Example 1

Assuming we have this XML file, and we want to retrieve the following values (in red):

```
<?xml version="1.0" encoding="utf-8"?>
<tnsb:Notification>
        <tnsb:NotificationType>GENERAL</tnsb:NotificationType>
        <tnsb:Request>
```

```
            <tnsb:Type>PAPER</tnsb:Type>
            <tnsb:TrackId>test_C33012646115</tnsb:TrackId>
            <tnsb:DepositId>25527400032</tnsb:DepositId>
            <tnsb:ReceptionDate>01/10/2015 1:17:34</tnsb:ReceptionDate>
            <tnsb:FoldsCount>1</tnsb:FoldsCount>
            <tnsb:Status>ACCEPT</tnsb:Status>
            <tnsb:PaperOptions>
                  <tnsb:EnvelopeType>C6</tnsb:EnvelopeType>
                  <tnsb:PrintDuplex>false</tnsb:PrintDuplex>
                  <tnsb:DocumentCount>1</tnsb:DocumentCount>
                  <tnsb:BilledPageCount>2</tnsb:BilledPageCount>
                  <tnsb:PageCount>2</tnsb:PageCount>
            </tnsb:PaperOptions>
      </tnsb:Request>
</tnsb:Notification>
```

If we execute the following command:

```
 LNCCMDR     CMD(PARSEXML) +
               PARM1('XMLFILE="\\ADX01\Docs\notification.xml"; +
Tags="tnsb:Notification.tnsb:Request.tnsb:TrackId;tnsb:N+
               otification.tnsb:Request.tnsb:ReceptionDate+
               ;tnsb:Notification.tnsb:Request.tnsb:Status')
RESULT(&RESULT)
```

In return, we get:

*MSGtest_12;01/10/2015 1:17:34;ACCEPT

Example 2

Assuming we have this XML file, and we want to retrieve the following values (in red):

```
<?xml version="1.0" encoding="UTF-8"?>
<records>
      <record name="export" type="extra" date="20200402">
            <produits quantity="8" name="CatA">
                  <produit name="A1" value="AS1254"/>
                  <produit name="A2" value="UIPLH"/>
                  <produit name="A3" value="125698YT"/>
                  <produit name="A4" value="KJZ745"/>
                  <produit name="A5" value="POJ523"/>
                  <produit name="A6" value="127YUI"/>
                  <produit name="A7" value="RTFG58"/>
                  <produit name="A8" value="GHUP263"/>
```

The following commands will have to be executed:

```
LNCCMDR    CMD(PARSEXML) +
```

```
            PARM1('XMLFILE="C:\test\produits.xml"+
            ;Tags="records.record";Attribut="type"') +
            RESULT(&RESULT)


LNCCMDR     CMD(PARSEXML) +
            PARM1('XMLFILE="C:\test\produits.xml"+
            ;Tags="records.record.produits.produit[6]"+
            ;Attribut="value"') RESULT(&RESULT)
```

Please note that the tag index starts at 0. To have the seventh product, we put the index = 6

PDFALLXL command

Allows you to create a PDF for each sheet of an Excel workbook passed as a parameter. Each PDF will have the name of the Excel sheet followed by the extension ".PDF"..

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFALLXL')
CHGVAR          VAR(&PARM1) VALUE('XL="Path of the Excel workbook"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**XL**: Complete path of the Excel workbook.

Example

```
LNCCMD    CMD(PDFALLXL) PARM1('XL="c:\A\test.xlsx"')
```

PDFCONCAT command

Allows to get concat several PDF files into a new PDF file.

The maximum number of PDF to concatenate with the PDFCONCAT command is 104.

You should have to increase size of Parm1 (see Example 2).

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFCONCAT')
CHGVAR          VAR(&PARM1) VALUE('"<path_file1>.pdf";"<path_file2>.pdf";
                "<path_file3>.pdf";"<path_file4>.pdf"')
CHGVAR          VAR(&PARM2) VALUE('Dest="Path of the new PDF"')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	List of the PDF files to concat (complete path between double quotes), separated by a semicolon.
Parm2	**Dest**: Complete path of the new PDF, corresponding to the concatenation of the PDF files contained into Parm1.

Example 1

```
LNCCMD      CMD(PDFCONCAT) +
             PARM1('"C:\A\a.pdf";"C:\A\b.pdf";"C:\A\R_27+
             WORD.pdf";"C:\A\c.pdf"') +
             PARM2('Dest="C:\A\res2.pdf"')
```

Example 2

You can use of the DIRLIST command to list PDFs in a directory, and send them as a parameter of PDFCONCAT.

In this example, the PROPERTY command was used to change the size of the Parm1, Parm2 and Result parameters, only for the current connection.

At the end of the connection, the default settings are applied.

PROPERTY has no effect when using LNCCMD commands. The CALL PGM (LNCCMD) program call must be used.

It will of course be necessary to change the size of the variables.

The cumulative size for Parm1 and Parm2 must not exceed 32700.
The size for Result should not exceed 32700.

```
PGM
DCL         VAR(&RES) TYPE(*CHAR) LEN(512)
DCL         VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL         VAR(&HDL) TYPE(*CHAR) LEN(100) VALUE('*ONLY ')
DCL         VAR(&PARM1) TYPE(*CHAR) LEN(31000)
DCL         VAR(&PARM2) TYPE(*CHAR) LEN(1000)
DCL         VAR(&LIST) TYPE(*CHAR) LEN(20000)
DCL         VAR(&CNT) TYPE(*DEC) LEN(3) VALUE(0)
DCL         VAR(&REP) TYPE(*CHAR) LEN(3000)

LNCOPEN

LNCCMD      CMD(PROPERTY) +
              PARM1('PARMSIZE(31000,1000,512)')

CHGVAR      VAR(&REP) VALUE('C:\A\concat')
CHGVAR      VAR(&LIST) VALUE(' ')
CHGVAR      VAR(&PARM1) VALUE('PATTERN="*.pdf";PATH="' +
              *TCAT &REP *TCAT '"')
CHGVAR      VAR(&PARM2) VALUE('FIRST=TRUE')

CALL        PGM(LNCCMD) PARM(&HDL 'DIRLIST' &OPT &PARM1 +
              &PARM2 &RES)

IF          COND(&RES *NE ' ') THEN(DO)
     CHGVAR     VAR(&LIST) VALUE(&LIST *TCAT '"' *TCAT &REP +
                  *TCAT '\' *TCAT &RES *TCAT '"')
     CHGVAR     VAR(&CNT) VALUE(&CNT + 1)
ENDDO

CHGVAR      VAR(&PARM2) VALUE('FIRST=FALSE')

DOWHILE     COND(&RES *NE ' ')

     IF          COND(&CNT *GT 1) THEN(DO)
           CHGVAR     VAR(&LIST) VALUE(&LIST *TCAT ';"' *TCAT &REP +
                 *TCAT '\' *TCAT &RES *TCAT '"')
     ENDDO

     CHGVAR      VAR(&CNT) VALUE(&CNT + 1)

     CALL        PGM(LNCCMD) PARM(&HDL 'DIRLIST' &OPT &PARM1 +
                  &PARM2 &RES)
ENDDO

CHGVAR      VAR(&PARM1) VALUE(&LIST)
CHGVAR      VAR(&PARM2) +
              VALUE('DEST="C:\TEMP\FINAL_CONCAT.PDF"')
```

```
CALL        PGM(LNCCMD) PARM(&HDL 'PDFCONCAT' &OPT +
              &PARM1 &PARM2 &RES)
LNCCLOSE

ENDPGM
```

PDFCOUNT command

Allows to get the number of pages of an existing PDF.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFCOUNT')
CHGVAR          VAR(&PARM1) VALUE('File="Path of the PDF file"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File**: Complete path of the PDF, for which we want to know the number of pages.
RESULT	In return of the command, we will have the number of pages of the PDF file. Example: PageCount=00002

Example

```
LNCCMDR     CMD(PDFCOUNT) PARM1('File="c:\A\b.pdf"') +
              RESULT(&RESULT)
```

&RESULT will contain the following value : PageCount=00002.

PDFENCRYPT command

Protects an existing PDF document.

The AES 256 encryption is used.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFENCRYPT')
CHGVAR          VAR(&PARM1) VALUE('
                File="PDF file name"
                ;OwnerPWD="Owner password"
                [;UserPWD="User password"]
                [;EnablePrint= True / False ]
                [;EnableCopy= True / False]
                [;EnableEdit= True / False]
                [;EnableNotes= True / False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

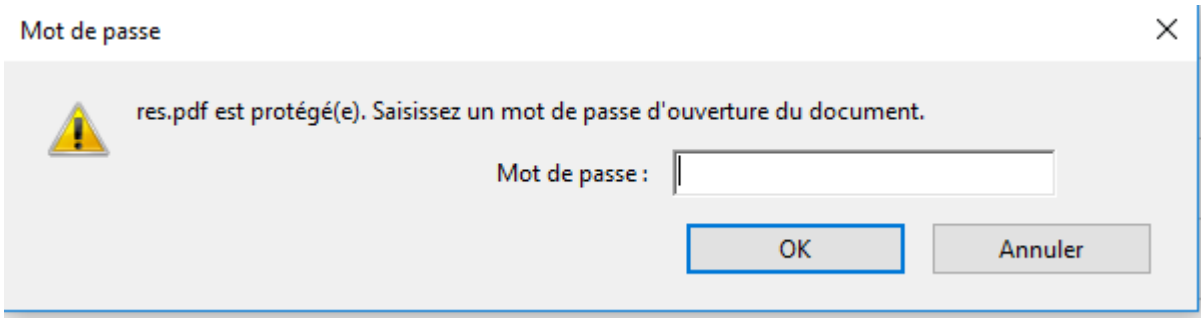
Parameters	
Parm1 or Parm2	**File** : Path to the existing PDF file to protect. **OwnerPWD**: Owner password (**mandatory**). With this password, the owner will be able to do any operation on the document. **UserPWD**: User password (**Optional**). If this password is set, it will be required when the document is opened. **EnablePrint**: Authorize to print the document. Optional. **EnableCopy**: Authorize to copy the document. Optional. **EnableEdit**: Authorize to modify the document. Optional. **EnableNotes**: Authorize the user to add notes. Optional.

Example

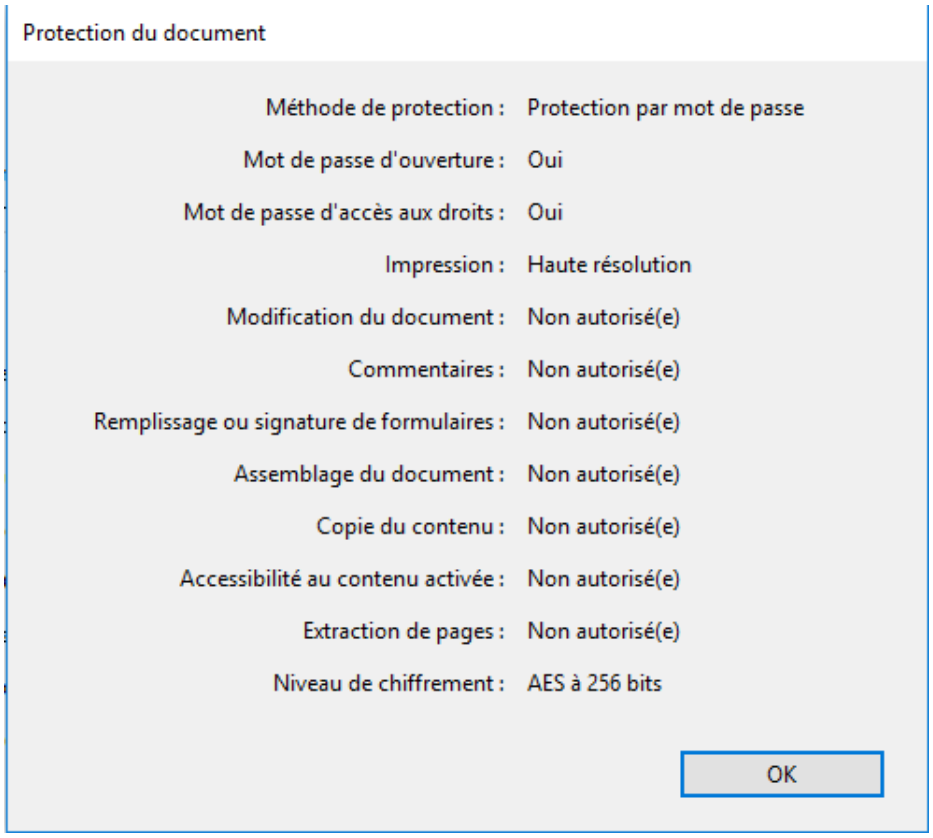
To protect a document by adding owner and user passwords, you can use the following command. Please note that printing is allowed.

```
LNCCMD      CMD(PDFENCRYPT) +
              PARM1('FILE="C:\temp\+
              res.pdf";OwnerPWD="ced";UserPWD="aura";E+
              nablePrint=true')
```

Opening the PDF will require a password (UserPWD):



The PDF will also have the following properties:



PDFEXTRACT command

Allows to extract some pages of an existing PDF to create another one.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFEXTRACT')
CHGVAR          VAR(&PARM1) VALUE('
                Dest="Path of the new PDF";
                Origin="Path of the original PDF";
                StartPage=page number of the fisrt page to extract;
                EndPage=page number of the last page to extract
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Dest**: Complete path of the new PDF, containing some pages of the original PDF. **Origin**: Complete path of the original PDF from which we want to extract some pages. **StartPage** : number of the first page of the original PDF to extract. **EndPage** : number of the last page of the original PDF to extract.

Example

```
LNCCMD      CMD(PDFEXTRACT) +
              PARM1('Dest="c:\A\res_5.pdf";Origin="C:\A\r+
              es.pdf";StartPage=1;EndPage=2')
```

The « res_5.pdf » PDF will contain the pages #1 and #2 of the « res.pdf » PDF.

PDFFACTX command

Allows you to create a Factur-X electronic invoice.

Factur-X is a Franco-German standard for mixed electronic invoicing, based on a PDF file (PDF/A3 standard) representing the original invoice and including a structured data file (XML).

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFFACTX')
CHGVAR          VAR(&PARM1) VALUE('
                PDF="Complete path of the PDF/A3 file"
                ;Xml="Complete path of the XML file"
                ;XMP=" Complete path of the XMP file"
                ;RelationShip="value";
                ;Description="Factur-X description";
                ;FacturX="Complete path of the Factur-X file"
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 ou Parm2	**PDF**: Complete path of the PDF/A3 file, **Xml**: Complete path of the XML file. **XMP** : Path of the XML file containing the XMP Meta data of the PDF/A. **RelationShip** : Data relationship between the attached XML file and the PDF/A. Can take one of the following values (be careful to respect case): Data, Source, Alternative, Supplement or Unspecified. **Description** : Factur-X description. **FacturX**: Complete path of the Factur-X file to save.

Note

If you want to create the PDF/A3 from a Word document, you can use the WSAVEAS command with Format=PDF and PdfA=true.

See the following example

Example

```
PGM

DCL         VAR(&FILE) TYPE(*CHAR) LEN(2000)
DCL         VAR(&PATH) TYPE(*CHAR) LEN(2000)

CHGVAR      VAR(&PATH) VALUE('C:\temp\4')
CHGVAR      VAR(&FILE) VALUE('Doc2.docx')
LNCOPEN     SVRADDR('*DEV')
LNCCMD      CMD(WORDOPEN)

LNCCMD      CMD(WOPENFILE) PARM1('File="' *TCAT &PATH +
              *TCAT '\' *TCAT &FILE *TCAT '"')
LNCCMD      CMD(WSAVEAS) +
              PARM1('File="C:\temp\4\Doc3.pdf";Forma+
              t=PDF;PdfA=true')
LNCCMD      CMD(WORDCLOSE) PARM1('SAVE=FALSE')
LNCCMD      CMD(PDFFACTX) +
              PARM1('PDF="C:\temp\5\Doc+
              3.pdf";Xml="C:\temp\5\fa+
              ctur-x.xml";FacturX="C:\temp+
              \5\res29.PDF";DESCRIPTION="AURA +
              FACTUR-X";RELATIONSHIP="Data";XMP="C:\temp+
              \5\metadata_2023-22027.xml+
              "')
LNCCLOSE
ENDPGM
```

PDFFRIM command

Allows to create a PDF from an image.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('PDFFRIM')
CHGVAR        VAR(&PARM1) VALUE('
              Image="Complete path of the image file"
              ;Dest="Complete path of the PDF file"
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 ou Parm2	**Image**: Complete path of the image file, with one of the following formats: - jpeg - bmp - png - gif - tiff **Dest**: Complete path of the PDF file.

Example

```
LNCCMD      CMD(PDFFRIM) +
              PARM1('Image="C:\temp\scanB.gif";Dest="C:\t+
              emp\scanC.pdf"')
```

PDFFRXL command

Create a PDF from the print area defined in the current Excel workbook.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFFRXL')
CHGVAR          VAR(&PARM1) VALUE('
                File="Complete path of the PDF file"
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 ou Parm2	**File**: Complete path of the PDF file to create. The PDF will be based on the print area of the current Excel workbook.

Example

Definition of the print area and creation of the PDF.

```
LNCCMD CMD(XLGOTOSH) PARM1('Sheet2')

LNCCMD CMD(XLCELLS)
PARM1('Ref="$A$1";RefTo="$J$27";Name="Print_Area"')

LNCCMD CMD(PDFFRXL) PARM1('File="C:\temp\resPDFFRXL.pdf"')
```

PDFMERGE command

Layer or concatenate two PDF files.

This command can be useful to add a watermark to a PDF document. It uses LAUNCHER_PDF.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFMERGE')
CHGVAR          VAR(&PARM1) VALUE('
                File="Output file name"
                ;Source1="PDF Document No 1"
                ;Source2="PDF Document No 2"
                [;Repeat= True / False]
                [;Above= True / False]
                [;Append= True / False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 Or Parm2	**File** : Full path and name of the PDF file resulting from the merge or concatenation, which will be performed between the two files designated by "Source1" and "Source2". **Source1** : Path and name of the first original PDF file. **Source2** : Path and name of the second PDF file. This file will be the watermark, or it will be append to the end of **Source1**, if **Append** is true. **Repeat** : If Repeat is true, and Above is true, the document **Source2** will be repeated on all the pages of **Source1**. Else, the output file will have a watermark only on the first page. **Above** : True for Source2 content to appear as a Source1 watermark. **Append** : True to append contain of Source2 to Source1. No watermark will be added to the document.

Note

If Above and Append are not specified, the default True value of Above will be taken into account. We will proceed to the establishment of a watermark.

If Append=true then we will concatenate, whatever the value of Above.

Example 1

Build a temporary PDF file with word "DUPLICATA" as watermark.
Merge this PDF file with an existing file, and repeat the watermark on all the pages.

```
                /* Open Word with a blank page */
CHGVAR          VAR(&CMD) VALUE('WORDOPEN')
CHGVAR          VAR(&PARM1) VALUE('New')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)

                /* Prepare the PDF driver, set watermark to "DUPLICATA" */
CHGVAR          VAR(&CMD) VALUE('PDFPRINTER')
CHGVAR          VAR(&PARM1) VALUE('File="%TEMP%\watermark.pdf" +
                WMText="DUPLICATA"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)

                /* Print the blank page with PDF driver */
CHGVAR          VAR(&CMD) VALUE('WPRINT')
CHGVAR          VAR(&PARM1) VALUE('Printer=="LAUNCHER_PDF"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
                /* Merge "Original" and " watermark " */
CHGVAR          VAR(&CMD) VALUE('PDFMERGE')
CHGVAR          VAR(&PARM1) VALUE('File="C:\documents\result.pdf"; +
                Source1="C:\Documents\original.pdf"; +
                Source2="%TEMP%\watermark.pdf"; +
                Repeat=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
                /* "C:\documents\result.pdf" contains "DUPLICATA"
                on all the pages */
```

Example 2

Concatenation of a.pdf and b.pdf to create res_con.pdf.

```
LNCCMD      CMD(PDFMERGE) +
              PARM1('FILE="C:\A\res_con.pdf";SOURCE1="C:\+
              A\a.pdf";SOURCE2="C:\A\b.pdf";APPEND=TRUE')
```

PDFORDERX command

Allows you to create a Order-X electronic order.

Based on Factur-X, Order-X is a Franco-German standard for mixed electronic order, based on a PDF file (PDF/A3 standard) representing the original order and including a structured data file (XML).

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFORDERX')
CHGVAR          VAR(&PARM1) VALUE('
                PDF="Complete path of the PDF/A3 file"
                ;Xml="Complete path of the XML file"
                ;XMP=" Complete path of the XMP file"
                ;RelationShip="value";
                ;Description="Order-X description";
                ;OrderX="Complete path of the Order-X file"
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 ou Parm2	**PDF**: Complete path of the PDF/A3 file, **Xml**: Complete path of the XML file. **XMP** : Path of the XML file containing the XMP Meta data of the PDF/A. **RelationShip** : Data relationship between the attached XML file and the PDF/A. Can take one of the following values (be careful to respect case): Data, Source, Alternative. **Description** : Order-X description. **OrderX**: Complete path of the Order-X file to save.

Note

If you want to create the PDF/A3 from a Word document, you can use the WSAVEAS command with Format=PDF and PdfA=true.

See the following example

Example

```
PGM

DCL         VAR(&FILE) TYPE(*CHAR) LEN(2000)
DCL         VAR(&PATH) TYPE(*CHAR) LEN(2000)

CHGVAR      VAR(&PATH) VALUE('C:\temp\5')
CHGVAR      VAR(&FILE) VALUE('order.docx')
LNCOPEN     SVRADDR('*DEV')
LNCCMD      CMD(WORDOPEN)

LNCCMD      CMD(WOPENFILE) PARM1('File="' *TCAT &PATH +
              *TCAT '\' *TCAT &FILE *TCAT '"')
LNCCMD      CMD(WSAVEAS) +
              PARM1('File="C:\temp\5\Doc.pdf";Forma+
              t=PDF;PdfA=true')
LNCCMD      CMD(WORDCLOSE) PARM1('SAVE=FALSE')
LNCCMD      CMD(PDFORDERX) +
              PARM1('PDF="C:\temp\5\Doc+
              .pdf";Xml="C:\temp\5\or+
              der-x.xml";OrderX="C:\temp+
              \5\res.PDF";DESCRIPTION="AURA +
              ORDER-X";RELATIONSHIP="Data";XMP="C:\temp+
              \5\metadata_2024.xml+
              "')
LNCCLOSE
ENDPGM
```

PDFOVER command

Allows to add an overlay on an existing PDF.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFOVER')
CHGVAR          VAR(&PARM1) VALUE('
                Dest="Path of the new PDF";
                Origin="Path of the original PDF";
                Overlay="Path of the PDF containing overlay"
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Dest** : complete path of the new PDF, corresponding to the original PDF with overlay applied. **Origin** : complete path of the original PDF on which we want to add an overlay. **Overlay** : complete path of the PDF file containing overlay.

Example

```
LNCCMD      CMD(PDFOVER) +
               PARM1('Dest="c:\A\res_2.pdf";Origin="C:\A\a+
               .pdf";Overlay="C:\A\over.pdf"')
```

PDFPRINT command

Print an existing PDF document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFPRINT')
CHGVAR          VAR(&PARM1) VALUE('
                File="PDF file name"
                ;Printer="Printer name"
                [PWD="User password"]
                [;StartPage= Number ]
                [;EndPages= Number]
                [;Copies= Number ]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File** : Path to the PDF file to print. **Printer**: Name of the printer to use. **PWD**: If the document is protected, set the user password. **StartPage**: Number of the first page to print. Default is 1. **EndPage**: Number of the last page to print. By default, the document is printed to the end. **Copies**: Number of copies to print. By default, number of copies=1.

Example

```
CHGVAR          VAR(&CMD) VALUE('PDFPRINT')
CHGVAR          VAR(&PARM1) VALUE('File="C:\Documents\Resultat.pdf"')
CHGVAR          VAR(&PARM2) VALUE('Printer="\\Server\HP Laser 2200"')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

See also

- PDFPRINTER

PDFPRINTER command

Allows to set PDF File generation options using LAUNCHER Office PDF print driver.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFPRINTER')
CHGVAR          VAR(&PARM1) VALUE('
                File="Output file"
                [;Directory="Output file directory"]
                [;Resolution=number]
                [;PaperSize= number]
                [;PaperWidth= number]
                [;PaperLength= number]
                [;Orientation= number]
                [;Append== True / False]
                [;EmbedFonts= True / False]
                [;WMText= "Text in watermark"]
                [;WMFont= "Police for the watermark"]
                [;WMSize= number]
                [;WMOrientation= number]
                [;WMHPos= number]
                [;WMVPos= number]
                [;WMColor= number]
                [;WMForeground= True / False]
                [;JPEGCompress= True / False]
                [;JPEGLeval= Nombre]
                [;Default= True / False]
                [;Encrypt= True / False]
                [;HyperLinks= True / False]
                [;ScalingOption= 1 / 0]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**File** : Path to PDF result file. **Directory**: Directory path to the resulting PDF file. This path can be ommited if it is set with the file name in the **File** property. **Resolution** : Output resolution in DPI. Possible values : 75, 150, 300, 600, 1200. **PaperSize** : Page format. Possible values : 1=Letter ; 5=legal ; 9=A4 ; 8=A3 ; 256=custom.

PaperWidth : Page width in tenth of millimeter.

PaperLength : Page length in tenth of millimeter.

Orientation: Orientation. 1=Portrait; 2=Landscape.

Append: True for add the new document at the end of PDF file if exist.
False by default.

EmbedFonts: True for embark the fonts in the PDF document.
False by default.

WMText : Watermark print text.

WMFont : Watermark font name (*Default*=Arial).

WMSize : Size in 1/100 inch (*Default*=70).

WMOrientation : Orientation in 1/10 degree (*Default*=3150).

WMColor : RGB color (*Default*=12632256).

WMHPos, WMVPos : Horizontal and vertical position in 1/100 inch (*Default= 100, 500*).

WMForeground : true = Watermark in foreground.

JPEGCompress : True to activate the compression of JPEG images.

JPEGLevel can have the following values :
3 for high compression,
7 for medium compression.
9 for low compression.

Default : True to make LAUNCHER_PDF as default printer.

Encrypt : True to protect the output PDF document. A protected PDF file cannot be printed, saved or edited.

HyperLinks : True to transform text begining with **"http:"** or **"www."**, into hyperlinks.

ScalingOption : If ScalingOption = 1, the content is adjusted to the size of the PDF (extended or shrunk). Default value =0 : no adjustment.

Note

PDFPRINTER command initialises the LAUNCHER_PDF driver.

The print command that follows the PDFPRINTER command call must indicate that the destination printer is "LAUNCHER_PDF".

If **WMText** is indicated, a watermark will be added on each page.
WMColor is set as red, Green, Blue, levels as shown bellow :
((Blue * 256) + Green) * 256) + Red.

Example 1

This example generates a PDF File from a Word file.
"DUPLICATE" will be watermark printed on each page.

```
PGM

LNCOPEN

LNCCMD CMD(WORDOPEN)

LNCCMD CMD(WOPENFILE) PARM1('C:\PROGRAM FILES\LAUNCHER400\+
SAMPLES\LNCSHELL.DOC') PARM2(VISIBLE)

LNCCMD CMD(PDFPRINTER) +
PARM1('FILE="C:\TEMP\LNCSHELL.PDF";wmtext+
="DUPLICATA"')

LNCCMD CMD(WPRINT) PARM1('PRINTER="LAUNCHER_PDF"')

LNCCMD CMD(WORDCLOSE)

LNCSHELL CMD('C:\TEMP\LNCSHELL.PDF') WAITCMD(*NO) +
VISIBLE(*YES) MINCURWIN(*YES) +
ACTION(OPEN) EXESRV(*CURRENT)
```

```
ENDPGM
```

Example 2

Create PDF from Excel file:

```
PGM

LNCOPEN

LNCCMD CMD(EXCELOPEN) PARM1('')

LNCCMD CMD(XLOPENFILE) +
PARM1('C:\MAGALIE\MIGRATIONAVEC.XLS')

LNCCMD CMD(PDFPRINTER) +
PARM1('FILE="C:\TEMP\MIGRATIONAVEC.PDF"')

LNCCMD CMD(XLPRINT) PARM1('PRINTER="LAUNCHER_PDF"')

LNCCMD CMD(EXCELCLOSE)

LNCSHELL CMD('C:\TEMP\MIGRATIONAVEC.PDF') +
WAITCMD(*NO) VISIBLE(*YES) +
MINCURWIN(*YES) ACTION(OPEN) EXESRV(*CURRENT)

ENDPGM
```

PDFSEARCH command

Allows to search text into the content of a PDF file.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFSEARCH')
CHGVAR          VAR(&PARM1) VALUE('
                File="Path of the PDF file";
                Text=text to search
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File**: Complete path of the PDF file (between double quotes). **Text**: Text to search into the content of the PDF file (without double quotes).
RESULT	In return: - true: if the text is found - false: if the text is not found.

Example

```
LNCCMDR     CMD(PDFSEARCH) +
               PARM1('File="c:\A\res.pdf";Text=Erica') +
               RESULT(&RESULT)
```

PDFSIGN command

Allows to sign an existing PDF with a PFX/P12/PKCS#12 cerificate.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PDFSIGN')
CHGVAR          VAR(&PARM1) VALUE('
                File="PDF file to sign"
                ;Certificate="Complete path of the certificate"
                ;SaveAs="Complete path of the signed PDF"
                ;PWD="Password of the certificate"
                ;SignedBy="Name that appears on signature"
                ;Reason="Reason that appears on signature"
                ;Location="Physical location of the person who signs the PDF"
                [;ImageFile= "Full path of the file containing the JPG image
                that is associated with the signature"]
                [;PageNumber= Page number on which to insert the signature]
                [;HorzPos= Horizontal position of the digital signature]
                 [;VertPos= Vertical position of the digital signature]
                 [;Zoom= increase (positive value) or decrease (negative
               value) image with x percent]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Paramètres	
Parm1 ou Parm2	**File** : PDF file to sign. **Certificate**: Complete path of the certificate with the following format : PFX/P12/PKCS#12. **PWD**: Password of the certificate. **SaveAs**: Complete path of the signed PDF. **SignedBy**: Name that appears on signature. **Reason**: Reason that appears on signature. **Location** : Physical location of the person who signs the PDF. **ImageFile** : Full path of the file containing the image that is associated with the signature (optional). JPG format mandatory.

PageNumber : Page number on which to insert the signature (optional). By default, the signature is inserted on page 1.

HorzPos : Horizontal position of the digital signature (optional). By default HorzPos = 0.

VertPos : Vertical position of the digital signature (optional). By default VertPos = 0.

Zoom : increase (positive value) or decrease (negative value) image with x percent (optional). By default Zoom = -50.

Example

```
LNCCMD      CMD(PDFSIGN) +
              PARM1('FILE="C:\A\PDF_test\resultat.pdf";Sa+
              veAs="C:\A\PDF_test\resultat_IM2.pdf";Signe+
              dBy="AURA";Reason="Approve";Location="NICE"+
              ;certificate="C:\temp\aura.p12";PWD="aura";+
              ImageFile="C:\temp\logo.jpg"')
```

PDFSPLIT command

Allows to split an existing PDF file into several new PDF files.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('PDFSPLIT')
CHGVAR        VAR(&PARM1) VALUE('"<path_file1>.pdf";"<path_file2>.pdf";
              "<path_file3>.pdf";"<path_file4>.pdf"')
CHGVAR        VAR(&PARM2) VALUE('
              Origin="Path of the PDF to split";
              SplitPagesQty=number of pages for each splitted PDF
              ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	List of the new PDF files created after splitting (complete path), separated by a semicolon.
Parm2	**Origin**: Complete path of the PDF file to split. **SplitPagesQty :** Number of pages for each splitted PDF. Note: If the PDF to split has 5 pages, and the SplitPagesQty parameter is equal to 2, therefore: - the 1st PDF specified into Parm1 will have 2 pages - the 2nd PDF specified into Parm1 will have 2 pages - the 3rd PDF specified into Parm1 will have 1 page

Example

```
LNCCMD      CMD(PDFSPLIT) +
              PARM1('"C:\A\b_1.pdf";"C:\A\b_2.pdf"') +
              PARM2('Origin="C:\A\b.pdf";SplitPagesQty=1')
```

PDFTOIM command

Create a JPG image for each page of a PDF file.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('PDFTOIM')
CHGVAR        VAR(&PARM1) VALUE('
              File="Chemin complet du fichier PDF"
              [;StartPage=page number of the fisrt page to extract]
              [;EndPage=page number of the last page to extract]
              [;PWD="password if necessary"]
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File**: Complete path of the PDF file. **StartPage** : number of the first page of the PDF to convert. **EndPage** : number of the last page of the PDF to convert. **PWD**: If the document is protected, set the user password.

Note

JPG files will be created in the same place as the PDF file.

They will have the same name as the PDF file with suffix, the number of the page.

Example: if the PDF is named "R_01WORD.PDF", then the JPEG files created next to the PDF will have the following names:
R_01WORD2.jpg, R_01WORD3.jpg, R_01WORD4.jpg ...

Example

```
LNCCMD CMD(PDFTOIM) +
PARM1('File="C:\A\PDFTOIMG\R_01WORD.PDF";StartPage=2;EndPage=4')
```

PRINTPRNF command

Sends a PRN file to a Windows printer.

A PRN file contains the data flow for the printer. (Ex : PCL). This data flow was previously generated by a "Print to file" command from Word or Excel.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PRINTPRNF')
CHGVAR          VAR(&PARM1) VALUE('
                File="Path to the PRN file to print";
                Printer="Printer name";
                [Document="Document name for the printer driver";]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**File** is the full path to the file to print. Most of the time, this file has the suffix PRN. It contains the printer dataflow. **Printer** sets the printer were the file must be printed. The printer must be able to manage the dataflow. For example : The PRN file was generated by using a PCL 5 printer driver, then, the printer used by command PRINTPRNF must be PCL 5 compliant. **Document** : This keyword allows to set the name of the print process into the Windows spooler. Optional.

Example

```
CHGVAR          VAR(&CMD) VALUE('WPRINT')
CHGVAR          VAR(&PARM1)
                VALUE('Printer="HPLJ1200";OutFile="%TEMP%\Out.prn"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
```

```
                    &PARM2 &RESULT)


CHGVAR              VAR(&CMD) VALUE('PRINTPRNF')
CHGVAR              VAR(&PARM1)
                    VALUE('File="%TEMP%\Out.prn";Printer="\\server\HPLJ
                    2200"')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

See also

- SETPRINTER

PROPERTY command

Allows to change some properties for the current LAUNCHER Office connexion.

So, you can change the LNCCMD command parameters size of LNCCMD.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('PROPERTY')
CHGVAR          VAR(&PARM1) VALUE('
                PARMSIZE(Parm1_Size,Parm2_Size,Result_Size)
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Property to change for the current LAUNCHER connexion. **PARMSIZE**(*Parm1_Size*,*Param_Size,Result_Size*) changes the size of the parameters Parm1, Parm2 and Result, used when calling the program LNCCMD. By default, the parameters size are : • 512 for Parm1, • 1024 for Parm2, • 512 for Result. If the given size for a parameter is null, the property is not changed for that parameter. The cumulated size for *Parm1* and *Parm2* cannot exceed 32700. The size for *Result* cannot exceed 32700.

Note

PROPERTY has no effect when using LNCCMD commands. It is necessary to use : CALL PGM(LNCCMD).

The LNCCMD command is defined with 512 characters for PARM1. The interpreter of the OS thus prevents the execution if superior.

In summary :

Add the following definitions at the beginning of your CL.

```
DCL VAR(&HANDLE) TYPE(*CHAR) LEN(50) VALUE('*ONLY')
DCL VAR(&CMD) TYPE(*CHAR) LEN(10)
DCL VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL VAR(&PARM1) TYPE(*CHAR) LEN(2000)
DCL VAR(&PARM2) TYPE(*CHAR) LEN(2000)
DCL VAR(&RESULT) TYPE(*CHAR) LEN(512)
```

When you need to send more than 512 characters, use :
`CALL PGM(LNCCMD)` and not the `LNCCMD` command

For instance, into your program, replace :
`LNCCMD CMD(FILEWRITE) PARM1(&FILEWRITE)`

By :

```
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &FILEWRITE &PARM2 &RESULT)
```

Example

```
DCL                     VAR(&PARM1) TYPE(*CHAR) LEN(256)
DCL                     VAR(&PARM2) TYPE(*CHAR) LEN(256)
DCL                     VAR(&RESULT) TYPE(*CHAR) LEN(512)

CHGVAR                  VAR(&CMD) VALUE('PROPERTY')
CHGVAR                  VAR(&PARM1) VALUE('PARMSIZE(256,256,512)')
CHGVAR                  VAR(&PARM2) VALUE(' ')
CALL                    PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                        &RESULT)
```

RCL command

Recalls the stored window.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('RCL')
CHGVAR          VAR(&PARM1) VALUE(['Window handle'])
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The "Window handle" is not mandadory. This handle was returned by a previous call to LNCCMD with command STO or MIN. If the handle is given, the windows having the specified handle is restored, otherwise, latest window stored by command STO, is restored.

See also

- MIN
- STO

REMOVEDIR command

Deletes an empty directory on a Windows PC/Server.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('REMOVEDIR')
CHGVAR              VAR(&PARM1) VALUE('
                    Dir="Directory to delete"
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2
                    &RESULT)
```

Parameters

Parameters	
Parm1 or Parm2	**Dir** : Complete path and name of the directory to delete.
RESULT	Contains eventually error message.

Note

The directory to delete has to be empty and sufficient rights are necessary for the user profile.

Example

```
LNCCMD CMD(REMOVEDIR) PARM1('Dir="C:\temp\DEL"')
```

RSTPRINTER command

Restore the origin parameters of the specified printer.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('RSTPRINTER')
CHGVAR              VAR(&PARM1) VALUE('Printer="Printer name"')

CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &RESULT)
```

Parameters

Parameters	
Parm1	**Printer** = Printer name.

See also

- SETPRINTER

SAISI command

Create a customized form.

It is possible to specify the form title, and 7 fields with tips.

The SAISI command has to be followed by the SAISIWAIT command.

To work correctly, it is recommanded to have the Firewall activated, and the Launcher program (lncsrv.exe and lncadm.exe) into the list of the allowed apps of the Firewall.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('SAISI')
CHGVAR          VAR(&PARM1) VALUE('
                CAPTIONS="Fields captions"
                [;TIPS="Fields tips"]
                [;FIELDNAME="Form title"]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

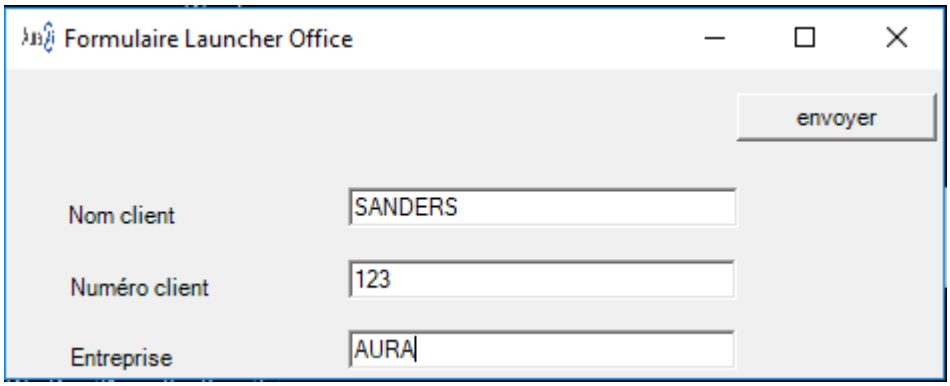
Parameters	
Parm1	**CAPTIONS** : caption for each field, separated by semicolon. Up to 7 fields. **TIPS :** tip for each field, when the mouse is over, separated by semicolon. The number of tips has to be the same than the number of captions. Optional. **FIELDNAME :** form title. Optional.

Example

```
/* CREATION MENU  */
             CHGVAR     VAR(&CMD) VALUE('SAISI')
             CHGVAR     VAR(&PARM1) VALUE('CAPTIONS="Nom +
                          client;Numéro +
                          client;Entreprise";TIPS="Ecrire le nom +
                          du client;Ecrire le numéro client;Ecrire +
                          le nom entreprise";FIELDNAME="Formulaire +
                          Launcher Office"')
```

```
             CHGVAR      VAR(&PARM2) VALUE(' ')
             CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                           &PARM2 &RESULT)

/* ATTENTE CLICK SUR BOUTON  */
            CHGVAR      VAR(&CMD) VALUE('SAISIWAIT')
            CHGVAR      VAR(&OPT) VALUE(' ')
            CHGVAR      VAR(&PARM1) VALUE(' ')
            CHGVAR      VAR(&PARM2) VALUE(' ')
            CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                          &PARM2 &RESULT2)
            LNCCMD      CMD(NOP) PARM1(&RESULT2)
```

The created form is the following:



When fields are filled, the user click on the "envoyer" button. The filled data are contained into &RESULT2 of the SAISIWAIT command, separated by semicolon.

In this example, &RESULT2 value contains :

SANDERS;123;AURA

SAISIWAIT command

Allows to retrieve data filled into the customized form, created by the SAISI command.

The SAISIWAIT command has to follow the SAISI command.

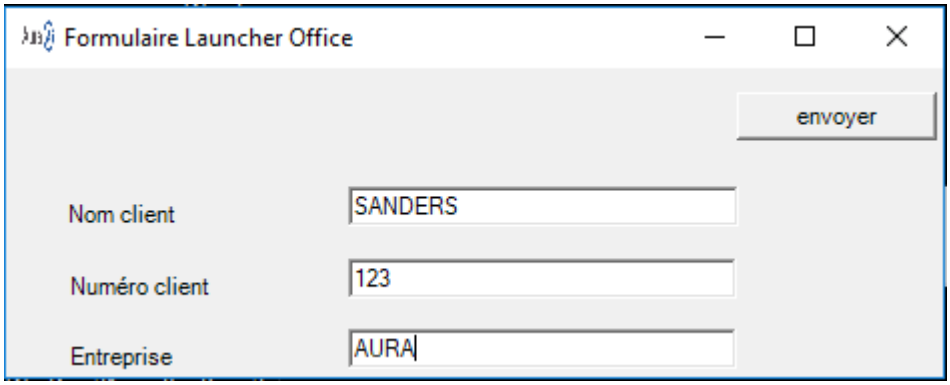
Syntax

```
CHGVAR        VAR(&CMD) VALUE('SAISIWAIT')
CHGVAR        VAR(&PARM1) VALUE(' ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Example

```
/* CREATION MENU  */
             CHGVAR     VAR(&CMD) VALUE('SAISI')
             CHGVAR     VAR(&PARM1) VALUE('CAPTIONS="Nom +
                          client;Numéro +
                          client;Entreprise";;TIPS="Ecrire le nom +
                          du client;Ecrire le numéro client;Ecrire +
                          le nom entreprise";FIELDNAME="Formulaire +
                          Launcher Office"')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                          &PARM2 &RESULT)

/* ATTENTE CLICK SUR BOUTON  */
            CHGVAR     VAR(&CMD) VALUE('SAISIWAIT')
            CHGVAR     VAR(&OPT) VALUE(' ')
            CHGVAR     VAR(&PARM1) VALUE(' ')
            CHGVAR     VAR(&PARM2) VALUE(' ')
            CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                         &PARM2 &RESULT2)
            LNCCMD     CMD(NOP) PARM1(&RESULT2)
```

The created for is the following:



When fields are filled, the user click on the "envoyer" button. The filled data are contained into &RESULT2 of the SAISIWAIT command, separated by semicolon.

In this example, &RESULT2 value contains :

SANDERS;123;AURA

SCAN command

Allows you to control a scanner and save the scanned document in different formats.

The name of the scanner must be the one returned by the LISTSCAN command.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('SCAN')
CHGVAR        VAR(&PARM1) VALUE('
              Scanner="Scanner name"
              ;Type="Format in which the scanned document will be saved"
              ;Dest="Complete path of the scanned document"
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1 ou Parm2	**Scanner** : Scanner name. Use the LISTSCAN command to see the scanner name to use. **Type**: Format in which the scanned document will be saved. The different possible formats are : - jpeg - bmp - png - gif - tiff - pdf **Dest**: Complete path of the scanned file.

Example

```
LNCCMD      CMD(SCAN) PARM1('Scanner="HP LJ100 M175 +
              Scan";Dest="C:\temp\scanB.gif";Type="gif"')


LNCCMD      CMD(SCAN) PARM1('Scanner="HP LJ100 M175 +
              Scan";Dest="C:\temp\scanC.pdf";Type="pdf"')
```

SELECTFILE command

Opens a dialog box for selecting a file.

The full path of the selected file is returned in the &RESULT variable.

If the user cancels the selection, the *CANCEL value is retrieved.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('SELECTFILE')
CHGVAR              VAR(&PARM1) VALUE('
                    [Pattern=Files to look for";]
                    [File="File name to look for";]
                    [Title="Window title";]
                    [Directory="Initial directory";]
                    [PathMustExist;]
                    [FileMustExist;]
                    [CreatePrompt;]
                    [OverWritePrompt;]
                    [NameOnly;]
                    [DftExt="Default file extension";]
                    [Save;]
                    [Folders;]
                    [Copy;]
                    ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Pattern** gives the list of files categories that can be selected. Each category is represented by a description text, followed by the character "=", then followed by the filters strings. The filter strings are separated from each other by a semi column. A double semi column separates the categories from each other. **Example :** **Pattern="Text files=*.doc ;*.dot ;*.txt ;;All files=*.*"** By default, LAUNCHER Office selects for Word files. **File** sets the initial file name to select. **Title** changes the Window title. **Directory** sets the initial directory.

PathMustExist : If set, the path to the selected file must exist.

FileMustExist : If set, the selected file must exist.

CreatePrompt : If the file does not exist , Windows will ask the user to confirm the file creation.

OverWritePrompt : If the file already exists, Windows will ask for the file replacement.

NameOnly to receive only the file name, without its full directory path.

DftExt sets the default extension that will be append to the file name if the user does not give one.

Save : This options will open the Windows dialog box to browse for a file to write.

Folders : To parse only the directories, set this option.

Copy will make LAUNCHER copy the selected path and file name to the clipboard. The User will be able to paste it in another application.

Note

When you enter a LAUNCHER Office CL command, like **LNCPRTDOC**, by pressing **F4** on a parameter containing a "File Path", the SELECTFILE command is called. The option **Copy** is set. This allows you to paste the path from your terminal emulation.

Example

```
CHGVAR          VAR(&CMD) VALUE('SELECTFILE')
CHGVAR          VAR(&PARM1) VALUE(' +
                Pattern="Text file=*.doc;*.dot;*.txt;;Excel
                files=*.xls;;All files=*.*"; +
                Directory="C:\My Documents";FileMustExist')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

See also

- CHKFILE
- LNCPRTDOC - CL Command

SETPRINTER command

This command allows the AS/400 program to:

- Change some properties of the designated printer.
- Configure a Windows printer with a previously saved configuration.
- Redirect the output of the printer to a file.
- Change the default printer.

Calling RSTPRINTER Command Cancels Changes Made by SETPRINTER

Syntax

```
CHGVAR          VAR(&CMD) VALUE('SETPRINTER')
CHGVAR          VAR(&PARM1) VALUE('Printer="Printer name"
                [ ; setdefault= True/False ]
                [ ; prnfile="Path to output file" ]
                [ ; configfile="Path to configuration file" ]
                [ ; trunc= True/False ]
                [ ; Source=Value ]
                [ ; Orientation=Value ]
                [ ; Duplex=Value ]
                [ ; Copies=Value ]
                ')

CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &RESULT)
```

Parameters

Parameters	
Parm1	**Printer =** Printer name. Use the name known from the Windows system. **Setdefault =** True/False. True : makes the printer as default printer. **Prnfile =** Full path to the output file to be generated. Do not use this option in a Windows TSE or CITRIX environment. **Configfile =** Full path to the printer configuration file. This configuration file was previously generated by the : Printer configuration file. **Trunc =** True/False True : The **Prnfile** is cleared.

False : The new data will be append to the **PrnFile**.

Orientation

Portrait = 1
Landscape = 2

Source

First = 1
Upper = 1
Onlyone = 1
Lower = 2
Middle = 3
Manual = 4
Envelope = 5
Envmanual = 6
Auto = 7
Tractor = 8
SmallFMT = 9
LargeFMT = 10
LargeCapacity = 11
Cassette = 14
FormSource = 15
Last = 15

Copies

Copies number to print.

Duplex

Simplex = 1
Vertical = 2
Horizontal = 3

Notes

- This command allows you to temporarily switch to a printer configuration that you have pre-registered in a file.

- When printing a mail merge, print to a file.

This speeds up the execution of mass mailings with a large amount of registration and helps bypass some Word bugs.

Notably the problems of Word with files taking up too much space in memory.

Printing in a file always concatenates the new data with the data already present in the file.

The Trunc option is used to empty the file when calling the SETPRINTER command. If multiple impressions are made, before calling RSTPRINTER, the data will be concatenated.

For example, you can redirect the output of the printer to a file (PRN file). Using the LAUNCHER IFSPUT command and the CL LNCPRTDOC command, you can move the PRN file to an AS400 ASQ.

Be careful, the use of printing in a file in TSE mode is strongly discouraged.

Example

This example prints the result of a Word mail merge into a text file (.PRN).

```
//Connect to LAUNCHER server
LNCOPEN
//Send the datasource to the PC
LNCXFER PCFILE('%LNCDIR%\samples\sp_cust.txt') FROMFILE(SP_CUST)
CLOSECOM(*NO)
//Change the printer configuration
LNCCMD CMD(SETPRINTER) PARM1('printer="HP LaserJet 1200 Series PCL
6";prnfile="%lncdir%\samples\test.prn";trunc=true;setdefault=true')
//Open Word and merge template and data
LNCCMD CMD(WORDOPEN)
LNCCMD CMD(WMAILMERGE) PARM1('document="%lncdir%\samples\sp_cust.doc"+
;destination=wdSendToPrinter;DataSource="%lncdir%\samples\sp_cust.txt"
')
//Restore printer configuration
LNCCMD CMD(RSTPRINTER) PARM1('printer="HP LaserJet 1200 Series PCL
6"')
```

Voir aussi

- RSTPRINTER
- WMAILMERGE

SHELL command

Allows to run DOS Commands, open documents or launch the execution of programs on the PC.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('SHELL')
CHGVAR          VAR(&PARM1) VALUE('DOC command | Document name')
CHGVAR          VAR(&PARM2) VALUE('
                Wait=True / False / Number
                ; Errorlevel=True / False
                ; Visible=True / False
                ; Focus=True / False
                ; Minimize= True / False
                ; OpenDocument=True / False
                ; Directory=Default directory"
                ; Action=open" | "explore" | "print"
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	DOS command to run on Windows station, (ex: COPY F:\Docs\Template.doc C:\Temp). Or, path to a document to open, or Internet URL to connect to.
Parm2	**Wait =** True / False / Number. Default = True. Application will wait until command processing ends, with no limit if Wait is True. If a value is given, the process will wait until the number of milliseconds has expired. **Errorlevel** = True / False. AS/400 program will receive error level as command call return. Error level returns in &RESULT parameter as 5 digits code. **Visible** = True / False. Default = False. Command processing may be hidden. **Focus** = True / False. Default = True. If **Focus** is false, the active windows remains unchanged. This is useful to display documents or images, and keep the terminal emulation always active.

Minimize =True / False. Default = False.
If **Minimize** is true, the active window at the time of execution of this command will be minimized.
If Wait is not set to 'False', this window will be restored when the command completes.

OpenDocument = True / False. Default = False.
If this option is true, &PARM1 contains a file to open name path, or an Internet URL.
Windows will decide on the appropriate application according to the indicated file type.

Directory = Sets default directory of command to process, or document to open.

Action = Indicates a particular action to apply to a document. "**OpenDocument=**" option is equivalent to **Action=**"**open**".
Action values are : ***"open", "print", "explore".***

Note

For some applications, like Internet Explorer, it is not possible to wait for the end of the command processing. Whatever you set for the **Wait** option, the command will be launched, and your process will continue immediately.

Examples

In this example a file is copied from a directory to another.

```
CHGVAR          VAR(&CMD) VALUE('SHELL')
CHGVAR          VAR(&PARM1) VALUE('COPY \Templates\Invoice.doc
                C:\Temp\Invoice.doc')
CHGVAR          VAR(&PARM2) VALUE('ErrorLevel')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

In this example a Windows station is connected to a Web site.

```
CHGVAR          VAR(&CMD) VALUE('SHELL')
CHGVAR          VAR(&PARM1) VALUE('http://www.easycom-aura.com’)
CHGVAR          VAR(&PARM2) VALUE('OpenDocument')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

In this example an HTML page is opened using the default navigator.

```
CHGVAR              VAR(&CMD) VALUE('SHELL')
CHGVAR              VAR(&PARM1) VALUE('\Pages\Document.htm')
CHGVAR              VAR(&PARM2) VALUE('OpenDocument')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

STO command

Stores the current window in memory, and returns its handle.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('STO')
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

The number of the active window is returned in 10 digits, in the &RESULT parameter.

This window can be restored by the RCL command.

See also

- MIN
- RCL

VAL command

Allows you to enter a value in one of the cells of the merge file created by the CRF command.

The entry of the value will always be in the current line.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('VAL')
CHGVAR          VAR(&PARM1) VALUE('column_name)
CHGVAR          VAR(&PARM2) VALUE('value')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Name of the column or number of the column (example: #2).
Parm2	Value to be entered.

Example

```
LNCOPEN

LNCCMD      CMD(CRF) PARM1('C:\A\fusion2.txt')

LNCCMD      CMD(NCL) PARM1('Nom')

LNCCMD      CMD(NCL) PARM1('Prenom')

LNCCMD      CMD(NCL) PARM1('Identifiant')

LNCCMD      CMD(NLN)

LNCCMD      CMD(VAL) PARM1('Nom') PARM2('Redfall')

LNCCMD      CMD(VAL) PARM1('Prenom') PARM2('Robert')

LNCCMD      CMD(VAL) PARM1('Identifiant') PARM2('1')

LNCCMD      CMD(ENF)

LNCCLOSE
```

In this example, the file 'fusion2.txt' is created in the directory "C: \ A".

See also

- CRF
- ENF
- NCL
- NLN

WADDINS command

Load a Microsoft Word add-in.

A complement is an additional program adding further customized commands or functions to Microsoft Word.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WADDINS')
CHGVAR          VAR(&PARM1) VALUE('Add-in')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Full path to the add-in to load.
Parm2	

Note

WADDINS may be used to add template including macros, making them available.

Example

Load Launcher add-in to create custom menu (WMENU command) :

```
CHGVAR      VAR(&CMD) VALUE('WADDINS')
CHGVAR      VAR(&PM1) VALUE('%LNCDIR%\LNCWordAddin.dot')
CHGVAR      VAR(&PM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HDL &CMD &OPT &PM1 &PM2 &RES)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Other example

Assign a style in a Word document by macro:

```
PGM
LNCOPEN
```

```
LNCCMD CMD(WORDOPEN)
LNCCMD CMD(WOPENFILE) +
PARM1('%LNCDIR%\SAMPLES\MODELE_STYLE.DOC')
LNCCMD CMD(WADDINS) PARM1('%LNCDIR%\SAMPLES\MODELE_MACRO.dot')
LNCCMD CMD(WMENU) +
PARM1('Captions="macro2";Tips="macro2"')
LNCCMD CMD(WORDWAIT)
LNCCMD CMD(WORDSHOW)
LNCCLOSE
ENDPGM
```

See also

- WMENU
- WORDWAIT
- WEXEMACRO
- WGETPROP

WBOOKMADD command

Adds a bookmark to the current cursor position.

Bookmark name must be Word compatible.
Current selection is identified with added bookmark name.
Using WBOOKMARK command, the program will find this particular position later on.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('WBOOKMADD')
CHGVAR        VAR(&PARM1) VALUE('
              Bookmark="Name of bookmark to insert"]
              [;Text="Text to insert" ]
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Bookmark :** Name of the bookmark to insert in the current document. **Text :** The value of the text to insert at the location of the created bookmark. Optional.

Example

```
DCL         VAR(&FILE) TYPE(*CHAR) LEN(2000)
DCL         VAR(&PATH) TYPE(*CHAR) LEN(2000)

CHGVAR      VAR(&PATH) VALUE('C:\A\')
CHGVAR      VAR(&FILE) VALUE('res.docx')

LNCOPEN     SVRADDR('*DEV')

LNCCMD      CMD(WORDOPEN)

LNCCMD      CMD(WOPENFILE) PARM1(&PATH *TCAT &FILE)

LNCCMD      CMD(WBOOKMADD) +
              PARM1('Bookmark="Name";Text="Client name"')

LNCCMD      CMD(WSAVEAS) +
              PARM1('FILE="C:\A\testBookmarkAdd.docx"')
```

```
LNCCMD      CMD(WORDCLOSE)

LNCCLOSE
```

See also

- WBOOKMARK

WBOOKMARK command

Inserts text at the specified bookmark location.

Syntax

```
CHGVAR          VAR(&CMD) VALUE(WBOOKMARK)
CHGVAR          VAR(&PARM1) VALUE('
                [ Bookmark="Nom de signet" ] +
                [ ;Start]
                [ ;End]
                [ ;Overtype]
                [ ;Adjust]
                [ ;Append] +
                [ ;InsertRow]
                [ ;Merge=n] +
                [ ;Unicode=True/False ] +
                [ ;ToBookmark="Nom de signet" ]
                [ ;NUMFMT(d DECPOINT=p GRPPOINT=g)] +
                [ ;PROP(Property)=Value]  +
                ')

CHGVAR          VAR(&PARM2) VALUE('Texte à insérer')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Bookmark** is the name of a bookmark present in the document, or a predefined bookmark of Microsoft Word, or a special value - see the notes below. A bookmark represents a position in the document, or a selection (range) of several characters. A bookmark is placed in a Word document by the "Insert" menu "Bookmark" of Word. If the bookmark name is missing in Parm1, or, if it is equal to ***NONE**, then the Word cursor is not moved, and the text will be inserted at the current position. If the bookmark specified here is missing from the document, no action is performed, and the word **'FALSE'** is returned in the &RESULT parameter. **Start :** When the bookmark is set to a selection, the **Start** option sets the cursor position at the beginning of the selection. **End :** When the bookmark is set to a selection, the **End** option sets the cursor position at the end of the selection.

	Overtype : the new text is typed over the existing text. By default the new text is inserted, or it replace the whole selection. **Adjust** will set the new location for the bookmark, on the inserted text. **Append** adds the new value to the current contents of the bookmark. **InsertRow** inserts a row into a table at the bookmark location. **Merge** : merges the n cells into one. **NUMFMT(d DECPOINT=p GRPPOINT=g)** to format a decimal digital value. **-d** decimal position number. **-p** decimal point character. **-g** thousand separator character. **PROP** sets the value of a property at the new cursor position. Multiple **PROP()** directives can be presents, separated by semicolons (;). Example : *PROP(Font.bold)=True;Prop(Font.italic)=True* **Unicode** : False by default. When Unicode is true, the text in &PARM2 must be in the Windows Unicode character set. API Program LNCCVTWCS helps you to convert to and from Unicode. Unicode is helpful to read and write foreign character sets, like Chinese, Korean, ... **ToBookmark** allows you to designate a bookmark. The selection will be extended to this bookmark, from the bookmark specified with the Bookmark parameter, or from the current position if Bookmark is not specified.
Parm2	**Text to insert.** If the text is absent (white), the cursor position is moved to the bookmark location, and no text is inserted. Special value **%NONE%** cleans up the actual value of the selection. If option **Unicode** is true, content of &PARM2 must be in Windows Unicode character set. And parameter **&OPT** must be set to **'W'**.
Opt	When option **Unicode** is true, **&OPT** must be set to **'W'**. **&PARM2** must be coded in the Windows Unicode character set.

Notes

1) Allowed bookmark name values :

- Bookmark name inserted in document template.
- Microsoft Word Bookmarks preset.
- ***RIGHT** to move to right cell.
- ***LEFT** to move to left cell.
- ***UP** to move to upper cell.
- ***DOWN** to move to lower cell.

If the bookmark does not exist, no text will be inserted.

2) If bookmark includes several selected characters, the whole selection will be replaced with the new value.

3) WBOOKMARK command may serve to delete all paragraphs included in the bookmark.

4) Bookmarks can overlay.

5) Bookmarks may be placed anywhere in a document, including headers, footers or watermarks.

6) Word fits display mode automatically.

7) In all cases, move take place before text inserting.

Example 1

```
CHGVAR VAR(&CMD) VALUE('WBOOKMARK')
CHGVAR VAR(&PARM1) VALUE('Bookmark="Bookmark1";NUMFMT(0)')
CHGVAR VAR(&PARM2) VALUE('0001234')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

Result : 1234

```
CHGVAR VAR(&CMD) VALUE('WBOOKMARK')
CHGVAR VAR(&PARM1) VALUE('Bookmark="Bookmark2";NUMFMT(2)')
CHGVAR VAR(&PARM2) VALUE('0001234')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
```

```
&PARM2 &RESULT)
```

Result : 12.34

Example 2

```
CHGVAR VAR(&CMD) VALUE('WORDOPEN')
CHGVAR VAR(&PARM1) VALUE('FILE="C:\TEMP\TEST.DOCX"')
CHGVAR VAR(&PARM2) VALUE('VISIBLE')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WBOOKMADD')
CHGVAR VAR(&PARM1) VALUE('Bookmark="S1"')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WBOOKMARK')
CHGVAR VAR(&PARM1) VALUE('Bookmark="S1";NUMFMT(2 DECPOINT=,
GRPPOINT=.)')
CHGVAR VAR(&PARM2) VALUE('12345678')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

Exemple 3

```
LNCCMD      CMD(WBOOKMADD) +
              PARM1('Bookmark="Name";Text="Client name"')

LNCCMD      CMD(WBOOKMARK) +
              PARM1('Bookmark="Name";Start;Overtype;Adjus+
              t;PROP(Font.bold)=True;Prop(Font.italic)=Tr+
              ue') PARM2('Aura Equipements')
```

RPG Example

If AS/400 sends value 12345678 in the following format :

NUMFMT(2 DECPOINT=, GRPPOINT=.)

Printed value is displayed as:

123.456,78

The following RPG example :

```
EVAL   LNC_PARM2=*BLANKS
EVAL   LNC_PARM1='*RIGHT NUMFMT(2)'
MOVEL   SUM   LNC_PARM2
CALL   'LNCCMD'
. . .
```

allows digital data on AS/400 side and formatted text zone on the document.

If *SUM* variable value is **12350** (data sent to LAUNCHER Office is shown as in UPDDTA, excepted for decimal point) with two figures as decimal positions. LAUNCHER Office will insert the value **123.50** in the document.

How to format a date?

There are no LAUNCHER Office commands for WORD to format dates.

The user must therefore use the command "CVTDAT" on the AS/400.

However, in the case of a mail merge, it is possible to define the date format in the WORD document for all "MERGEFIELD" fields.

See also

- Special values
- LNCWBM - API Program
- LNCCVTWCS

WBOOKMDEL command

Deletes one or all bookmarks in a document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WBOOKMDEL')
CHGVAR          VAR(&PARM1) VALUE('Bookmark="Bookmark name"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Bookmark:** name of the bookmark to delete in the current document, or "***ALL**" to delete all bookmarks into the document, or "**Pref ***" to delete bookmarks whose name begins with "Pref".

Note

This command is used if an application must insert another document template, or the same template with similar bookmark names, in the current document.

If the bookmark names of the inserted document already exist in the current document, then Word retains the existing bookmarks.

Example

```
CHGVAR          VAR(&CMD) VALUE('WBOOKMDEL')
CHGVAR          VAR(&PARM1) VALUE('*ALL')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)

CHGVAR          VAR(&CMD) VALUE('WINSERTF')
CHGVAR          VAR(&PARM1) VALUE('File="%MASTERS%\MyDocument.docx" +
                Bookmark ="\EndOfDoc"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

WCHDIR command

Change the current directory of Word.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WCHDIR')
CHGVAR          VAR(&PARM1) VALUE('Directory="Directory path"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Directory:** Specifies the directory path from which Word will look for documents or other files, when the full path will not be specified. This search path is also used by some Word directives: IncludeText, IncludePicture, ...

Example

```
CHGVAR          VAR(&CMD) VALUE('WCHDIR')
CHGVAR          VAR(&PARM1) VALUE('Directory ="%TEMP%"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

WCLOSEFILE command

Closes the current document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WCLOSEFILE')
CHGVAR          VAR(&PARM1) VALUE('[*YES|*NO]')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	This command closes the current document using one of these options : ***YES** : Saves the modifications and close the document. ***NO** : Closes the document without saving the modifications. **Default**: if the document has not been saved before closing, and if no option is choosen, a Word popup occurs in order to save the document.

Example

```
CHGVAR          VAR(&CMD) VALUE('WCLOSEFILE')
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

or

```
LNCCMD      CMD(WCLOSEFILE)
```

In this example, WORD will display a dialog box to confirm the saving operation. If the WORD window has been minimized, it will blink in the task bar. Restore the window to display the dialog box.

See also

- WOPENFILE
- WSAVEAS

WCOPY command

Copy the current selection or the contents of a bookmark to the clipboard or to another bookmark.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WCOPY')
CHGVAR          VAR(&PARM1) VALUE('
                [From="Bookmark source"];
                [To="Bookmark of destination"];
                [Count=Number]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**From** : Indicates document bookmark which content must be copied. If **From** is not defined, the current selection will be copied to the clipboard. **To** : Indicates document bookmark to copy to. If **To** is not defined, data can only be copied to the clipboard. **Count** : Indicates the copies number. This parameter is only valid if **To** is defined.

Examples

The following example copies the value 123 456.78 from "Line1" to the clipboard.

```
CHGVAR VAR(&CMD) VALUE('WORDOPEN')
CHGVAR VAR(&PARM1) VALUE('FILE ="C:\TEMP\TEST.DOC"')
CHGVAR VAR(&PARM2) VALUE('VISIBLE')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WBOOKMADD')
CHGVAR VAR(&PARM1) VALUE('Ligne1')
```

```
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WBOOKMARK')
CHGVAR VAR(&PARM1) VALUE('Ligne1;NUMFMT(2 DECPOINT=, GRPPOINT=.)')
CHGVAR VAR(&PARM2) VALUE('12345678')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WSELECT')
CHGVAR VAR(&PARM1) VALUE('*LINE')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WCOPY')
CHGVAR VAR(&PARM1) VALUE(' ')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

In the following example, "Ligne1" content will be copied 10 times to "Ligne2" position.

```
CHGVAR          VAR(&CMD) VALUE('WCOPY)
CHGVAR          VAR(&PARM1) VALUE('From ="Ligne1";To="Ligne2";Count=10')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

In the following example, the current selection is copied to the clipboard.

```
CHGVAR          VAR(&CMD) VALUE('WCOPY)
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WPASTE
- WSELECT

WDOCUMENT command

Opens WORD document.

This synthesis command is somewhat similar to WOPENFILE or WNEWFILE command with opportunity to include parameters equivalent to WORDSHOW/WORDHIDE and WMAXIMIZE/WMINIMIZE commands.

Word must be opened before with WORDOPEN command.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('WDOCUMENT')
CHGVAR        VAR(&PARM1) VALUE('File name' OU ' 'Template name')
CHGVAR        VAR(&PARM2) VALUE('
              [Directory="Directory path";]
              [Document="File name";]
              [Detach = True/False;]
              [Visible;]
              [ReadOnly;]
              [New;]
              [Minimize;]
              [Maximize;]
              [DOCPWD="Password for opening";]
              [WRTPWD="Password for modification";]
              ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Name of the file to open. It may be empty if a new document has to be created, or includes a template name, if NEW is specified as parameter 2.
Parm2	**Directory** This option allows you to specify a path to the directory where the document to be opened is located. This directory path is added in front of the path and the file name given in parameter 1. **Document** By the keyword "Document", LAUNCHER Office is asked to search for the document among those already opened by Word on the system. If the requested document is already open, it is made active, it becomes the current document for Word. No new file is opened, so the name of the file or template is

	ignored. **Detach** = True/False If "Detach" is true, Word detaches the data source from the current document or the new active document when the WDOCUMENT command is returned. By detaching the data source, the document is no longer a mail merge template for Word. The WMAILMERGE command will redo the attachment to a data source. When a document is opened and attached to a data source (Merge File), it is maintained by Word, and therefore not editable. You must detach the data source from the current document if you want to send new data to the merge file. **VISIBLE** =to show Word. **MINIMIZE**= window is reduced. **MAXIMIZE**= window is widened. **READONLY** = open for reading only. *Default value is False* **NEW** = allows to create a new document if Parm1 is empty. **DOCPWD** = opening password. Uppercase/lowercase must be respected if the document is password protected. **WRTPWD**= modification password. Must be filled if the document is modification protected. **NOTICE !!! Word must be shown to open a password protected document.**
RESULT	**Name of the document** When returning the WDOCUMENT command, the &RESULT parameter contains the name of the active document. For a document opened from a file, the name of the

	document is equal to the name of the file, without the path and without the suffix ".DOCX". If the document has just been created, its name was built by Word.

Example

Creates a new document in a widened and visible window.

```
CHGVAR        VAR(&CMD) VALUE('WDOCUMENT')
CHGVAR        VAR(&PARM1) VALUE(' ')
CHGVAR        VAR(&PARM2) VALUE('VISIBLE;MAXIMIZE;NEW')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)
MONMSG        MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Opens Rapport.docx document, reading only, in a reduced window.

```
CHGVAR        VAR(&CMD) VALUE('WDOCUMENT')
CHGVAR        VAR(&PARM1) VALUE('C:\temp\Rapport.docx')
CHGVAR        VAR(&PARM2) VALUE('VISIBLE;MINIMIZE;READONLY')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
MONMSG        MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WORDOPEN
- WNEWFILE

WDPROTECT command

Removes the protection from the document.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WDPROTECT')
CHGVAR              VAR(&PARM1)VALUE('Password')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The password specified to the WPROTECT command.
Parm2	

Example

Assuming a Word document has been protected by WPROTECT :

```
LNCCMD      CMD(WORDOPEN)

LNCCMD      CMD(WOPENFILE) PARM1('c:\A\resProtect.docx')

LNCCMD      CMD(WPROTECT) +
              PARM1('Level=wdAllowOnlyComments;PWD="Aura"')

LNCCMD      CMD(WSAVE)

LNCCMD      CMD(WORDCLOSE) PARM1('SAVE=FALSE')
```

To remove the protection, it will be necessary to use the following command:

```
LNCCMD      CMD(WDPROTECT) PARM1('Aura')
```

See also

- WPROTECT

WEXEMACRO command

Execute a Microsoft Word macro.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WEXEMACRO')
CHGVAR              VAR(&PARM1)VALUE('Macro Name')
CHGVAR              VAR(&PARM2)VALUE('[Macro Parameters]')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Name of the macro. Sometimes the full name of the macro is needed. Example: Template.Module.Macro.
Parm2	Parameters passed to the macro. Parameters must be within double quotes, separated with semicolon.

Example 1

```
CHGVAR      VAR(&CMD) VALUE('WEXEMACRO')
CHGVAR      VAR(&PARM1) VALUE('Macro1')
CHGVAR      VAR(&PARM2) VALUE('"12000";"FRANCE" ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
            &PARM2 &RESULT)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Exemple 2

Assuming we have the following macro with 2 parameters :

Sub InsertTextAtEndOfDocument(param1 as string, param2 as string)
ActiveDocument.Content.InsertAfter Text:= param1
ActiveDocument.Content.InsertAfter Text:= param2
End Sub

To run the macro, you can run the following command:

```
LNCCMD      CMD(WEXEMACRO) +
               PARM1('InsertTextAtEndOfDocument') +
               PARM2('"Aura";"Equipements"')
```

See also

- WADDINS
- WGETPROP

WFIELDS command

Inserts a Microsoft field in a document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WFIELDS')
CHGVAR          VAR(&PARM1) VALUE('
                Text="Expression to insert"
                [; Type=Value]
                [; Bookmark="Bookmark"]
                [; PreserveFormat=True/False]
                [; Update=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Text =** Additional text for the creation of the field. **Type** = Type of field to add. (Default = wdFieldTypeEmpty). See accepted values below. **Bookmark =** indicates an existing document bookmark, where new field must be inserted. **PreserveFormat =** allows to preserve field format when updating. **Update =** updates all document fields.

Note

Accepted values for the field type:

wdFieldAddin	*wdFieldAdvance*	*wdFieldAsk*
wdFieldAuthor	*wdFieldAutoNum*	*wdFieldAutoNumLegal*
wdFieldAutoNumOutline	*wdFieldAutoText*	*wdFieldAutoTextList*
wdFieldBarCode	*wdFieldComments*	*wdFieldCompare*

wdFieldCreateDate	*wdFieldData*	*wdFieldDatabase*
wdFieldDate	*wdFieldDDE*	*wdFieldDDEAuto*
wdFieldDocProperty	*wdFieldDocVariable*	*wdFieldEditTime*
wdFieldEmbed	*wdFieldEmpty*	*wdFieldExpression*
wdFieldFileName	*wdFieldFileSize*	*wdFieldFillIn*
wdFieldFootnoteRef	*wdFieldFormCheckBox*	*wdFieldFormDropDown*
wdFieldFormTextInput	*wdFieldFormula*	*wdFieldGlossary*
wdFieldGotoButton	*wdFieldHTMLActiveX*	*wdFieldHyperlink*
wdFieldIf	*wdFieldImport*	*wdFieldInclude*
wdFieldIncludePicture	*wdFieldIncludeText*	*wdFieldIndex*
wdFieldIndexEntry	*wdFieldInfo*	*wdFieldKeyWord*
wdFieldLastSavedBy	*wdFieldLink*	*wdFieldListNum*
wdFieldMacroButton	*wdFieldMergeField*	*wdFieldMergeRec*
wdFieldMergeSeq	*wdFieldNext*	*wdFieldNextIf*
wdFieldNoteRef	*wdFieldNumChars*	*wdFieldNumPages*
wdFieldNumWords	*wdFieldOCX*	*wdFieldPage*
wdFieldPageRef	*wdFieldPrint*	*wdFieldPrintDate*
wdFieldPrivate	*wdFieldQuote*	*wdFieldRef*
wdFieldRefDoc	*wdFieldRevisionNum*	*wdFieldSaveDate*
wdFieldSection	*wdFieldSectionPages*	*wdFieldSequence*
wdFieldSet	*wdFieldSkipIf*	*wdFieldStyleRef*
wdFieldSubject	*wdFieldSubscriber*	*wdFieldSymbol*
wdFieldTemplate	*wdFieldTime*	*wdFieldTitle*
wdFieldTOA	*wdFieldTOAEntry*	*wdFieldTOC*
wdFieldTOCEntry	*wdFieldUserAddress*	*wdFieldUserInitials*
wdFieldUserName		

Examples

This example adds the variable field "Variable1" and assigns it the value "Sir".

Quotation marks are doubled around the value.

```
CHGVAR          VAR(&CMD) VALUE('WFIELDS')
CHGVAR          VAR(&PARM1) VALUE('Text="SET Variable1 ""Sir""" ; +
                TYPE= WDFIELDDOCVARIABLE;BOOKMARK="\STARTOFDOC"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

In this example all document fields are updated.

```
CHGVAR          VAR(&CMD) VALUE('WFIELDS')
CHGVAR          VAR(&PARM1) VALUE('Update =True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Finally this example creates a bookmark and assigns a merge field named CUST_ID on this bookmark:

```
LNCCMD      CMD(WBOOKMADD) +
              PARM1('BOOKMARK="Name";TEXT="Aura"')

LNCCMD      CMD(WFIELDS) +
  PARM1('Text="CUST_ID";Type=wdFieldMergeField;Bookmark="Name"')
```

WFINDTEXT command

Searchs for text in the current document.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WFINDTEXT')
CHGVAR              VAR(&PARM1)VALUE('Text="Text to search"')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Text =**Text to be found between double quotes.
RESULT	If the text is found the ***TRUE** value is returned in the &RESULT parameter. If not the ***FALSE** value is returned.

Example

```
CHGVAR VAR(&CMD) VALUE('WORDOPEN')
CHGVAR VAR(&PARM1) VALUE('FILE ="C:\TEMP\TEST.DOC"')
CHGVAR VAR(&PARM2) VALUE('VISIBLE')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WFINDTEXT')
CHGVAR VAR(&PARM1) VALUE('Text="Aura"')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

Other example

```
LNCCMD CMD(WFINDTEXT) PARM1('Text="1234"')
```

WFORMFIELD command

Allows you to programmatically change the value of a field on a Microsoft Word form.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WFORMFIELD')
CHGVAR          VAR(&PARM1) VALUE('
                Field="Bookmark name"
                [;NUMFMT(d DECPOINT=p GRPPOINT=g)]
                [;Clear=True/False]
                [;AddEntries=True/False]
                [;Value=Indice_de_liste]
                [;SetText=True/False]
                [;GetValue=True/False]
                [;GetText=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE('[text]')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Field :** bookmark name from current document. **NUMFMT(d DECPOINT=p GRPPOINT=** allows to format a decimal digital value. **-d** is the decimal position number. **-p** indicates which character to use as decimal separator. **-g** indicates which character to use as thousand separator. **Clear :** erase the contents of the field before affecting new value. If the field is a drop-down list, all the entries are erased. **AddEntries :** add entries in a drop-down list. Parameter 2 (&PARM2) contains the list of the values, between double quotes, and separated by semicolon. **Value :** change the index of the value in progress in a drop-down list, or in a chek box. If **SetText** is true, then the contents of parameter 2 will be regarded as a text to assign to the field, and not as the index of the value in progress in a list.

	If **SetText** is false, then the contents of parameter 2 will be regarded as an index, if it is numerical. When **GetValue** is true, the current index of a drop-down list is returned in the &RESULT parameter. When **GetText** is true, the value of the field is returned in the &RESULT parameter.
Parm2	Text to insert, or list of entries to add to a drop-down list.

Example

The following example gives to field "FF01" the value of variable **&FIELDVAL**.

```
CHGVAR          VAR(&CMD) VALUE('WFORMFIELD)
CHGVAR          VAR(&PARM1) VALUE('Field ="FF01";SetText')
CHGVAR          VAR(&PARM2) VALUE(&FIELDVAL)
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
```

The following example fixes the entries of a drop-down list.

```
CHGVAR          VAR(&CMD) VALUE('WFORMFIELD)
CHGVAR          VAR(&PARM1) VALUE('Field ="FF02";AddEntries')
CHGVAR          VAR(&PARM2) VALUE('"France";"Italie";"Espagne"')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
```

The example recovers the value text of a field of form, then erases it.

```
CHGVAR          VAR(&CMD) VALUE('WFORMFIELD)
CHGVAR          VAR(&PARM1) VALUE('Field="FF03";GetText;Clear')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
```

WGETHPOS command

Retrieves the cursor horizontal position from the left edge of the page.

This is the distance from the cursor to the left edge of the page measured in points (1 point = 20 twips, 72 points = 1 inch).

The result is sent through the &RESULT variable.

Syntax

```
CHGVAR                 VAR(&CMD)VALUE('WGETHPOS')
CHGVAR                 VAR(&PARM1)VALUE(' ')
CHGVAR                 VAR(&PARM2)VALUE(' ')
CALL                   PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                       &PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

WGETPAGE command

Retrieves the active page number of the Microsoft Word document currently in use.

The result is retrieved via the &RESULT variable.

Syntax

```
CHGVAR                VAR(&CMD)VALUE('WGETPAGE')
CHGVAR                VAR(&PARM1)VALUE(' ')
CHGVAR                VAR(&PARM2)VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                      &PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Example

```
CHGVAR VAR(&CMD) VALUE('WORDOPEN')
CHGVAR VAR(&PARM1) VALUE('C:\TEMP\FORMULAIRE.DOCX')
CHGVAR VAR(&PARM2) VALUE('VISIBLE')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WGETPAGE')
CHGVAR VAR(&PARM1) VALUE(' ')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

WGETPROP command

Reads the value of a property of Word.

Word and Excel use the same Visual Basic Application objects. All objects and their properties can be displayed with Visual Basic object explorer (Tools – Macro – Visual Basic Editor (or Alt-F11) and Display – Object Explorer (or F2).

Syntax 1

```
CHGVAR          VAR(&CMD) VALUE('WGETPROP')
CHGVAR          VAR(&PARM1) VALUE('property path')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Full path of the Word property. The syntax of the path to the property follows the same rules as for XLGETPROP. The value of the property is returned as a string in the &RESULT variable.
RESULT	Value of the property specified in Parm1.

Example

```
CHGVAR                  VAR(&CMD) VALUE('WGETPROP')
CHGVAR                  VAR(&PARM1) VALUE('Selection.Font.Color')
CALL                    PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
```

Syntax 2

Use of the **Property** parameter. Necessary if you use webservices.

```
CHGVAR          VAR(&CMD) VALUE('WGETPROP')
CHGVAR          VAR(&PARM1) VALUE('Property="Path_property"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Property**= Full path of the Word property. The syntax of the path to the property follows the same rules as for XLGETPROP. The value of the property is returned as a string in the &RESULT variable.
RESULT	Value of the property specified with the **Property** parameter.

Example

```
CHGVAR          VAR(&CMD) VALUE('WGETPROP')
CHGVAR          VAR(&PARM1)
                VALUE('Property="Selection.Font.Color"')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

See also

- WSETPROP

WGETTEXT command

Allows to retrieve the content of:

- **the current selection**
- **a bookmark**
- **a form field**

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WGETTEXT')
CHGVAR          VAR(&PARM1) VALUE('
                [Bookmark="Bookmark name"]
                [;Variable="Variable name"]
                [;Field="Form field name"]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**Bookmark**: name of the bookmark whose value we want to retrieve. **Variable**: name of the variable whose value we want to retrieve. **Field**: name of the form field whose value we want to retrieve. **If Parm1 is empty, the current selection is retrieved**.
RESULT	Content of the specified item.

Example

Retrieve the value of a bookmark:

```
CHGVAR VAR(&CMD) VALUE('WORDOPEN')
CHGVAR VAR(&PARM1) VALUE('C:\TEMP\FORM.DOCX')
CHGVAR VAR(&PARM2) VALUE('VISIBLE')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
```

```
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WGETTEXT')
CHGVAR VAR(&PARM1) VALUE('Bookmark="FF03"')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

See also

- WSELECT

WGETVPOS command

Retrieves the cursor vertical position from the top of the page.

This is the distance from the cursor to the top edge of the page measured in points (1 point = 20 twips, 72 points = 1 inch).

The result is sent through the &RESULT variable.

Syntax

```
CHGVAR                VAR(&CMD)VALUE('WGETVPOS')
CHGVAR                VAR(&PARM1)VALUE(' ')
CHGVAR                VAR(&PARM2)VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                      &PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

WGOTOTBL command

Moves the cursor to the next table in the Word active document.

Syntax

```
CHGVAR                 VAR(&CMD)VALUE('WGOTOTBL')
CHGVAR                 VAR(&PARM1)VALUE(' ')
CHGVAR                 VAR(&PARM2)VALUE(' ')
CALL                   PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                       &PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

WINSERTOBJ command

Inserts an OLE object into the active Word document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WINSERTOBJ')
CHGVAR          VAR(&PARM1) VALUE('
                File="Object path";
                [;Bookmark="Bookmark"]
                [;Link=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**File** : Full path to file to insert. **Bookmark** : (Optional) Bookmark name where new document will be inserted. Preset Word bookmarks can be used. **Link** : True/False, default is false. If **Link** is true, it is a link to the object that is inserted. Changes made to the object will automatically be reflected on the documents with which it is linked.

Example

Insert PowerPoint slides into a Word document:

```
LNCCMD CMD(WINSERTOBJ) PARM1('File="C:\temp\stored_proc.ppt"')
```

WINSERTBRK command

Inserts page, column or section break into a Word document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WINSERTBRK')
CHGVAR          VAR(&PARM1) VALUE('[Break=Constant]')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Break** = type of break to insert (Default= wdPageBreak). Possible values are : *wdPageBreak, wdColumnBreak, wdSectionBreakNextPage, wdSectionBreakContinuous, wdSectionBreakEvenPage, wdSectionBreakOddPage, wdLineBreak, wdLineBreakClearLeft, wdLineBreakClearRight, wdTextWrappingBreak.*

Example

```
CHGVAR          VAR(&CMD) VALUE('WINSERTBRK')
CHGVAR          VAR(&PARM1) VALUE('Break=wdColumnBreak')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

WINSERTCOL command

Inserts one or more columns to the left in a Word table.

The cursor must be inside a table.

The columns are inserted to the left of the column where the cursor is.

Syntax

```
CHGVAR                VAR(&CMD)VALUE('WINSERTCOL')
CHGVAR                VAR(&PARM1)VALUE('[NbCols]')
CHGVAR                VAR(&PARM2)VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                      &PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	Number of column to be inserted. 1 column is the default value.

Example

```
CHGVAR        VAR(&CMD) VALUE('WINSERTCOL')
CHGVAR        VAR(&PARM1) VALUE('4')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
MONMSG        MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WINSERTROW
- WINSERTIMG

WINSERTF command

Inserts an existing Word document into the active Word document.

The text formatting of the inserted document remains the same.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WINSERTF')
CHGVAR          VAR(&PARM1) VALUE('
                File="Document name"
                [;Directory="Directory path"]
                [;Bookmark="Signet"]
                [;Adjust=True/False]
                [;Append=True/False]
                [;range="range"]
                [;Link=True/False]
                [;DeleteBookmark=True/False/"Nom"]
                [;AsText=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**File** : Name of the file to insert. If the full path is specified, then the **Directory** parameter is not useful. **Directory**: (Optional) This option specifies a path to the directory where the document to be inserted is located. This directory path is added in front of the file name specified with the **File** option. This parameter is not needed if the full path is specified in the **File** parameter. **Bookmark** : (Optional) Bookmark name where new document will be inserted. Word preset bookmarks can be used. **Adjust**: (Optional) If bookmarked positioning is requested by the **Bookmark** option, this option places the bookmark on all inserted text. **Append**: (Optional) If bookmark positioning is requested by the **Bookmark** option, this option adds the contents of the document to be inserted to the current contents of the bookmark.

Link = True/False, default is false.

If **Link** is true, it is a link towards the file which is inserted. The modifications made to the file will be automatically deferred on the documents with which it is dependent.

Range : Range to insert from the source document. If the file to be inserted is a Word document, **Range** can represent a bookmark. If the file is an Excel document, **Range** can represent a range of cells

DeleteBookmark : If this option has the true value, then the bookmark specified by the option **Bookmark** is removed.

If this option contains a alpha numerical value, then the bookmarks will be removed as described by the order WBOOKMDEL.

If the inserted document itself contains bookmarks, they will be added to the current document, if their names do not already exist.

If **AsText** is true, then the file to be inserted must be text only. It will be inserted into the document as if it were typed. The format of characters and paragraphs will be the one applied at the insertion point.

Examples

Inserting a Word document at the end of the active Word document:

```
CHGVAR          VAR(&CMD) VALUE('WINSERTF')
CHGVAR          VAR(&PARM1) VALUE('File="c:\temp\lettre.docx";
                Bookmark="\ENDOFDOC"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Inserting a Word document at the location of the "State" bookmark:

```
LNCCMD      CMD(WINSERTF) +
              PARM1('FILE="C:\temp\etat.docx";BOOKMARK="State"')
```

WINSERTIMG command

Inserts picture into the current Word document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WINSERTIMG')
CHGVAR          VAR(&PARM1) VALUE('
                File="Picture file name"
                [;Directory="Directory path"]
                [;Bookmark="Bookmark"]
                [;Adjust=True/False]
                [;Append=True/False]
                [;Link=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**File** : Name of the picture file to insert. If the full path is specified, then the **Directory** parameter is not useful. **Directory**: (Optional) This option specifies a path to the directory where the file to be inserted is located. This directory path is added in front of the file name specified with the **File** option. This parameter is not needed if the full path is specified in the **File** parameter. **Bookmark** : (Optional) Bookmark name where picture will be inserted. Word preset bookmarks can be used. **Adjust**: (Optional) If bookmarked positioning is requested by the **Bookmark** option, this option places the bookmark on the inserted picture. **Append**: (Optional) If bookmark positioning is requested by the **Bookmark** option, this option adds the picture to be inserted to the current contents of the bookmark. **Link** = True/False, default is false. If **Link** is true, it is a link towards the picture which is inserted. The modifications made to the picture will be automatically deferred on

the documents with which it is dependent.

Example

```
LNCCMD CMD(WORDOPEN)

LNCCMD CMD(WOPENFILE) +
PARM1('%LNCDIR%\SAMPLES\MODELE_STYLE.DOCX')

LNCCMD CMD(WINSERTIMG) +
PARM1('FILE="%LNCDIR%\SAMPLES\LNC_Ofc.jpg";+
bookmark="image";link=true')
```

Other example

```
 LNCCMD     CMD(WINSERTIMG) +
               PARM1('FILE="C:\A\GRAPHE_IMMO.JPG";BOO+
               KMARK="GRAPHE_IMMO"')
```

See also

- WINSERTF

WINSERTROW command

Inserts one or more lines into a Word table.

The cursor must be in a table in the document.
Lines are inserted above the cursor line.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('WINSERTROW')
CHGVAR        VAR(&PARM1) VALUE('
              [Count=Rows number]
              [;MoveStart=True/False]
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
MONMSG        MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**Count :** Number of lines to insert. Default is 1 line. **MoveStart** : If True, current selection will be moved to inserted lines start. Default, current selection remains on the current line (line below inserted lines).

Example

```
CHGVAR        VAR(&CMD) VALUE('WINSERTROW')
CHGVAR        VAR(&PARM1) VALUE('Count=20')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
MONMSG        MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WINSERTCOL

WINSERTTBL command

Inserts a table in the active Word document.

The table can be inserted at the current position or at a bookmark position.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WINSERTTBL')
CHGVAR          VAR(&PARM1) VALUE('
                Columns=Columns number
                ;Rows=Rows number
                [;Bookmark="Bookmark name"]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**Columns** = number of columns in the table. **Rows** = number of rows in the table. **Bookmark** = name of the bookmark where the table will be inserted. By default the table is inserted at the current position.

Example

```
CHGVAR          VAR(&CMD) VALUE('WINSERTTBL')
CHGVAR          VAR(&PARM1) VALUE('Bookmark="table";Rows=5;Columns=6')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WGOTOTBL
- WINSERTROW
- WINSERTCOL

WMAILMERGE command

Performs a merge between the Word template and the data file.

This command uses Word merge fields.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WMAILMERGE')
CHGVAR          VAR(&PARM1) VALUE('
                [Document="model name"]
                [;Destination=constant]
                [;SaveAs="saved document name"]
                [;SaveAsFmt=constant]
                [;DataSource="data source"]
                [;HeaderSource="headers file"]
                [;FirstRecord= value]
                [;LastRecord= value]
                [;SuppressBlankLines=True/False]
                [;SourceFormat= value]
                [;HeaderFormat= value]
                [;Type=constant]
                [;Execute=True/False ]
                [;MailAddress="field for recipient address"]
                [;MailSubject="message subject"]
                [;AsAttachment=True/False]
                [;OneDoc=True/False]
                [;CleanData=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**Document**: name of the main document, if this parameter is empty the current document will be considered as main document. If it has been initialized by a first merge operation, it is retained. **Destination**: destination of the merge, can be one of the following constants: **wdSendToEmail**, for sending by email. Fill in the **MailAddress** parameter. **wdSendToFax**, for sending by fax. **wdSendToNewDocument**, to create a new document

(default value). Fill in the **SaveAS** parameter.

wdSendToPrinter, for sending to the printer.

IncSendToFile, to save each letter to an individual file. The **SaveAs** parameter must then contain a path and rules for constructing a unique filename for each letter.

SaveAs: Path and name of the file where to save the result of the merge.
If **Destination** = **IncSendToFile**, or **OneDoc=true**, then the file name specified by the **SaveAs** parameter must be constructed from field values of the data source.

SaveFmt: wdFormatXMLDocument / wdFormatDocument / wdFormatDOSText / wdFormatDOSTextLineBreaks / wdFormatEncodedText / wdFormatHTML / wdFormatRTF / wdFormatTemplate / wdFormatText / wdFormatTextLineBreaks / wdFormatUnicodeText.

If this parameter is not specified, the default Word format is used.
For example, for Word 2019, this is the format wdFormatXMLDocument (DOCX).
To save in **PDF** format (from Word 2010) we can use the value **17**.

DataSource: name of the source file. This parameter is not required if the main document already contains the name of the associated data source.

HeaderSource : name of the file that contains the headers. Allows you to specify a separate source that contains the name of the column headers (and which appear in the main document as merge fields).

FirstRecord: Number of the first record of the data source to use for the mail merge.
If this parameter is not specified, the first record of the data source is taken into account.

LastRecord: The number of the last record of the data source to use for the mail merge.
If this parameter is not specified, the last record of the data source is taken into account.

SuppressBlankLines, indicates whether to keep empty lines or not, after the merge. Default: true.

SourceFormat, **HeaderFormat** indicate the format of the source file and the header file, respectively.
The possible values are:

wdOpenFormatAllWord, wdOpenFormatAuto,

wdOpenFormatDocument, wdOpenFormatEncodedText, wdOpenFormatRTF, wdOpenFormatTemplate, wdOpenFormatText, wdOpenFormatUnicodeText, or wdOpenFormatWebPages.
The default value is: *wdOpenFormatAuto*.

Type : type of merge document. The possible values are: *wdCatalog, wdEnvelopes, wdFormLetters (default), wdMailingLabels, wdNotAMergeDocument.*

Execute (Default execute = true): If "execute = false", the document is simply prepared for a merge, but not merged.

Onedoc = true equals "**Destination = IncSendToFile**", to save each letter to an individual file. The **SaveAs** parameter must then contain a path and rules for constructing a unique filename for each letter. Default: false.

CleanData: (Default: False): If True, removes the following non-printable characters from the data file: NUL, SI, and CR (not CRLF).

If the destination is an email (**wdSendToEmail**), then the following properties can be set:

MailAddress: Refers to the field of the DataSource file containing the recipient's address.

MailSubject: Subject of the email message.

AsAttachment: indicates that the document should be attached to the message. Default: false.

Notes

1) No parameter is required.

2) If the main document has been prepared for a merge, it contains the name of the data source and the destination, it can then be opened by the WDOCUMENT command before using the WMAILMERGE command without any parameters.

The main document type allows you to modify the characteristics of the merge, the different types are: standard letters, labels, envelopes, catalog (no page breaks between two recordings). It must be defined when preparing the main document.

If a first operation has initialized the values of the main document, the data source and the destination, we can restart a WMAILMERGE which will take all

these parameters. For example, to merge only a particular record or range of records.

3)

The "Execute = False" property allows you to programmatically prepare a merge document, attaching the data source to it, without executing it.

Once the fusion document is prepared, possibly empty, the program can present it to the user (WORDSHOW) so that the latter can compose it.

4)

Construct a file name from a field in the data source:

When **Destination = lncSendToFile** is specified, or when **OneDoc = true**, each letter is saved to a file. The path and file name given by the **SaveAs** keyword can contain references to fields in the data source. The field to which reference is made must be between characters <and>.

Example:
Destination=lncSendToFile; SaveAs="D:\ Letters\ Prop_<CUST_ID>.DOCX";

In this example, each letter will be saved in the sub-directory "Letters", and the name will be composed of the word "Prop_" followed by the value of the field "CUST_ID".

Example

```
             VAR(&CMD) VALUE('WMAILMERGE')
CHGVAR       VAR(&PARM1) VALUE('Document="C:\chemin\maitre.docx" ;+
             Destination="wdSendToEmail";+
             MailAddress="tech@easycom-aura.com"')
CHGVAR       VAR(&PARM2) VALUE(' ')
CALL         PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
             &PARM2 &RESULT)
MONMSG       MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Merge the main document "Master.docx" with the data source that is already associated with it and send the result by mail to AURA Equipements technical service.

```
CHGVAR              VAR(&CMD) VALUE('WMAILMERGE')
CHGVAR              VAR(&PARM1) VALUE('Document="C:\chemin\maitre.docx" ;+
                    DataSource="C:\chemin\SPCUST.txt";+
                    SaveAs="result_5_6.docx";FirstRecord=5;+
```

```
                    LastRecord=6')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

CHGVAR              VAR(&CMD) VALUE('WMAILMERGE')
CHGVAR              VAR(&PARM1) VALUE('SaveAS="Result_17_20.docx"+
                    FirstRecord=17;LastRecord=20')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Prepares the merging of the spcust.txt data source with the main document, merges the records 5 and 6 to the result_5_6.docx document and then merges, with the same elements, the records 17 to 20.

```
LNCCMD      CMD(WMAILMERGE) +
             PARM1('DOCUMENT="C:\temp\test\+
             sp_cust.dotx";DATASOURCE="C:\temp\test+
             \csv2.TXT";SAVEAS="C:\temp\res\+
             res_<CUST_ID>.pdf";OneDoc=true;SaveFmt=17')
```

Executes the merge using the specified model and data source. A PDF will be generated (based on the template) for each record.

See also

- LNCPRTDOC - Commande CL
- DBXFER
- Create a merge file

WMAXIMIZE command

Maximizes the active window of Word.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WMAXIMIZE')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                    &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WMINIMIZE

WMENU command

Creates a custom menu bar in Word.

WMENU must be used with **WADDINS** and **WORDWAIT** commands.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('WMENU')
CHGVAR        VAR(&PARM1) VALUE('
              Captions="Buttons labels"
              [;Tips="Tooltips"]
              [;Remove]
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Captions :** Lets you label each button of the new menu bar. The labels are separated from each other by a semicolon (;). The number of labels determines the number of buttons on the menu bar. **Tips :** Displays a tooltip for each button. The tips are separated from each other by a semicolon (;). Their number must match the number of labels. **Remove :** removes the previous menu.

Notes

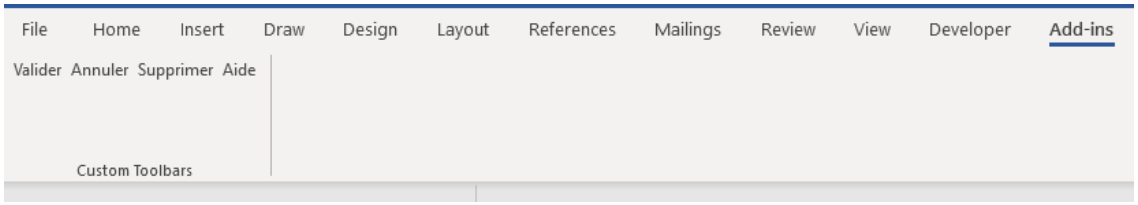
1)

The add-in "LNCWordAddin.dot" must be loaded by the **WADDINS** command before using **WMENU**.

When the **WORDWAIT** command is called, the AS/400 program waits for an action on the part of the user. He takes again the hand when Word is closed, or on the action of one of the buttons of the new menu.

When returning from the **WORDWAIT** command, the &RESULT variable contains the 5-digit number of the actuated button (from 1 to n).

2)
The custom menu will appear into the "add-in" tab of the Word menu.



Example

```
CHGVAR        VAR(&CMD) VALUE('WADDINS')
CHGVAR        VAR(&PARM1) VALUE('%LNCDIR%\ LNCWordAddin.dot')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)

CHGVAR        VAR(&CMD) VALUE('WMENU')
CHGVAR        VAR(&PARM1) VALUE('Captions="To validate;to cancel";Tips="to
              validate the document;to give up the document"')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)

CHGVAR        VAR(&CMD) VALUE('WORDWAIT')
CHGVAR        VAR(&PARM1) VALUE(' ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

WMERGECELL command

Merges several cells in a Word table.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WMERGECELL')
CHGVAR          VAR(&PARM1) VALUE('
                Cols=Columns number
                ;Rows=Rows number
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**Cols** : Number of columns to select from the current position. **Rows** : Number of rows to select from the current position.

Example

```
CHGVAR          VAR(&CMD) VALUE('WMERGECELL')
CHGVAR          VAR(&PARM1) VALUE('cols=4;rows=2')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WSPLITCELL

WMETHOD command

Calls method from Word.Application object.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WMETHOD')
CHGVAR          VAR(&PARM1) VALUE('Method')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	Full path of the method to call. Example : Documents.Item("Document2").Activate
Parm2	

Note 1

The **WMETHOD** command is used to execute methods of the Word VBA language from the AS/400.

The contents of Parm1 respect the syntax used in Visual Basic.

It is possible to use all the constants of Word and Visual Basic Application or the value of the constant itself.

To get an idea of the methods to call, go to Word in "Macro Recording" mode: **"Tools" menu - "Macro" - "New Macro".**

Do the desired operations on the keyboard and mouse.

Stop the macro recording, and see the code generated by Word: **Menu "Tools" - "Macro" - "Macros" - "Edit".**

Note 2

There are 2 differences to note between the VB syntax of Word and the LAUNCHER syntax:

1) When selecting an object from a collection, add ".Item" after the name of the collection.

```
Documents("Document2").Activate
```

becomes

```
Documents.Item("Document2").Activate
```

2) The parameter names (Name: = value) are specific to the VB syntax.

With LAUNCHER, it is necessary to list the values of each parameter, in the expected order, separated by ';'
The following statement

```
ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, Address:=
_
        "http://www.easycom-aura.com", SubAddress:="",
ScreenTip:="", _
        TextToDisplay:="fffff"
```

becomes

```
ActiveDocument.Hyperlinks.Add(prop(Selection.Range) ;
http://www.easycom-aura.com; "" ; "" ; "Aura")
```

And so the command is:

```
LNCCMD CMD(WMETHOD) +
PARM1('ActiveDocument.Hyperlinks.Add(prop(Selection.Range);+
"http://www.easycom-aura.com"; ""; ""; "Aura")')
```

Otherwise, choose to write a macro in Word, which you will call by WEXEMACRO.

Example

This example activates the "Document2" document.

```
CHGVAR          VAR(&CMD) VALUE('WMETHOD')
CHGVAR          VAR(&PARM1) VALUE('Documents.Item("Document2").Activate')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
```

WMINIMIZE command

Minimizes the Word window.

Syntax

```
CHGVAR                VAR(&CMD)VALUE('WMINIMIZE')
CHGVAR                VAR(&PARM1)VALUE(' ')
CHGVAR                VAR(&PARM2)VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                      &PARM1 &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WMAXIMIZE

WMOVE command

Moves or extends the current selection.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WMOVE')
CHGVAR          VAR(&PARM1) VALUE('
                [From="Bookmark"];
                [Direction="Direction"];
                [Unit=move unit];
                [Count=move units number];
                [While="Lists of characters"];
                [Until="Lists of characters"];
                [Extend=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**From** : Indicates the document bookmark from which selection will start. If « From » is omitted, selection start remains unchanged. **Direction** : Indicates the current selection move direction. Move is made according to « Unit » property. Possible values are : Start : Move selection to unit start. End : Move selection to the end. Down : Move selection downward. Up : Move selection upward. Right : Move selection to right. Left : Move selection to left. **See details part to use correctly others parameters (above) according to the Direction parameter.** **Unit** : Sets move unit. Possible values are : wdCharacter, wdCell, wdWord, wdSentence, wdParagraph, wdSection, wdColumn, wdRow, wdTable, wdLine. **Count** : Indicates move unit number. This number can be positive or negative. **While** : Selection is moved as long as a list character is present.

Until : Selection is moved until a list character is met.

Extend : Value must be « True » to extend selection, otherwise current selection is replaced.

Details

Direction parameter	Usable parameters
not used	**while** Moves the specified selection while any of the specified characters are found in the document. **[count]** default: **wdForward**. The maximum number of characters by which the specified selection is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If Count is a positive number, the specified selection is moved forward in the document, beginning at the end position. If it is a negative number, the selection is moved backward, beginning at the start position. Example 1.
	until Moves the specified selection until one of the specified characters is found in the document. **[count]** default: **wdForward**. The maximum number of characters by which the specified selection is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If Count is a positive number, the selection is moved forward in the document, beginning at the end position. If it is a negative number, the selection is moved backward, beginning at the start position. Example 2.
	Collapses the specified selection to its start or end position and then moves the collapsed object by the specified number of units. This method returns a **Long** value that represents the number of units by which the selection was moved, or it returns 0 (zero) if the move was unsuccessful.

	[**unit]** default: **wdCharacter** The unit by which to move the ending character position. **[count]** default: **1**. The number of units by which the specified range or selection is to be moved. If Count is a positive number, the object is collapsed to its end position and moved backward in the document by the specified number of units. If Count is a negative number, the object is collapsed to its start position and moved forward by the specified number of units. Example 3.
start	**while** Moves the start position of the specified selection while any of the specified characters are found in the document. **[count]** The maximum number of characters by which the specified range is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If Count is a positive number, the range is moved forward in the document. If it is a negative number, the range is moved backward. The default value is **wdForward** . Example 4.
	until Moves the start position of the specified selection until one of the specified characters is found in the document. If the movement is backward through the document, the selection is expanded. **[count]** The maximum number of characters by which the specified range is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If Count is a positive number, the range is moved forward in the document. If it is a negative number, the range is moved backward. The default value is **wdForward** . Example 5.
	Moves the start position of the specified selection. **[unit]** The unit by which start position of the specified selection is to be moved. Can be one of the **WdUnits** constants. The

	default value is **wdCharacter**. **[count]** The maximum number of units by which the specified range is to be moved. If Count is a positive number, the start position of the range is moved forward in the document. If it is a negative number, the start position is moved backward. If the start position is moved forward to a position beyond the end position, the range is collapsed and both the start and end positions are moved together. The default value is 1. Example 6
End	**While** Moves the ending character position of a range while any of the specified characters are found in the document. **[count]** The maximum number of characters by which the range is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If Count is a positive number, the range is moved forward in the document. If it is a negative number, the range is moved backward. The default value is **wdForward** . Example 7.
	until Moves the end position of the specified selection until any of the specified characters are found in the document. **[count]** The maximum number of characters by which the specified selection is to be moved. Can be a number or either **wdForward**or **wdBackward**. If Count is a positive number, the selection is moved forward in the document. If it is a negative number, the selection is moved backward. The default value is **wdForward**. Example 8.
	Moves the ending character position of a range or selection. **[unit]** The unit by which to move the ending character position. The default value is **wdCharacter**. **[count]** The number of units to move. If this number is positive, the ending character position is moved forward in the

	document. If this number is negative, the end is moved backward. If the ending position overtakes the starting position, the range collapses and both character positions move together. The default value is 1. Example 9.
down Moves the selection down and returns the number of units it has been moved.	**[unit]** The unit by which the selection is to be moved. The default value is **wdLine** . **[count]** The number of units the selection is to be moved. The default value is 1. **[extend].** Can be either **false** or **true** . If **false** is used, the selection is collapsed to the endpoint and moved down. If **true** is used, the selection is extended down. The default value is **false** . Example 10.
up Moves the selection up and returns the number of units that it has been moved.	**[unit]** Can be one of the following constants: wdLine , wdParagraph , wdWindow or wdScreen . The default value is wdLine . You can use the wdWindow constant for the Unit argument to move to the top or bottom of the active window. Regardless of the value of Count (greater than 1 or less than 1), the wdWindow constant moves only one unit. Use the wdScreen constant to move more than one screen. **[count]** The number of units the selection is to be moved. The default value is 1. **[extend]** Specifies whether the selection is moved or extended. Can be either **false** or **true** . If **false** is used, the selection is collapsed to the endpoint and moved up. If **true** is used, the selection is extended up. The default value is **false** . Example 11.
right Moves the selection to the right and returns the number of units it has been moved.	**[unit]** The unit by which the selection is to be moved.The default value is **wdCharacter** . **[count]** The number of units the selection is to be moved. The default value is 1.

	[extend] Can be either **false** or **true** . If **false** is used, the selection is collapsed to the endpoint and moved to the right. If **true** is used, the selection is extended to the right. The default value is **false** Example 12.
left Moves the selection to the left and returns the number of units it has been moved.	**[unit]** The unit by which the selection is to be moved.The default value is **wdCharacter** . **[count]** The number of units the selection is to be moved. The default value is 1. **[extend]** Can be either **false** or **true** . If **false** is used, the selection is collapsed to the endpoint and moved to the left. If **true** is used, the selection is extended to the left. The default value is **false** . Example 13.

Example 1:

a)

GOOD MORNING CEDRICK. AURA

LNCCMD CMD(WMOVE) PARM1('While="Z"') : result = 0

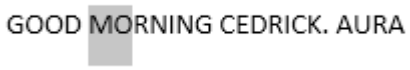
b)

GOOD MORNING CEDRICK. AURA

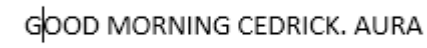
LNCCMD CMD(WMOVE) PARM1('While="R"'): cursor has moved by 1 character. result = 1
LNCCMD CMD(WMOVE) PARM1('While="R";count=2') : cursor has moved by 1 character. result = 1

GOOD MORNING CEDRICK. AURA

c)

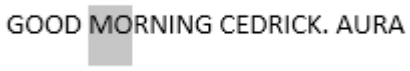


LNCCMD CMD(WMOVE) PARM1('While=" DOM";count=-5') : return =-4

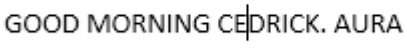


Example 2:

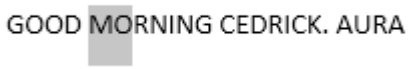
a)



LNCCMD CMD(WMOVE) PARM1('Until="D";count=20') : result=8. cursor has moved by 8 characters



b)

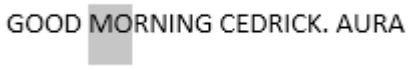


LNCCMD CMD(WMOVE) PARM1('Until="D";count=2'): result=0

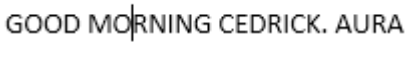
No characters “D” found into the range specified by count=2. The range isn't not changed and the method returns 0 (zero).

Example 3:

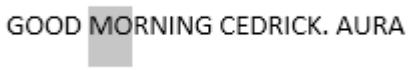
a)



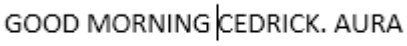
LNCCMD CMD(WMOVE) : result=1 . One character moved



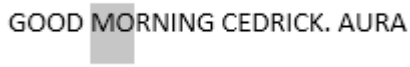
b)



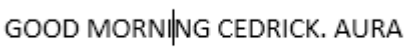
LNCCMD CMD(WMOVE) PARM1('Unit=wdWord') : result= 1. One Word moved.



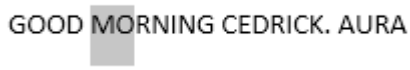
c)



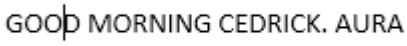
LNCCMD CMD(WMOVE) PARM1('count=4')



d)

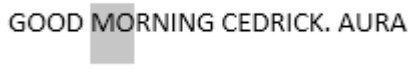


LNCCMD CMD(WMOVE) PARM1('count=-3')



Example 4:

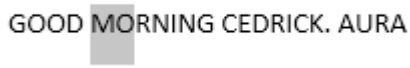
a)



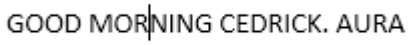
LNCCMD CMD(WMOVE) PARM1('Direction="Start";While="Z"') : result = 0

LNCCMD CMD(WMOVE) PARM1('Direction="Start";While="R"'): result = 0. The range starts with the M character

b)

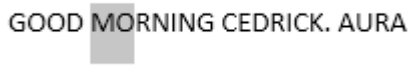


LNCCMD CMD(WMOVE) PARM1('Direction="Start";While="MOR"'): result = 3. The start of the Range has moved by 3 characters.

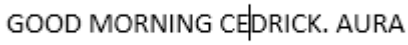


Example 5:

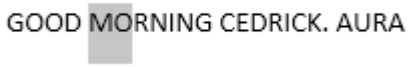
a)



LNCCMD CMD(WMOVE) PARM1('Direction="Start";Until="D";count=20') : result=11. cursor has moved by 11 characters from the start of the selection.

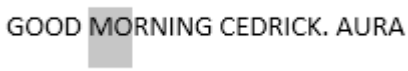


b)



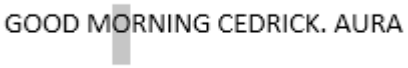
LNCCMD CMD(WMOVE) PARM1('Direction="Start";Until="D";count=2'): result=0

c)



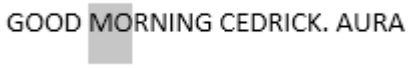
No characters “D” found into the range specified by count=2. The range isn't not changed and the method returns 0 (zero).

LNCCMD CMD(WMOVE) PARM1('Direction="Start";Until="O";count=2')

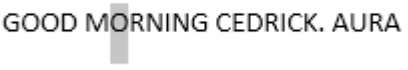


Example 6:

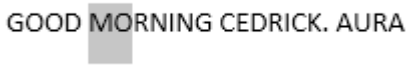
a)



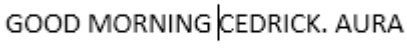
LNCCMD CMD(WMOVE) PARM1('Direction="Start"') : result=1 . One character moved.



b)

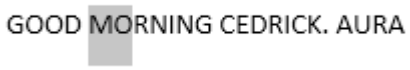


LNCCMD CMD(WMOVE) PARM1('Direction="Start";Unit=wdWord') : result= 1. One Word moved.

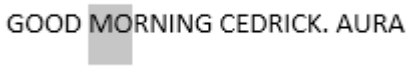


Example 7:

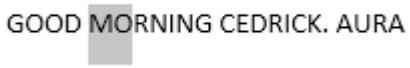
a)



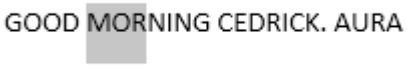
LNCCMD CMD(WMOVE) PARM1('Direction="End";While="Z"') : result = 0



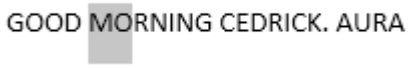
b)



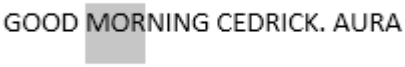
LNCCMD CMD(WMOVE) PARM1('Direction="End";While="R"'): result =1. One character moved.



c)

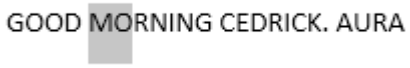


LNCCMD CMD(WMOVE) PARM1('Direction="End";While="MOR"'): result = 1. The end of the Range has moved by a character.

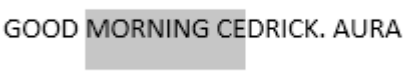


Example 8:

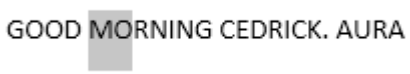
a)



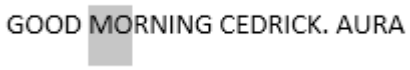
LNCCMD CMD(WMOVE) PARM1('Direction="End";Until="D";count=20') : result=9 cursor has moved forward by 9 characters from the end of the selection.



b)



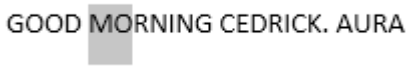
LNCCMD CMD(WMOVE) PARM1('Direction="End";Until="D";count=2'): result=0



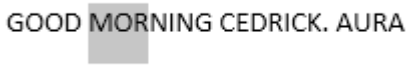
No characters “D” found into the range specified by count=2. The range isn't not changed and the method returns 0 (zero).

Example 9:

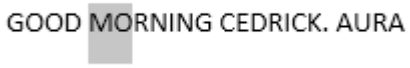
a)



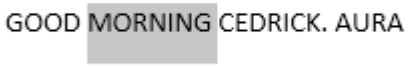
LNCCMD CMD(WMOVE) PARM1('Direction="End"') : result=1 . One character moved



b)

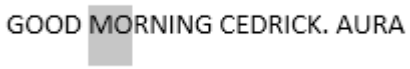


LNCCMD CMD(WMOVE) PARM1('Direction="End";Unit=wdWord') : result= 1. One Word moved.

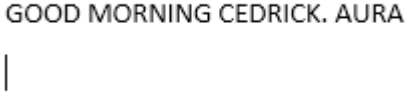


Example 10:

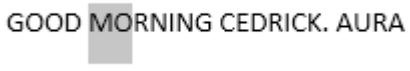
a)



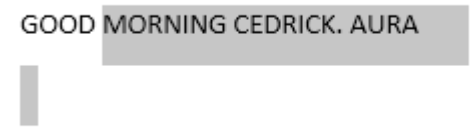
LNCCMD CMD(WMOVE) PARM1('Direction="Down"') : result=1 . One line moved



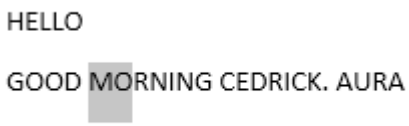
b)



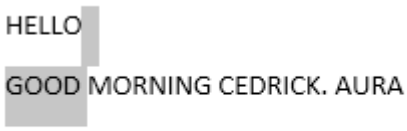
LNCCMD CMD(WMOVE) PARM1('Direction="Down";Extend=true') : result=1 . One line moved



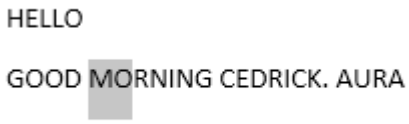
Example 11:



LNCCMD CMD(WMOVE) PARM1('Direction="Up";Extend=true'): result=1 . One line moved



Example 12:



LNCCMD CMD(WMOVE) PARM1('Direction="Right";Extend=true'): result=1 . One character moved.

Example 13:

HELLO

GOOD MORNING CEDRICK. AURA

LNCCMD CMD(WMOVE) PARM1('Direction="Left";Extend=true'): result=1 . One character moved.

HELLO

GOOD MORNING CEDRICK. AURA

WMOVERIGHT command

Moves cursor to right cell of a table.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WMOVERIGHT')
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Note

If the cursor is positioned in the last cell of the array and a WMOVERIGHT statement is executed, a new row is added to the array.

WNEWFILE command

Creates document from a Word template.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WNEWFILE')
CHGVAR          VAR(&PARM1) VALUE('model')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	Path and Name of the template, used to create new document. **.dotx or .docx must be defined.**
Parm2	

Example

```
CHGVAR          VAR(&CMD) VALUE('WNEWFILE')
CHGVAR          VAR(&PARM1) VALUE('c:\temp\model.dotx')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WOPENFILE
- WSAVE
- WSAVEAS
- WDOCUMENT

WOPENFILE command

Opens a WORD document.

The WORDOPEN command must have been used before.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WOPENFILE')
CHGVAR          VAR(&PARM1) VALUE('file')
CHGVAR          VAR(&PARM2) VALUE('
                [;*ONLY]
                [;*DOCPWD=xxx]
                [;*WRTPWD=yyy]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**Path and name of the file to open.**
Parm2	Specifies ***ONLY** to open file in read only mode. Specifies ***DOCPWD** to indicate document opening password. Specifies ***WRTPWD** to indicate document modifying password.

Example

```
CHGVAR              VAR(&CMD)VALUE('WOPENFILE')
CHGVAR              VAR(&PARM1) + VALUE('c:\temp\file.docx')
CHGVAR              VAR(&PARM2)VALUE('*ONLY')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WCLOSEFILE
- WSAVEAS
- WDOCUMENT

WORDCLOSE command

Closes Microsoft Word.

Standard closing messages will be displayed if a user action becomes necessary (background printing, unsaved modified file).

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WORDCLOSE')
CHGVAR          VAR(&PARM1) VALUE('save=true/false')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Paramètres

Parameters	
Parm1	**Save =** True/False If 'Save' is not filled, Word will ask you a question. "Do you want to save the changes to docname.docx?" "
Parm2	

See also

- WORDOPEN

WORDHIDE command

Hides active Word instance.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WORDHIDE')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WORDSHOW

WORDOPEN command

Opens Microsoft WORD.

A document can be specified in Parm1.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('WORDOPEN')
CHGVAR        VAR(&PARM1) VALUE('File name')
CHGVAR        VAR(&PARM2) VALUE('
              [Directory="Directory path";]
              [Document="File name";]
              [Detach = True/False;]
              [Visible;]
              [ReadOnly;]
              [New;]
              [Minimize;]
              [Maximize;]
              [DOCPWD="Password for opening";]
              [WRTPWD="Password for modification";]
              ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Name of the file to open. It may be empty if a new document has to be created, or includes a template name, **if NEW is specified as parameter 2**.
Parm2	**Directory** This option allows you to specify a path to the directory where the document to be opened is located. This directory path is added in front of the path and the file name given in parameter 1. **Document** By the keyword "Document", LAUNCHER Office is asked to search for the document among those already opened by Word on the system. If the requested document is already open, it is made active, it becomes the current document for Word. No new file is opened, so the name of the file or template is

	ignored. **Detach** = True/False If "Detach" is true, Word detaches the data source from the current document or the new active document when the WDOCUMENT command is returned. By detaching the data source, the document is no longer a mail merge template for Word. The WMAILMERGE command will redo the attachment to a data source. When a document is opened and attached to a data source (Merge File), it is maintained by Word, and therefore not editable. You must detach the data source from the current document if you want to send new data to the merge file. **VISIBLE** =to show Word. **MINIMIZE**= window is reduced. **MAXIMIZE**= window is widened. **READONLY** = open for reading only. *Default value is False* **NEW** = allows to create a new document if Parm1 is empty. **DOCPWD** = opening password. Uppercase/lowercase must be respected if the document is password protected. **WRTPWD**= modification password. Must be filled if the document is modification protected. **NOTICE !!! Word must be shown to open a password protected document.**
RESULT	**Name of the document** When returning the WDOCUMENT command, the &RESULT parameter contains the name of the active document. For a document opened from a file, the name of the

	document is equal to the name of the file, without the path and without the suffix ".DOCX". If the document has just been created, its name was built by Word.

Example

```
LNCCMD CMD(WORDOPEN) PARM1('C:\A\Templates\sp_cust.docx')
```

See also

- WORDCLOSE
- WDOCUMENT

WORDSHOW command

Shows active instance of Word.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WORDSHOW')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WORDHIDE

WORDWAIT command

Wait for a Word event.

The AS / 400 program waits for the closing of Word (**WORDCLOSE**), or an action on the custom menu (**WMENU** command), to continue its execution.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WORDWAIT')
CHGVAR          VAR(&PARM1) VALUE('[BackPrint=Value] ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The **BackPrint** property allows you to wait for a background print to finish. The value is expressed in seconds. 0 = infinite.

See also

- WMENU

WPAGEBRK command

Inserts a page break at the current position in the Word document.

In a table, the current line of the insertion point is moved to the next page.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WPAGEBRK')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

WPASTE command

Paste at the current cursor position, the contents of the clipboard.

The **WCOPY** command can be used to copy text to the clipboard.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WPASTE')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WCOPY

WPRINT command

Allows to print the current Word document, and set some properties and options.

Syntax

```
CHGVAR      VAR(&CMD) VALUE('WPRINT')
CHGVAR      VAR(&PARM1) VALUE('
            [Printer="Printer name"]
            [;TrayID=Tray ID]
            [;Tray="Tray name"]
            [;Copies=Copies number]
            [;BackGround=True/False]
            [;Pages="Pages to print"]
            [;PageType=Pages type]
            [;FirstPageTray= Tray ID or name]
            [;OtherPagesTray Tray ID or name]
            [;OutFile="Output file"]
            [Execute=True/False]
            ')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
            &PARM2 &RESULT))
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1 or Parm2	**Printer** is the name of the printer. If this parameter is not present, printing is done on the Windows default printer (see Control Panel). **TrayID** is a tray identifier. The possible values are listed below. **Tray** is the name of the bin. You must specify a string as it appears in the the properties of the printer. **Copies** is the number of copies to print. **BackGround**: True/False. False by default. To print in the background. **PageType** is the type of pages to print, can take one of the following constant values: *wdPrintAllPages*: print even and odd pages *wdPrintEvenPagesOnly*: print even pages *wdPrintOddPagesOnly*. : print odd pages

Pages is the number of pages and/or the range of pages to print, separated by commas (,).
For example, "2, 6-10" means: print page 2 and pages 6 to 10.

FirstPageTray is the identifier or name of the bin for the first page. The possible values for the identifier are listed below.

OtherPagesTray is the identifier or the name of the bin for the following pages. The possible values for the identifier are listed below.

OutFile: Full path to an output file that will receive the print stream.

Execute: If "Execute" is false, then the properties of the document are changed (Copies, Pages, Trays, ...) but the document is not printed.

Notes

1)
All parameters are optional.

2)
Possible values for Tray ID :

wdPrinterAutomaticSheetFeed ; wdPrinterDefaultBin ; wdPrinterEnvelopeFeedwd ; PrinterFormSource ; wdPrinterLargeCapacityBin ; wdPrinterLargeFormatBin ; wdPrinterLowerBin ; wdPrinterManualEnvelopeFeed ; wdPrinterManualFeed ; wdPrinterMiddleBin ; wdPrinterOnlyBin ; wdPrinterPaperCassette ; wdPrinterSmallFormalBin ; wdPrinterTractorFeed ; wdPrinterUpperBin

Be careful, these constants do not seem to be allowed with some printers. It is therefore better to use the name of the bin.

Example

```
CHGVAR          VAR(&CMD) VALUE('WPRINT')
CHGVAR          VAR(&PARM1) VALUE('Printer="Brother HL-1270N +
                series on Ne00"; Copies=2;+
                TrayID=wdPrinterLowerBin;BackGround=False;+
                Pages="2,4"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- IFSPUT
- CRTPRNSPLF - CL Command
- SETPRINTER
- WPRINTOUT

WPROTECT command

Protect the active Word document from modifications.

If a documents is protected, the user is allowed to make only some restricted modifications : comments, modify the document, but the track changes option is enabled.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WPROTECT')
CHGVAR          VAR(&PARM1) VALUE('
                Level=Protection type
                [;Sections="Sections to protect"]
                [;PWD="Password"]
                [;Exclude]
                [;Add]
                [;Remove]
                [;UnProtect]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1 or Parm2	**Level** is used to set the protection level to apply to the document. Available values are : • **wdAllowOnlyComments** : only comments are allowed. • **wdAllowOnlyFormFields** : document is considered as form. • **wdAllowOnlyRevisions** : changes with traking are allowed. • **wdNoProtection :** no protection. **PWD** : password associated to the protection. **UnProtect** : remove protection of the document. Only the **PWD** parameter is taking into account with this option. The following options only apply if **Level** is equal to **wdAllowOnlyFormFields** : - **Sections** allows to choose the sections to protect, or to exclude from the protection. Section numbers are enclosed in quotation marks, separated by commas. The special value "***ALL**" indicates that you want to protect or unprotect all sections.

The special value "***CURRENT**" refers to the section in which the current selection is located.
If the **Sections** keyword is missing, the protection applies to the sections as it was before any protection removal.

- **Exclude :** the sections specified into the **Sections** parameter are not protected. Others sections of the document are protected**.**
- **Add :** the sections specified into the **Sections** parameter are protected. Others sections of the document remain unchanged.
- **Remove :** the protection of the sections specified into the **Sections** parameter is removed. Others sections of the document remain unchanged.

If the **Exclude**, **Add** and **Remove** options are not present, the protection is applied to the sections specified into the **Sections** parameter, and removed for the others ones.

Note

If the **wdAllowOnlyFormFields** level is applied on a document containing no form fields, therefore no modification will be possible on the document.

Examples

```
CHGVAR          VAR(&CMD) VALUE('WPROTECT')
CHGVAR          VAR(&PARM1) VALUE('Level=wdallownonlycomments')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))


CHGVAR          VAR(&CMD) VALUE('WPROTECT')
CHGVAR          VAR(&PARM1) VALUE('Level=wdAllowOnlyFormFields')
CHGVAR          VAR(&PARM2) VALUE('Sections ="2,3,5"')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))


CHGVAR          VAR(&CMD) VALUE('WPROTECT')
CHGVAR          VAR(&PARM1) VALUE('Level= wdAllowOnlyFormFields')
CHGVAR          VAR(&PARM2) VALUE('Sections="*CURRENT";Exclude')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WDPROTECT

WREFVAR command

Writes the value of a document variable to the cursor location in the current Word document.

This variable may have been created or modified using the **WSETVAR** command.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WREFVAR')
CHGVAR          VAR(&PARM1) VALUE('
                Text="Variable name"
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Text**: Name of the document variable. The value of the variable will be written to the current Word document at the cursor location.

Example

In the following example, we place ourselves on the bookmark "VAR7RES", and we assign it the value of the variable "VAR7":

```
LNCCMD      CMD(WOPENFILE) PARM1('C:\A\Test.docx')
LNCCMD      CMD(WBOOKMARK) PARM1('BOOKMARK="VAR7RES"')
LNCCMD      CMD(WREFVAR) PARM1('Text="VAR7"')
```

See also

- WSETVAR

WREPLACE command

Finds text in the active Word document and can replace it by a new text value. Some properties of the replacement can be set.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WREPLACE')
CHGVAR          VAR(&PARM1) VALUE('
                Find="Text to find"
                [;ReplaceWith="New text"]
                [;MatchCase=True/False]
                [;MatchWholeWord=True/False]
                [;MatchWildcards=True/False]
                [;MatchSoundsLike=True/False]
                [;Replace=constant]
                [;Forward=True/False]
                [;Wrap=constant]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Find**: the text that will be searched. **ReplaceWith**: the replacement text. To remove all occurrences of a word or text, specify the empty string ("") for this argument. If no value is given, the found text is simply selected. **MatchCase**: True/False. True if the search must be case sensitive. False by default. **MatchWholeWord**: True/False. True to search for occurrences constituting whole words and not parts of words. False by default. **MatchWildcards**: True/False. True to use wildcards and special operators. For example: ^p for a paragraph mark, ^t for a tabulation ... False by default. **MatchSoundsLike**: True/False. Searches words that sound phonetically but with a different spelling. False by default. **Replace**: indicates the number of replacements to perform: one, all or none. Uses the following constants:

wdReplaceAll (default), **wdReplaceNone**, or **wdReplaceOne**.

Forward: True/False. Search forward, near the end of the document. True by default.

Wrap:

WdFindAsk: After replacing in the selection, Microsoft displays a message asking if you want to search the rest of the document.
WdFindContinue (default): The replace operation continues after selection.
WdFindStop: The replace operation ends when the end of the selection is reached.

Example

```
CHGVAR VAR(&CMD) VALUE('WORDOPEN')
CHGVAR VAR(&PARM1) VALUE('FILE="C:\TEMP\TEST.DOCX"')
CHGVAR VAR(&PARM2) VALUE('VISIBLE')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('WREPLACE')
CHGVAR VAR(&PARM1) VALUE('FIND="Bonjour";+
REPLACEWITH="Hello"')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

See also

- WFINDTEXT

WSAVE command

Saves changes to the current Word document.

If this document has not been saved before, a dialog box will appear asking for the backup name.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WSAVE')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

WSAVEAS command

Save the current WORD document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WSAVEAS')
CHGVAR          VAR(&PARM1) VALUE('
                File="File name"
                [;RecentFiles=True/False]
                [;Format=constant]
                [;OptimizeSize=True/False]
                [;PdfA=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**File** : Path and backup name of the current file. **RecentFiles**: True/False. If True, the saved file is added to the list of last used files. False by default. **Format:** The format must be consistent with the extension of the saved file. You can use the Word syntax: *wdFormatDocument, wdFormatDOSText, wdFormatDOSTextLineBreaks, wdFormatEncodedText, wdFormatHTML, wdFormatRTF, wdFormatTemplate, wdFormatText, wdFormatTextLineBreaks, wdFormatUnicodeText ...* or the following keywords: - DOC - DOCX - DOCM - PDF - HTML - XPS If you do not set the **Format** parameter, then the default format of MS Word is used. Example: DOCX for Word 2010 and later versions.

OptimizeSize: True/False. False by default.

This parameter is taken into account only if Format=PDF.

If True, the PDF resolution and quality are optimized for on-screen viewing. The quality is less and the size is minimal.

If False, the PDF is optimized for printing. The quality is optimal and the file size is larger.

PdfA: True/False. False by default.

This parameter is taken into account only if Format=PDF.

If True, the saved PDF will be compatible with the PDF/A-3A standard (ISO name: ISO 19005-3).

Examples

1)

```
LNCCMD CMD(WSAVEAS)
PARM1('File="c:\temp\test_wsaveas.docx";format="DOCX"')
```

2)

```
LNCCMD CMD(WSAVEAS)
PARM1('File="c:\temp\test_wsaveas.docm";format="DOCM"')
```

3)

```
LNCCMD     CMD(WSAVEAS)
PARM1('FILE="C:\A\test.PDF";FORMAT=PDF;OptimizeSize=true')
```

To have a minimum PDF size, the **OptimizeSize** option is equal to true.

4)

```
LNCCMD     CMD(WSAVEAS)
PARM1('File="C:\temp\test\4\Doc3.pdf";Format=PDF;PdfA=true')
```

To have a PDF compatible with the PDF/A-3A standard.

See also

- WCLOSEFILE
- WOPENFILE

WSELECT command

Selects text in Word document.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('WSELECT')
CHGVAR        VAR(&PARM1) VALUE('*WORD or *SENTENCE or *PARAGRAPH or *LINE
              or *COLUMN or *ROW or *CELL or *TABLE')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT))
MONMSG        MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	Selection type to perform : ***WORD** : selects word. ***SENTENCE** : selects sentence. ***PARAGRAPH** : selects section. ***LINE** : selects line ***COLUMN** : selects column (in table) ***ROW** : selects row (in table) ***CELL** : selects cell (in table) ***TABLE** : selects complete table

Example

```
CHGVAR        VAR(&CMD) VALUE('WSELECT')
CHGVAR        VAR(&PARM1) VALUE('*TABLE')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT))
MONMSG        MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- WCOPY
- WGETTEXT

WSELROW command

Selects a row in a table.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('WSELROW')
CHGVAR              VAR(&PARM1) VALUE(' ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

WSETLINE command

Assigns values to cells of a Word table.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WSETLINE')
CHGVAR              VAR(&PARM1)VALUE('Values of cells')
CHGVAR              VAR(&PARM1)VALUE('Formats of cells')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**A string containing the values to assign to cells.** The values of each cell are separated by the word : **%SEP%**. The string can contain the following symbolic words: **%SEP%**: To move to the next cell. If we were positioned on the last column of the table, we go to the beginning of the next line. **%INS%**: To insert a new line, and move to the beginning of the new line. **%MRG%**: To merge the current cell with the next one.
Parm2	**Character string that can contain cell formats.** The formats of each cell are separated by the word: **%SEP%**. Several formats can be applied to a cell. They are then separated by a semicolon. The formats can be: **NUMFMT(d [DECPOINT=p] [GRPPOINT=g])** with : -**d** is the number of decimals, -**p** represents the character to use for the decimal point. (Point character by default), -**g** is the character to use to separate groups of 3 digits. **PROP** sets the value of a property for the contents of the corresponding cell. Example:

PROP(Font.bold)=True; PROP(Font.italic)=True

Example

```
CHGVAR          VAR(&CMD) VALUE('WSETLINE')
CHGVAR          VAR(&PARM1) VALUE('%INS%Valeur colonne 1%SEP%Valeur Colonne
                2 & 3%MRG%%SEP%12300%SEP%')
CHGVAR          VAR(&PARM2) VALUE('%SEP%PROP(Font.Italic)=True;PROP(Par+
                agraphs.Item(1).Alignment)=wdAlignParagraph+
                Right%SEP%NUMFMT(2)')
```

In this example, PARM1 contains values.

%INS% inserts a new line.

« Column 1 value » is assigned to first cell.

« Columns 2 & 3 values » is assigned to 2nd cell.

Cells #2 and #3 are merged with %MRG%.

« 123,00 » is assigned to 4th cell.

Last %SEP% moves current position to the next cell.
If the table contains only 4 columns, current position is moved to next line beginning.

PARM2 contains formats :

PARM2 begins with %SEP%, so, 1st cell has no particular format.

2nd cell (merged with 3rd cell) will be italics, right aligned.

4th cell is digital with two decimals.

See also

- Special values

WSETVAR command

Assigns a value to a document variable in the current Word document.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WSETVAR')
CHGVAR          VAR(&PARM1) VALUE('
                [Variable="Variable name"]
                [;Value="Value"]
                [;Update=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Variable**: Name of the variable of the document. If no name is specified, only the **Update** property will be taken into consideration. **Value**: The text to assign to the variable. **Update**: Allows you to update all fields and variables in the document if **True** is specified.
Parm2	Parm2 can contain the value to be assigned to the variable if it was not given in Parm1.

Note

The Word document variable, modified by the WSETVAR command, can then be used in VBA code.

Suppose we execute the following commands:

```
LNCCMD CMD (WSETVAR) PARM1 ('variable = "A"; value = "100"')

LNCCMD CMD (WSETVAR) PARM1 ('VARIABLE = "B"; VALUE = "3"')
```

In VBA, variables can be retrieved as follows:

```
ActiveDocument.Variables.Item( "A")

ActiveDocument.Variables.Item( "B")
```

Example

In this example the "Sir" value is assigned to the "Civil" variable.

```
CHGVAR          VAR(&CMD) VALUE('WSETVAR')
CHGVAR          VAR(&PARM1) VALUE('Variable="Civil";Value="Sir"')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

See also

- WREFVAR

WSETPASSWD command

Allows to define password for the current Word document.

Two password kinds exist :

- One to open a document.
- Another to allows its modification.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('WSETPASSWD')
CHGVAR          VAR(&PARM1) VALUE('
                Type="Password type"
                ;PWD="Password"
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

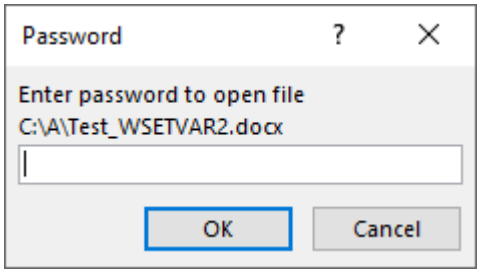
Parameters	
Parm1	**Type** is the type of password to add to the document: • ***DOCPWD** to specify the password at the opening • ***WRTPWD** for the one allowing the writing. The password is given by the keyword **PWD**.

Example

```
LNCCMD    CMD(WOPENFILE) PARM1('C:\A\Test.docx')

LNCCMD    CMD(WSETPASSWD) PARM1('Type=""*DOCPWD";PWD="aura"')
```

Note

Opening a password-protected Word document, with **WOPENFILE** for example, will make Word visible and the appearance of a popup that will interfere with Launcher in service mode, if we do not specify the password in the command.



To avoid having a popup, you must specify the password with **WOPENFILE**:

```
LNCCMD      CMD(WOPENFILE) PARM1('C:\A\Test_WSETVAR2.docx') +
                  PARM2('*DOCPWD=aura')
```

WSETPROP command

Sets a property value to the active Word document or the application.

Syntax 1

```
CHGVAR          VAR(&CMD) VALUE('WSETPROP')
CHGVAR          VAR(&PARM1) VALUE('Property')
CHGVAR          VAR(&PARM2) VALUE('Value')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	Full path to the property to modified. Example : Selection.Font.Bold
Parm2	Value to assign to property. Examples : TRUE : True value FALSE : False value INT(chain) : When Word waits for an integer value.

Examples 1

1) The following example sets selection font color to red.

```
CHGVAR          VAR(&CMD) VALUE('WSETPROP')
CHGVAR          VAR(&PARM1) VALUE('Selection.Font.Color')
CHGVAR          VAR(&PARM2) VALUE('INT(255)')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
```

Note : Parm2 = INT(255) : Red Color.

2) This example inserts text in Page header and page footer.

```
LNCCMD CMD(WORDOPEN)

LNCCMD CMD(WOPENFILE) +
PARM1('%LNCDIR%\SAMPLES\MODELE_STYLE.DOC')

LNCCMD CMD(WSETPROP) +
PARM1(activewindow.activepane.view.seekview+
) PARM2(wdseekcurrentpageheader)

LNCCMD CMD(WTYPETEXT) PARM1('PAGE HEADER')

LNCCMD CMD(WSETPROP) +
PARM1(ACTIVEWINDOW.ACTIVEPANE.VIEW.SEEKVIEW+
) PARM2(WDSEEKCURRENTPAGEFOOTER)

LNCCMD CMD(WTYPETEXT) PARM1('PAGE FOOTER')

LNCCMD CMD(WORDSHOW)
```

3) This example puts the selection in bold.

```
LNCCMD     CMD(WSETPROP) PARM1('Selection.Font.Bold') +
              PARM2('True')
```

4) This example disables the display of Word popups.

```
LNCCMD     CMD(WSETPROP) +
              PARM1('application.displayalerts') +
             PARM2('wdalertsnone')
```

Syntax 2

```
CHGVAR         VAR(&CMD) VALUE('WSETPROP')
CHGVAR         VAR(&PARM1) VALUE('
               Property="Property";
CHGVAR         Value="Value"')
CALL           PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
               &PARM2 &RESULT))
MONMSG         MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	**Property** = Full path to property to modified.
or Parm2	**Value** = Value to assign to the property.

Example 2

```
LNCCMD     CMD(WSETPROP) PARM1('Property= "SELECTION.STYLE" ;Value="
ACCENTUATION"') +
```

Note

The path to the property follows the syntax used in Visual Basic.

Some examples :

Selection.Font.Color: Changes the color of the selection.
Selection.Borders.Item(wdBorderLeft).LineStyle: Changes the style of the left border.
Selection.Font.Bold: Allows to put in bold the selection.

It is possible to use **all the constants of Word and Visual Basic Application** or the value of the constant itself.

When the path to the property includes an element of a collection (such as Selection.Borders, which has 4 borders), you must specify the element concerned (Item).

WSPLITCELL command

Splits the current cell into several cells.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WSPLITCELL')
CHGVAR              VAR(&PARM1)VALUE('Number Cells')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Number indicating in how many cells will be splitted the active cell.
Parm2	

Example

```
CHGVAR      VAR(&CMD) VALUE('WSPLITCELL')
CHGVAR      VAR(&PARM1) VALUE('4')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
            &PARM2 &RESULT)
```

See also

- WMERGECELL

WTYPETEXT command

Inserts text into a Word document at the cursor location.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('WTYPETEXT')
CHGVAR              VAR(&PARM1)VALUE('Text')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Text to insert at the cursor location..
Parm2	

Example

```
LNCCMD     CMD(WTYPETEXT) PARM1('Text added with WTYPETEXT')
```

XLADDINS command

Adds an EXCEL add-in to the list of add-ins (File, Options, Add-ins, Excel add-ins, Go). The add-in is also activated.

An add-in is an add-on program that adds custom commands or features to Microsoft EXCEL.

An Excel add-in is in **.xla** format (2003 compatibility mode) or **.xlam**.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLADDINS')
CHGVAR          VAR(&PARM1) VALUE('Add-in')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Full path to add-in to install.
Parm2	

Note

XLADDINS can be used to add a template that contains macros and make them available.

Example

Loading the add-in provided with Launcher to create a custom menu with Launcher (**XLMENU** command):

```
CHGVAR          VAR(&CMD) VALUE('XLADDINS')
CHGVAR          VAR(&PARM1) VALUE('%LNCDIR%\LNCEXCELADDIN.XLA')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

See also

- XLMENU
- XLENAADD
- XLDISADD

XLADDSHEET command

Adds a sheet to the current workbook.

The new sheet is added before the current sheet.
The new sheet becomes the current sheet.

Syntax

```
CHGVAR              VAR(&CMD) VALUE('XLADDSHEET')
CHGVAR              VAR(&PARM1) VALUE(' ')
CHGVAR              VAR(&PARM2) VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

See also

- XLDELSHEET
- XLGOTOSH
- XLPRINTSH
- XLSHNAME

XLBOLD command

This command is used to put the value of the cell in bold.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLBOLD')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

See also

- XLITALIC
- XLUNDERLN

XLCALCULAT command

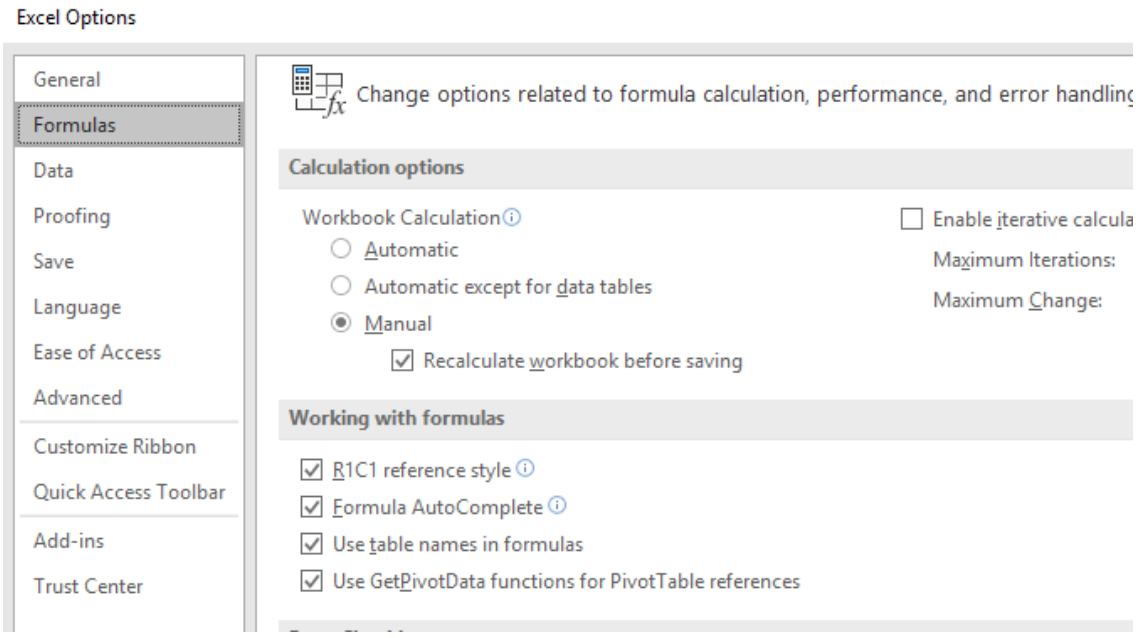
Calculates the current sheet.

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLCALCULAT')
CHGVAR          VAR(&PARM1)VALUE(' ')
CHGVAR          VAR(&PARM2)VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Note

This command will only work if the "manual" option is configured for the Excel calculation options (F9 for manual execution**).**



XLCELLNAME command

Renames a cell in the current sheet.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLCELLNAME')
CHGVAR          VAR(&PARM1) VALUE('coordinates')
CHGVAR          VAR(&PARM2) VALUE('name')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**The coordinates of the cell to rename.** $Column$Row
Parm2	**New name of the cell.**

Example

```
CHGVAR          VAR(&CMD)VALUE('XLCELLNAME')
CHGVAR          VAR(&PARM1)VALUE('$A$3')
CHGVAR          VAR(&PARM2) VALUE('Sum')
```

See also

- XLGOTOCELL
- XLSETTEXT
- XLSETVALUE

XLCELLS Command

Allows you to perform actions on a cell or group of cells.

- Read the value
- Change the value
- Change the formula of the cell
- Change the format
- Assign a name
- Edit properties
- Read a property

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLCELLS')
CHGVAR          VAR(&PARM1) VALUE('
                [Ref="Cells reference"]
                [;Sheet="Sheet name"]
                [;Row= Row number or displacement]
                [;Col= Column number or displacement]
                [;RowCnt= Number of selected rows]
                [;ColCnt= Number of selected columns]
                [;RefTo= " End of selected area reference " ]
                [;NextOutLine="Levels"]
                [;SetValue=True/False ]
                [;SetFormula=True/False ]
                [;Format=" Format to be assigned "]
                [;Name=" Name to be assigned "]
                [;Comment="Comment"]
                [;GetValue=True/False]
                [;GetText=True/False]
                [;GetProp("Property")]
                [;GetSize=True/False]
                [;Select=True/False]
                [;Clear=True/False]
                [;Freeze=True/False]
                [;Calculate=True/False]
                [;Merge=xlColumn|xlNone|xlAll]
                [;Prop(Property)= Property value]
                [;DateFmt(Date format)]
                [;NumFmt(Numbers format)]
                [;IgnoreError=True/False]
                [;Unicode=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE('Value assigned to cells')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters

Parm1

Ref: The reference to the cell can be given in the form:

- Syntax **$B$3.** Example: $B$3 denotes column B, row 3.
- Syntax **CL**. **C** to designate the column and **L** to designate the row. Example: B3 designates row 3, and column 2 (B).
- Symbolic name given to a cell or group of cells.
- "." will designate the active cell.

Sheet: Specifies the name of the sheet to be addressed.
If a cell name is indicated by **Ref**, then the **Sheet** keyword is not useful. Otherwise, if the **Sheet** keyword is missing, the active sheet is addressed.

Row, Col: These two keywords represent respectively:

- The coordinates (Line, Column) of a cell, if **Ref** is absent.
- The relative coordinates in rows and columns from the base referenced by "Ref =" when it is present. The upper left cell has relative coordinates (1,1).

RowCnt, ColCnt: These two keywords represent respectively the number of rows and columns selected, from the reference given by **Ref** or by (**Row, Col**).
These keywords can not be used if **RefTo** is specified.

RefTo: Reference to the cell at the bottom right of the selected area. This reference is given by key words **Ref**, **Row** and **Col**, in a quoted string.
This keyword can not be specified if **RowCnt** or **ColCnt** are specified.

NextOutline: When this option is specified, LAUNCHER Office searches for the next row from the active cell, with a hierarchy level matching the criteria.
Rows have different hierarchy levels when subtotals have been applied to the sheet.
In the **NextOutline="x-y"** syntax, x represents the minimum hierarchy level, y the maximum level.
Example: The following syntax looks for the following subtotal row and applies background color and character formats:

```
NEXTOUTLINE="1-2";SELECT;IgnoreError;
PROP(INTERIOR.COLOR)=RGB(200;200;200); PROP(Font.Bold)= True
```

SetValue : Indicates that the **Parm2** parameter contains a value to assign to the cell.

SetFormula : Indicates that the **Parm2** parameter contains the text of a formula to apply to the cell.

Format: Allows you to set the format of the cell or group of cells.

Name: Give a name to the cell or group of cells.

Comment: Add a comment to the cell. This comment is displayed by Excel when we fly over the cell with the mouse.

GetValue: Indicates that you want to receive the internal value of the cell, not the displayed value. The value will be returned in the &RESULT parameter.

GetText: Indicates that you want to receive the displayed value of the cell. The displayed value may differ from the internal value.
For example, if the cell has a "Percentage" format, the internal value "**0.10**" will be displayed "**10.00%**".
The value will be returned in the & RESULT parameter.

GetProp ("Property"): Returns the value of a property assigned to cells in the referenced region.
Examples:
Prop ("Address") returns the coordinates of the referenced region.
Prop ("Font.color") returns the color of the characters.

GetSize: Returns in &RESULT the size of the region referenced by **Ref**.
The size is returned in &RESULT under the form:
lllll; ccccc
lllll represents the number of lines on 5 digits,
cccc is the number of columns in 5 digits.

Select: **True** indicates that the cell or group of cells should be selected.

Clear: **True** clears the cell.

Freeze: **True** replaces the expressions with their calculated values.

Calculate: True recalculates the entire workbook.

Merge: When this keyword is specified, the merge of the cells in the selection is changed.
Merge can take one of the following symbolic values, expressed without quotation marks:

> **xlAll**: All cells in the selection are merged into a single cell.
> **xlColumn**: The cells in each row of the selection are merged to form a merged cell per line.
> **xlNone**: The merge is deleted.
> Be careful, if merged cells contain values, Excel displays a message asking for confirmation of the loss of these values.
> Use the XLSETPROP command, with "DisplayAlerts = false" to disable the display of this message.

Prop(): allows to modify a property of the pointed cells.
The keyword "**Prop**" can appear several times in Parm1.

	Example: Change the background color, and the alignment *Prop(Interior.color)=RGB(100,0,0);Prop(HorizontalAlignment)=xlCenter*. **IgnoreError**: If this option is true, in the event of an error, no message will be generated, and the value "*ERROR" will be returned in &RESULT. **DateFmt(format)** : When a date value is provided in the "Parm2" parameter, this keyword indicates in what format this value is. The **dd, mm, yy** and **yyyy** strings will be used to denote, respectively, the day, the month, the year on 2 digits and the year on 4 digits. By default, Excel considers dates in the format **'mm/dd yyyy'** Examples: **DateFmt (dd/mm/yyyy)** indicates that the date is provided in Day-Month-Year format, separated by '/'. **DateFmt (yyyymmdd)**, the date is in the format Year-Month-Day, without separator. **NumFmt (format)** : When a decimal number, without a decimal point, is provided in the "Parm2" parameter, this key word indicates how many decimal places have this number. Example: Parm2 = '**123456**'; **NumFmt(2)** indicates that the value sent is: **1234.56**. **Unicode**: When Unicode is true, the value read by **GetValue**, **GetText**, or written by **SetValue** is in the Unicode character set of Windows. To write a value in Unicode, &PARM2 must contain a Unicode string, which can be converted by the program **LNCCVTWCS**. When reading a value, the &RESULT parameter will contain a value in Unicode, which can be converted with **LNCCVTWCS**.
Parm2	If one of the keywords **SetValue** or **SetFormula** is set into Parm1, then Parm2 contains the value for the cell or the formula. If **Unicode** is true, &PARM2 value must be in Unicode character set, and parameter **&OPT** must set to '**W**'.
Opt	When option **Unicode** is true, **&OPT** must be set to '**W**'. **&PARM2** must be coded in the Windows Unicode character set.

Examples

```
                /* Read the value of cell B2 */
                /* and assign it a new value */
CHGVAR          VAR(&CMD) VALUE('XLCELLS')
```

```
CHGVAR        VAR(&PARM1) VALUE('Ref="$B$2";SetValue=True;GetText=True')
CHGVAR        VAR(&PARM2) VALUE(&NEWVAL)
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
              /* &RESULT contient l'ancienne valeur de B2 */

              /* Change the format of the cell Row 10, Column 5*/
CHGVAR        VAR(&CMD) VALUE('XLCELLS')
CHGVAR        VAR(&PARM1) VALUE('Row=10;Col=5;Format="# ##0,00"')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)


              /* Write a Chinese text in a cell */
             /* Convert from chinese CCSID to unicode Windows */

CHGVAR        VAR(&WCS_LEN) VALUE(512)
CHGVAR        VAR(&CCSID) VALUE(835)
CALL          PGM(LNCCVTWCS) PARM(&HANDLE '*TO ' &PARM2 &WCS_LEN +
              &CHINA &CHINA_LEN &CCSID &RES_SIZE)

CHGVAR        VAR(&CMD) VALUE('XLCELLS')
CHGVAR        VAR(&PARM1) VALUE('Ref="$B$2";SetValue=True;Unicode=True')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD 'W' &PARM1 +
              &PARM2 &RESULT)


              /* Select an area of 3 rows, 5 columns */
              /* From the cell named "Table". */
CHGVAR        VAR(&CMD) VALUE('XLCELLS')
CHGVAR        VAR(&PARM1) VALUE('Ref="Table";RowCnt=3;ColCnt=5;Select')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)


              /* Select a zone */
              /* from the cell named "Table" */
              /* to the cell named "FinTable". */
CHGVAR        VAR(&CMD) VALUE('XLCELLS')
CHGVAR        VAR(&PARM1) VALUE('Ref="Table";RefTo="FinTable";Select')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)


              /* Select a zone */
              /* from the cell named "Table" */
              /* to the cell Line 12, column 20. */
CHGVAR        VAR(&CMD) VALUE('XLCELLS')
CHGVAR        VAR(&PARM1) VALUE('Ref="Table";+
```

```
                RefTo="Row=12;Col=20";Select')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)

                /* Apply formula */
LNCCMD CMD(XLCELLS)
PARM1('Ref="H8";setformula=True;calculate=True')
PARM2('=IF(ISERROR(-1+D5/E5)," ",-1+D5/E5)')
```

Note

To use the **Font.Color** property, it is necessary to use Windows color codes.
Color constants:

Constant	Value	Description
vbBlack	&H0	Black
vbRed	&HFF	Red
vbGreen	&HFF00	Green
vbYellow	&HFFFF	Yellow
vbBlue	&HFF0000	Blue
vbMagenta	&HFF00FF	Magenta
vbCyan	&HFFFF00	Cyan
vbWhite	&HFFFFFF	White

Constants values:

vbBlack	0
vbBlue	16711680
vbCyan	16776960
vbGreen	65280
vbMagenta	16711935
vbRed	255
vbWhite	16777215
vbYellow	65535

PROP(Font.Color)=vbMagenta
PROP(Font.Color)=INT(255) //red

You can also use BGR code with the RGB function:

```
PROP(Font.Color)=RGB(255,0,0) //blue
PROP(Font.Color)=RGB(0,255,0) // green
PROP(Font.Color)=RGB(0,0,255) // red
```

XLCLOSEFILE command

Closes the active EXCEL workbook.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLCLOSEFILE')
CHGVAR              VAR(&PARM1)VALUE('[*YES|*NO]')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Save options:** ***YES**: Saves modifications and closes the workbook. ***NO**: Closes the workbook without saving modifications. **If no option is chosen, an Office popup will appears** : « Do you want to save modifications ? ».
Parm2	

Note 1 :

Be careful when you use the following command :

```
LNCCMD CMD(XLCLOSFILE) PARM1('*YES')
```

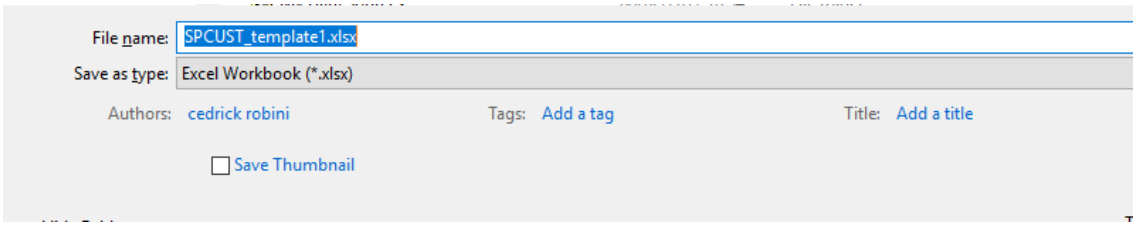
This command has to be used on workbook already save one time.

It should not be used on a new workbook because otherwise an Excel pop-up will open (if Launcher is in interactive mode) to ask to save the workbook.

Example :

```
LNCTOXLS TOXLS('C:\A\Templates\SPCUST_template.xls')
FROMFILE(SP_CUST) TONAME(TABLEAU1) XLSMAP(*MAPNAME) SAVFMT(*NORMAL)
ENDOPT(*NONE) SAVDOC(*NONE)

LNCCMD CMD(XLCELLS) PARM1('Ref="$C$11";Setvalue=true') PARM2('1')
LNCCMD CMD(XLCLOSFILE) PARM1('*YES')
```

After LNCTOXLS ENDOPT(*NONE) SAVDOC(*NONE), a new workbook, based on model, is created. After XLCLOSFILE *YES, there is this Excel popup (if Launcher is in interactive mode) to save the workbook:



Note 2 :

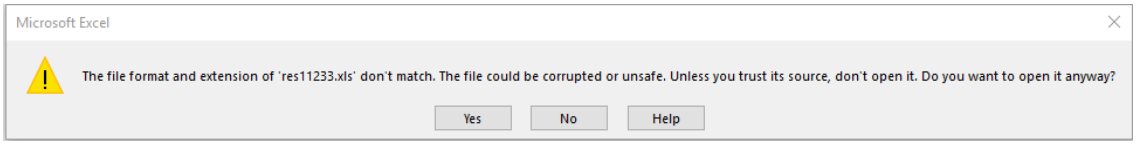
Be careful if you use Excel in compatibility mode !
By default, the SAVFMT parameter of LNCTOXLS is equal to *NORMAL: the default format of Excel. With Excel versions newer than 2003, the defaut format is XLSX.

Example :

```
LNCTOXLS TOXLS('C:\A\Templates\SPCUST_template.xls')
FROMFILE(SP_CUST) TONAME(TABLEAU1) XLSMAP(*MAPNAME)
SAVDOC('c:\temp\res11233.xls') SAVFMT(*NORMAL) ENDOPT(*NONE)

LNCCMD CMD(XLCELLS) PARM1('Ref="$C$11";Setvalue=true') PARM2('1')

LNCCMD CMD(XLCLOSFILE) PARM1('*YES')

LNCCMD CMD(EXCELCLOSE)
```

The generated workbook will not be correct: the extension (.xls) and the format will not be homogeneous. Indeed the default format of Excel 2016 is .XLSX.

This is the Excel popup appearing when you open the generated workbook :



The right syntax is the following :

```
LNCTOXLS TOXLS('C:\A\Templates\SPCUST_template.xls')
FROMFILE(SP_CUST) TONAME(TABLEAU1) XLSMAP(*MAPNAME)
SAVDOC('c:\temp\res11233.xls') SAVFMT(*XLS) ENDOPT(*NONE)

LNCCMD CMD(XLCELLS) PARM1('Ref="$C$11";Setvalue=true') PARM2('1')

LNCCMD CMD(XLCLOSFILE) PARM1('*YES')

LNCCMD CMD(EXCELCLOSE)
```

See also

- XLOPENFILE
- XLSAVEAS

XLCOPY command

Copy the current selection or the contents of a cell to the clipboard or to a specified cell.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLCOPY')
CHGVAR          VAR(&PARM1) VALUE('
                [From="Coordinates cells to copy"];
                [To="Coordinates cells to destination"]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**From** : Indicates cells coordinates from which content must be copied. If **From** is not defined, the current selection will be copied to the clipboard. **To** : Indicates cells coordinates to copy to. If **To** is not defined, data will be copied to clipboard.

Notes

1)
The cells can be in the current sheet or in another sheet of the workbook. The name of the sheet and the coordinates of the cell will then be separated by a question mark.

Example: dest?F3 for cell F3 of the sheet named "dest".

2)
The name of the sheet must be enclosed in quotation marks if it contains spaces.

Example: "dest leaf"?A1 for cell A1 of the sheet named "dest leaf".

Examples

This example copies "Revenus" content to "$B$22" location.

```
CHGVAR          VAR(&CMD) VALUE('XLCOPY)
CHGVAR          VAR(&PARM1) VALUE('From="Revenus";To="$B$2"')
CHGVAR          VAR(&PARM2) VALUE(' ')
```

```
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

The following example copies current selection to clipboard.

```
CHGVAR          VAR(&CMD) VALUE('XLCOPY)
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Copy of the contents of A1:B2 of the sheet "gold leaf" in the sheet "dest".
The cell at the top left of the receiving area is cell F3.

```
LNCCMD CMD(XLCOPY) PARM1('From=""gold leaf"?A1:B2";To="dest?F3"')
```

See also

- XLPASTE

XLCOPYSH command

Copy the current sheet into the same workbook. The copy appears after the copied sheet.

The copy then becomes the current sheet.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLCOPYSH')
CHGVAR          VAR(&PARM1) VALUE(' ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Example

```
CHGVAR VAR(&CMD) VALUE('XLOPENFILE')
CHGVAR VAR(&PARM1) VALUE('FILE="C:\TEMP\CLASSEUR1.XLSX"')
CHGVAR VAR(&PARM2) VALUE('VISIBLE')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)

CHGVAR VAR(&CMD) VALUE('XLCOPYSH')
CHGVAR VAR(&PARM1) VALUE(' ')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
```

XLDELSHEET command

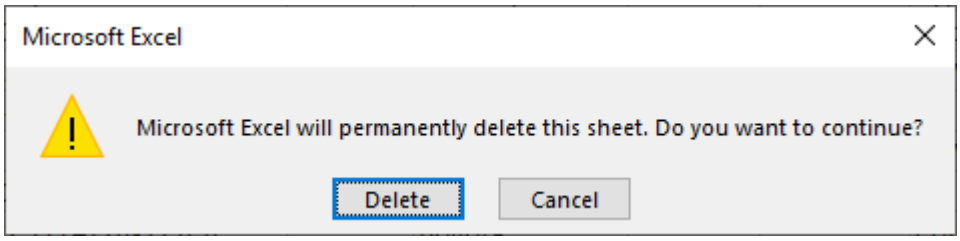
Deletes the current sheet. The sheets to the right of the deleted one are shifted to the left.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLDELSHEET')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Note

When you want to delete an Excel sheet with the **XLDELSHEET** instruction, the system sends a window for confirmation:



One solution to avoid this Excel window is to set the "DisplayAlerts" property to "False".

Use LNCCMD with the **XLSETPROP** command:

```
CHGVAR VAR(&CMD) VALUE(XLSETPROP)
CHGVAR VAR(&PARM1) VALUE('displayalerts')
CHGVAR VAR(PARM2) VALUE(false)
CALL         PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
             &PARM2 &RESULT)
```

See also

- XLADDSHEET
- XLGOTOSH
- XLPRINTSH
- XLSHNAME
- XLSETPROP

XLDISADD command

Disables an EXCEL add-in (enabled in the list: File, Options, Add-ins, Excel add-ins, Go).

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLDISADD')
CHGVAR          VAR(&PARM1) VALUE('Add-in')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Full path to add-in to disable.
Parm2	

Example

```
LNCCMD      CMD(XLDISADD) PARM1('C:\A\ModelMacro2.xlam')
```

See also

- XLADDINS
- XLENAADD

XLDPROTECT command

Disables the protection against modification applied on the active sheet.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLDPROTECT')
CHGVAR              VAR(&PARM1)VALUE('Password')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Password protection for the active sheet.
Parm2	

Example

```
CHGVAR      VAR(&CMD) VALUE('XLDPROTECT')
CHGVAR      VAR(&PARM1) VALUE('Aura')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
            &PARM2 &RESULT)
```

See also

- XLPROTECT

XLDRAWGR command

Creates a graph in a new sheet.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLDRAWGR')
CHGVAR          VAR(&PARM1) VALUE('selection')
CHGVAR          VAR(&PARM2) VALUE(' ''TYPE'' ''[ROWS|COLS]'' +
                ''Nb_Categorie'' ''Nb_Serie'' ''Name'' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Selection of cells containing the data or the data labels (co-ordinates separated by colons ":"). If several cells are specified, it is necessary to separate them with a semicolon (;) . **The numbers of characters used to specify selection has to be less than 260. Excel area has to be used to limit number of characters.**
Parm2	Other parameters This command can accept the following parameters : - **Type of graph** (AREA, BAR, COLUMN, LINE, RADAR, DONUT, XYSCATTER, 2DPIE, 3DAREA, 3DBAR, 3DCOLUMN, 3DLINE, 3DPIE). - **Arrangement of data** (ROWS or COLS). - **Number of lines or columns** of the source range containing the **categories**. - **Number of lines or columns** of the source range containing the **series.** - **Title of the graph**.

Example 1

Create un graph from the following Excel table :

	A	B	C

1	Années	ITEMSTOTAL	AMOUNTPAID
2	2001	162 730,05 €	162 730,05
3	2002	315 910,20 €	273 096,75
4	2003	162 379,40 €	162 379,40
5	2004	953 405,35 €	786 008,75
6	2005	967 225,30 €	935 238,30
7	2006	360 950,80 €	
8	2007	65,00 €	

```
PGM

DCL VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL VAR(&CMD) TYPE(*CHAR) LEN(10)
DCL VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL VAR(&PARM2) TYPE(*CHAR) LEN(1024)
DCL VAR(&RESULT) TYPE(*CHAR) LEN(512)

LNCOPEN

LNCCMD CMD(EXCELOPEN)

LNCCMD CMD(XLOPENFILE) +
PARM1('%LNCDIR%\samples\test.xls') +
PARM2(visible)

LNCCMD CMD(XLDRAWGR) PARM1('A1:C8') +
PARM2('''COLUMN'' ''COLS'' ''1'' ''1'' ''TEST1''')

LNCCMD CMD(EXCELSHOW)

LNCCMD CMD(END)

SNDPGMMSG MSGID(CPF9898) MSGF(QCPFMSG) +
MSGDTA(&RESULT) MSGTYPE(*INFO)

LNCCLOSE

ENDPGM
```

Exemple 2

```
  PGM

  LNCOPEN

  LNCCMD      CMD(EXCELOPEN)

  LNCCMD      CMD(XLOPENFILE) +
                PARM1('%LNCDIR%\samples\test.xlsx')

  LNCCMD      CMD(XLDRAWGR) PARM1('Sheet1!$A$1:C$4') +
```

```
              PARM2('''COLUMN'' ''COLS'' ''1'' ''1'' +
              ''TEST1''')
LNCCMD      CMD(XLMETHOD) +
              PARM1('Sheets.Item("Chart1").Select')
LNCCMD      CMD(EXCELSHOW)
LNCCLOSE
ENDPGM
```

Exemple 3: several cells used to define selection (limit of 260 characters)

```
LNCCMD      CMD(XLDRAWGR) +
              PARM1('Sheet1!$A$1;Sheet1!$A$2;Sheet1!$A$3;+
              Sheet1!$A$4;Sheet1!$A$5;Sheet1!$A$6;Sheet1!+
              $A$7;Sheet1!$A$8;Sheet1!$B$1;Sheet1!$B$2;Sh+
              eet1!$B$3;Sheet1!$B$4;Sheet1!$B$5;Sheet1!$B+
              $6;Sheet1!$B$7;Sheet1!$B$8;Sheet1!$C1;Sheet+
              1!$C$2;Sheet1!$C$3') PARM2('''COLUMN'' +
              ''COLS'' ''1'' ''1'' ''TEST1''')
```

Exemple 4: use "data" area

```
LNCCMD      CMD(XLDRAWGR) PARM1('Sheet1!data') +
              PARM2('''COLUMN'' ''COLS'' ''1'' ''1'' +
              ''TEST1''')
```

Exemple 5 :

```
CHGVAR              VAR(&CMD)VALUE('XLDRAWGR')
CHGVAR              VAR(&PARM1)VALUE(' A2:D3 ')
CHGVAR              VAR(&PARM2) +
                    VALUE(' ''3DPIE'' ''ROWS'' ''1'' ''0'' ''Market''
')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Note

Graphing functions in LAUNCHER are limited with XLDRAWGR.

To modify some properties of the graph, you can use the command **XLMETHOD** which allows to execute from the AS/400, Excel VBA methods :

```
LNCCMD CMD(XLMETHOD) PARM1('ActiveSheet.ChartObjects.Item("Graphique
1").Activate')
```

```
LNCCMD CMD(XLMETHOD)
PARM1('ActiveChart.ApplyDataLabels(xlDataLabelsShowValue)')
```

To get an idea of the methods to call, go to Excel in "Macro saving" mode: "**Tools**" menu - "**Macro**" - "**New macro**".

Do the desired operations on the keyboard and mouse.

Stop the macro recording, and see the code generated by Excel: Menu "**Tools**" - "**Macro**" - "**Macros**" - "**Edit**"

Example :

The macro that activates the chart and modifies a property is as follows:

```
ActiveSheet.ChartObjects("Graphique 1").Activate
ActiveChart.ApplyDataLabels Type:=xlDataLabelsShowPercent,
LegendKey:=True _
, HasLeaderLines:= False
```

Note : There are 2 differences to note between Excel VB syntax and LAUNCHER syntax:

1) When selecting an object from a collection, add ".Item" after the name of the collection.
To select "Graph 1" in the graphics collection, the VB syntax is :

```
ActiveSheet .ChartObjects("Graphique 1")
```

It becomes:

```
ActiveSheet .ChartObjects.Item("Graphique 1")
```

2) The parameter names (Name:= value) are specific to the VB syntax.

With LAUNCHER, it is necessary to list the values of each parameter, in the expected order, separated by ';'.

The following statement

```
ActiveChart.ApplyDataLabels Type:=xlDataLabelsShowPercent,
LegendKey:=True
```

Becomes :

```
ActiveChart.ApplyDataLabels(xlDataLabelsShowPercent;True)
```

Otherwise, choose to write a macro in Excel, which you will call by **XLEXEMACRO**.

XLENAADD command

Enables an available EXCEL add-in in the list (File, Options, Add-ins, Excel add-ins, Go).

An add-in can also be added to the list and activated with the **XLADDINS** command.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLENAADD')
CHGVAR          VAR(&PARM1) VALUE('Add-in')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Full path to add-in to activate.
Parm2	

Example

```
LNCCMD      CMD(XLENAADD) PARM1('C:\A\ModelMacro2.xlam')
```

See also

- XLADDINS
- XLDISADD

XLEXEMACRO command

Executes an EXCEL macro.

The macro can be present in the open workbook (.xls or .xlsm for example), or in an Excel add-in (.xla or .xlam), which can be loaded with the **XLADDINS** command.

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLEXEMACRO')
CHGVAR          VAR(&PARM1)VALUE('Macro name')
CHGVAR          VAR(&PARM2)VALUE('[Macro parameters]')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Macro name.
Parm2	Parameters of the macro. Parameters must be within doubles quotes, separated with semicolon.

Example

```
CHGVAR          VAR(&CMD)VALUE('XLEXEMACRO')
CHGVAR          VAR(&PARM1)VALUE('Macro1')
CHGVAR          VAR(&PARM2) VALUE('"12000";"FRANCE"')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

or

```
CHGVAR          VAR(&CMD)VALUE('XLEXEMACRO')
CHGVAR          VAR(&PARM1)VALUE('Macro')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

In this example, macro is contained in the active workbook.

Other example

Suppose we have the following macro, with two parameters:

```
Sub ProcMsgBox(sParam1 As String, sParam2 As String)

MsgBox sParam1 & " - " & sParam2

End Sub
```

If the macro is in the active workbook, then this code will be used to run the macro:

```
LNCCMD CMD(EXCELOPEN)
LNCCMD CMD(XLOPENFILE) PARM1('C:\A\temp.xlsm')
LNCCMD CMD(XLEXEMACRO) PARM1('ProcMsgBox') +
PARM2('"Aura";"Equipements"')
```

If the macro is in an XLAM add-in (or an XLA add-in), then we will use the following code to parse the add-in and then run the macro:

```
LNCCMD CMD(EXCELOPEN)
LNCCMD CMD(XLOPENFILE) PARM1('C:\A\temp.xlsx')
LNCCMD CMD(XLADDINS) PARM1('C:\A\ModelMacro.xlam')
LNCCMD CMD(XLEXEMACRO) PARM1('ProcMsgBox') +
PARM2('"Aura";"Equipements"')
```

See also

- XLMETHOD
- XLADDINS

XLEXISTSH command

Check if a sheet already exists in the active workbook.

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLEXIST')
CHGVAR          VAR(&PARM1)VALUE('SheetName')
CHGVAR          VAR(&PARM2)VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The name of the sheet to search.
RESULT	If the sheet exists, the value **TRUE** is returned. Otherwise, **FALSE** is returned. You can get the value by testing the RESULT variable.

Example

```
DCL         VAR(&RESULT) TYPE(*CHAR) LEN(512)

LNCOPEN

LNCCMD      CMD(EXCELOPEN)

LNCCMD      CMD(XLOPENFILE) PARM1('C:\A\temp.xlsx')

LNCCMD      CMD(EXCELSHOW)
LNCCMDR     CMD(XLEXISTSH) PARM1('tab') RESULT(&RESULT)
LNCCMD      CMD(NOP) PARM1(&RESULT)
```

See also

- CHKFILE

XLFIND command

Search for a value in the active sheet.

The first cell that contains the required value, is activated.

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLFIND')
CHGVAR          VAR(&PARM1)VALUE('value')
CHGVAR          VAR(&PARM2)VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The value to look for in the sheet.
RESULT	If the value is found, the value **TRUE** is returned. Otherwise, it is **FALSE**. The result passes through the return message from the PC (variable &RESULT).

Example

```
DCL         VAR(&RESULT) TYPE(*CHAR) LEN(512)

LNCOPEN

LNCCMD      CMD(EXCELOPEN)

LNCCMD      CMD(XLOPENFILE) PARM1('C:\A\temp.xlsx')

LNCCMD      CMD(EXCELSHOW)
LNCCMDR     CMD(XLFIND) PARM1('Aura') RESULT(&RESULT)
LNCCMD      CMD(NOP) PARM1(&RESULT)
```

See also

- XLREPLACE

XLFORMULA command

Writes a formula to a cell in an Excel workbook.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLFORMULA')
CHGVAR          VAR(&PARM1) VALUE('[cell coordinates]')
CHGVAR          VAR(&PARM2) VALUE('formula')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Coordinates of the cell to modify.** Optional. If Parm1 is not specified, the active cell will be taken into account. The coordinates of the cell can be given as following: - Syntax **$B$3**. Example: $B$3 is column B, line 3. - Syntax **CL**. **C** to designate the column and **L** to designate the line. Example: B3 designates line 3, and column 2 (B). - Symbolic name given to a cell.
Parm2	**Formula to execute.**

Example

```
CHGVAR          VAR(&CMD) VALUE('XLFORMULA')
CHGVAR          VAR(&PARM1) VALUE($A$2)
CHGVAR          VAR(&PARM2) VALUE ('=B8+E10')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

or

```
CHGVAR          VAR(&CMD) VALUE('XLFORMULA')
CHGVAR          VAR(&PARM1) VALUE('A2')
CHGVAR          VAR(&PARM2) VALUE ('=B8+E10')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

XLGETFILE command

Imports a text file into a sheet of an Excel workbook.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('XLGETFILE')
CHGVAR        VAR(&PARM1) VALUE('text file')
CHGVAR        VAR(&PARM2) VALUE('
              [Start=Number]
              [;RecordCnt= Number]
              [;Destination=Cell reference]
              [;MapColName=True/False]
              [;AutoFit=True/False]
              [;FormatValue=True/False]
              ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Path and name of the text file to import into the Excel sheet.**
Parm2	**Parm2 allows you to set read properties of the text file.** Properties are separated by semicolons. Property names can be in lowercase or uppercase letters. Properties: **Start**: to set the rank of the first line of the source file to insert. If the source file is from a transfer made by the **DBXFER** command, then it includes the names of the columns in the first line. This line will be ignored when inserting with "Start = 2". **RecordCnt**: to set the maximum number of lines to insert. **Destination**: Lets you give the cell reference at the top left of the destination range that will receive the data from the file. By default, the active cell will be the first receiving cell. This parameter is not taken into account if **MapColName = true**. If **MapColName** is true, the cells with the column names of the text file receive the values. The first line of the text file must contain the names of the columns. If **Autofit** is true, this allows you to adjust the width of the columns to

the content.

FormatValue:
True (default): The cell format is set according to the AS400 data type.
False: The format of the cells is fixed by Excel, according to the values.

Example

```
LNCCMD      CMD(XLOPENFILE) +
              PARM1('C:\A\Templates\SPCUST_template.xlsx')

LNCCMD      CMD(XLGOTOSH) PARM1('Invoice')

LNCCMD      CMD(XLGETFILE) +
              PARM1('File="C:\A\LNC002.TXT"') +
              PARM2('Start=2;FormatValue=True;Aut+
              oFit=False;RecordCnt=4;MapColName=true')

LNCCMD      CMD(XLGOTOSH) PARM1('Data')

LNCCMD      CMD(XLGETFILE) +
              PARM1('File="C:\A\LNC002.TXT"') +
              PARM2('Start=2;FormatValue=True;Aut+
              oFit=true;RecordCnt=4;Destination=$B$3')
```

In this example, the text file data (4 records), from the second record, are copied to the "Invoice" sheet, using data mapping based on their names.

In addition, the data from the text file (4 records), from the second record, are also copied to the "Data" sheet. The cell at the top left of the destination range is cell B3.

See also

- LNCTOXLS - CL Command

Description file (*.LFD)

The XLGETFILE command takes into account the LFD file, if it exists.

This file has the same name as the text file to be imported, but with the suffix "LFD".

The first line of the LFD file contains the name of the text file to import, without its extension, in brackets [].

[My file]

The following lines describe the format of each column of the text file to import.
There is one line in the LFD file per column of the text file.

Format of a description line : **N=Type,format**

N = Column number. The first column is number 0.

Type =

Num : For a numeric type column.

Format : **n.d**

n= Number of digits

d= Number of decimals

Text : For a text column.

Format : **n**

n is the number of characters (optional)

Date : For a date column.

Format :

DMY : Day/Month/Year

MDY : Month/Day/Year

YMD : Year/Month/Day

Skip : To ignore the column, and not to import it.

Example :

```
[LNC002]
0=Num,5.0
1=Num,5.0
2=Date,YMD
3=Date,YMD
4=Num,5.0
5=Text,7
6=Skip
7=Text,7
8=Num,8.2
```

XLGETPOS command

Returns the current position of the cursor in the active sheet.

The returned value is of the form **XXYYYY**.
XX being the number of the column (the letter).
YYYY being the number of the row.

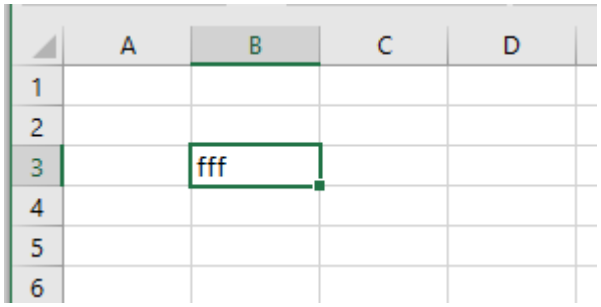
The result passes through the return message from the PC (variable &RESULT).

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLGETPOS')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) +
                    PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)
```

Example

Suppose the following cell is activated:



After running the following command:

```
LNCCMD CMD(XLGETPOS)
```

the &RESULT variable will contain the following value: $B$3.

XLGETPROP command

Reads the value of a property on the workbook, sheet, cell, or selection.

Syntax

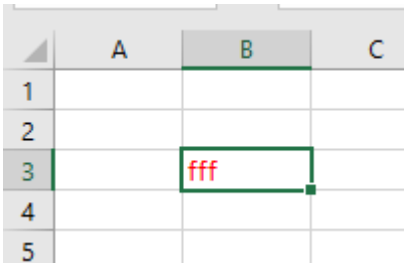
```
CHGVAR          VAR(&CMD) VALUE('XLGETPROP')
CHGVAR          VAR(&PARM1) VALUE('Property')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Full path to the property to read. Example : ActiveCell.Font.Color
RESULT	Receives the property value after calling.

Example

Suppose the following cell is activated:



After running the following command:

```
CHGVAR          VAR(&CMD) VALUE('XLGETPROP')
CHGVAR          VAR(&PARM2) VALUE('ActiveCell.Font.Color')
CHGVAR          VAR(&PARM1) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

the variable &RESULT will contain the following value: 255, which corresponds to the color code Red.

As a reminder, here are the Windows color codes:

Constant	Value	Description
vbBlack	&H0	Black
vbRed	&HFF	Red
vbGreen	&HFF00	Green
vbYellow	&HFFFF	Yellow
vbBlue	&HFF0000	Blue
vbMagenta	&HFF00FF	Magenta
vbCyan	&HFFFF00	Cyan
vbWhite	&HFFFFFF	White

See also

- XLSETPROP

XLGETVALUE command

Retrieves the value of a cell in the current sheet.

The value is sent back to the AS/400 by the return message from the PC (&RESULT variable).

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLGETVALUE')
CHGVAR          VAR(&PARM1)VALUE('coordinates')
CHGVAR          VAR(&PARM2)VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**The coordinates of the cell whose value we want to recover**. The coordinates can be given in the form: - Syntax **$B$3**. Example: $B 3 is column B, row 3. - Syntax **CL**. **C** to designate the column and **L** to designate the row. Example: B3 designates row 3, and column 2 (B). - Symbolic name given to a cell or group of cells. If the coordinates are not filled in, the active cell is taken into account.
RESULT	The result passes through the return message from the PC (variable &RESULT).

Example

To retrieve the value of the cell named "Sum":

```
CHGVAR          VAR(&CMD)VALUE('XLGETVALUE')
CHGVAR          VAR(&PARM1)VALUE('Sum')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)

/ * TO PRINT THE RETURN VALUE ON THE SCREEN */
SNDPGMMSG           MSGID(CPF9898) MSGF(QSYS/QCPFMSG) +
```

```
                                MSGDTA(&RESULT) TOPGMQ(*EXT)
MSGTYPE(*STATUS)
```

To recover the value of cell C4:

```
LNCCMD CMD(XLGETVALUE) PARM1('C4')
```

See also

- XLSETTEXT
- XLSETVALUE

XLGOTOCELL command

Moves the cursor position in the current sheet.

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLGOTOCELL')
CHGVAR          VAR(&PARM1)VALUE('coordinates')
CHGVAR          VAR(&PARM2)VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**The coordinates of the destination cell.** The coordinates can be given in the form: - Syntax **$B$3**. Example: $B$3 is column B, line 3. - Symbolic name given to a cell or group of cells.

Example

```
CHGVAR          VAR(&CMD)VALUE('XLGOTOCELL')
CHGVAR          VAR(&PARM1)VALUE('Sum')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

or

```
CHGVAR          VAR(&CMD)VALUE('XLGOTOCELL')
CHGVAR          VAR(&PARM1)VALUE('$B$3')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

See also

- XLCELLNAME
- XLSETTEXT
- XLSETVALUE

XLGOTOSH command

Activate a sheet in the current workbook.

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLGOTOSH')
CHGVAR          VAR(&PARM1)VALUE('name')
CHGVAR          VAR(&PARM2)VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The name of the sheet to activate. The name may have been set by a call to the XLSHNAME command.

Example

```
CHGVAR          VAR(&CMD)VALUE('XLGOTOSH')
CHGVAR          VAR(&PARM1)VALUE('Data')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

See also

- XLADDSHEET
- XLDELSHEET
- XLPRINTSH
- XLSHNAME

XLINSERT command

Inserts a cell , a range of cells, a row, a column into the worksheet and shifts other cells away.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLINSERT')
CHGVAR          VAR(&PARM1) VALUE('
                [Ref="Cell coordinates"]
                [;Sheet="Sheet name"]
                [;Row=Row number or displacement]
                [;Col=Column number or displacement]
                [;Shift=xlShiftDown/xlShifToRight]
                [;Count=number]
                [;EntireRow=True/False]
                [;EntireColumn=True/False]
                [;After=True/False]
                [;Fill=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 + &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Ref**: The reference to the cell can be given in the form: - Syntax **$B$3.** Example: $B$3 denotes column B, row 3. - Syntax **CL**. **C** to designate the column and **L** to designate the row. Example: B3 designates row 3, and column 2 (B). - Symbolic name given to a cell or group of cells. - "." will designate the active cell. **Sheet**: Specifies the name of the sheet to be addressed. If a cell name is indicated by **Ref**, then the **Sheet** keyword is not useful. Otherwise, if the **Sheet** keyword is missing, the active sheet is addressed. **Row, Col**: These two keywords represent respectively: - The coordinates (Line, Column) of a cell, if **Ref** is absent. - The relative coordinates in rows and columns from the base referenced by "Ref =" when it is present. The upper left cell has relative coordinates (1,1). **Shift**= *xlShiftToRight* to shift the actual cells to the right, or *xlShiftDown* (default) pour shift the cells down.

Count=Number of insertions to proceed.

EntireRow=*True* to insert entire rows, or *False* (default) to move cells down the width of the selection.
This keyword is valid only if **Shift**= *xlShiftDown.*

EntireColumn= *True* to insert entire columns, or *False* (default) to shift cells to the right, over the height of the selection.
This keyword is valid only if **Shift**= *xlShiftToRight.*

After=*True* to insert after the selection, *False* (default) to insert before the selection.

Fill=*True* to copy the selection to the set of new inserted cells., False (default) to keep the new cells emptys.
When **Fill** is true, **After** must also be true.

Examples

The following example inserts 1 entire row, at the current active cell position.

```
CHGVAR          VAR(&CMD) VALUE('XLINSERT')
CHGVAR          VAR(&PARM1) VALUE('EntireRow=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Insert 10 entire rows, after the 5th row ; Copy the 5th row over the 10 new rows.

```
CHGVAR          VAR(&CMD) VALUE('XLINSERT')
CHGVAR          VAR(&PARM1) +
                VALUE('Selection="$A$5";Count=10;EntireRow=True; +
                After=true;Fill=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

See also

- XLCELLS

XLINSERTCL command

Inserts a column to the left of the active cell in the current sheet.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLINSERTCL')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

See also

- XLINSERTRW

XLINSERTRW command

Inserts a line above the active cell in the current sheet.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLINSERTRW')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

See also

- XLINSERTCL

XLITALIC command

Puts the value in the cell into italic font.

Syntax

```
CHGVAR                VAR(&CMD)VALUE('XLITALIC')
CHGVAR                VAR(&PARM1)VALUE(' ')
CHGVAR                VAR(&PARM2)VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                      &PARM1 &PARM2 &RESULT)
```

See also

- XLBOLD
- XLUNDERLN

XLMAXIMIZE command

Maximizes the active EXCEL window.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLMAXIMIZE')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

See also

- XLMINIMIZE

XLMENU command

Creates a custom menu bar in Excel.

XLMENU must be used together with **XLADDINS** and **XLWAIT** commands.

Syntax

```
CHGVAR        VAR(&CMD) VALUE('WMENU')
CHGVAR        VAR(&PARM1) VALUE('
              Captions="Buttons labels"
              [;Tips="Tooltips"]
              [;Remove]
              ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Captions :** Lets you label each button of the new menu bar. The labels are separated from each other by a semicolon (;). The number of labels determines the number of buttons on the menu bar. **Tips :** Displays a tooltip for each button. The tips are separated from each other by a semicolon (;). Their number must match the number of labels. **Remove :** removes the previous menu.

Note

1)

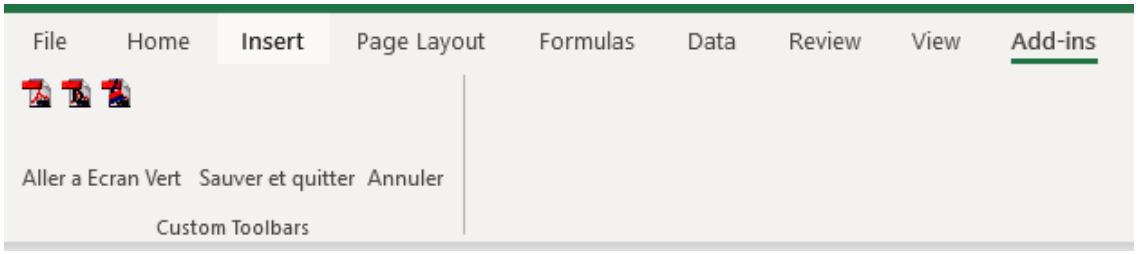
The add-in " **LNCExcelAddin.xla**" must be loaded by the **XLADDINS** command before using **XLMENU**.

When the **XLWAIT** command is called, the AS/400 program waits for an action on the part of the user. He takes again the hand when Word is closed, or on the action of one of the buttons of the new menu.

When returning from the **XLWAIT** command, the &RESULT variable contains the 5-digit number of the actuated button (from 1 to n).

2)

The custom menu will appear in the "Add-ins" tab of the Excel menu.



Example

```
CHGVAR        VAR(&CMD) VALUE('XLADDINS')
CHGVAR        VAR(&PARM1) VALUE('%LNCDIR%\LNCExcelAddin.xla')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)

CHGVAR        VAR(&CMD) VALUE('XLMENU')
CHGVAR        VAR(&PARM1) VALUE('Captions="To validate;To
              cancel;Tips="Validate document;to cancel document"')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)

CHGVAR        VAR(&CMD) VALUE('XLWAIT')
CHGVAR        VAR(&PARM1) VALUE(' ')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

XLMETHOD command

Calls method from Excel.Application object.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLMETHOD')
CHGVAR          VAR(&PARM1) VALUE('Method')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
Parm1	Full path of the method to call.

Note

The **XLMETHOD** command is used to execute methods of the Excel VBA language from the AS/400.

The contents of Parm1 respect the syntax used in Visual Basic.

It is possible to use all the constants of Excel and Visual Basic Application or the value of the constant itself.

To get an idea of the methods to call, go to Word in "Macro Recording" mode: **"Tools" menu - "Macro" - "New Macro".**

Do the desired operations on the keyboard and mouse.

Stop the macro recording, and see the code generated by Excel: **Menu "Tools" - "Macro" - "Macros" - "Edit".**

Example:

The macro that activates the chart and modifies a property is as follows:

```
ActiveSheet.ChartObjects ("Chart 1"). Activate
ActiveChart.ApplyDataLabels Type: = xlDataLabelsShowPercent,
LegendKey: = True _
, HasLeaderLines: = False
```

Note: There are 2 differences to note between the VB syntax of Excel and the LAUNCHER syntax:

1) When selecting an object from a collection, add ".Item" after the name of the collection.

To select "Graph 1" in the graphics collection, the VB syntax:

```
ActiveSheet.ChartObjects("Chart 1")
```

becomes

```
ActiveSheet.ChartObjects.Item("Chart 1")
```

2) The parameter names (Name: = value) are specific to the VB syntax.

With LAUNCHER, it is necessary to list the values of each parameter, in the expected order, separated by ';'
The following statement

```
ActiveChart.ApplyDataLabels Type: = xlDataLabelsShowPercent,
LegendKey: = True
```

becomes

```
ActiveChart.ApplyDataLabels(xlDataLabelsShowPercent; True)
```

Otherwise, choose to write a macro in Excel, which you will call by XLEXEMACRO.

Example

Below example activates the sheet 2 of Excel

```
CHGVAR          VAR(&CMD) VALUE('XLMETHOD')
CHGVAR          VAR(&PARM1) VALUE('Sheets.Item("Sheet2").Activate')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT))
```

See also

- XLEXEMACRO

XLMINIMIZE command

Minimizes the active EXCEL window.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLMINIMIZE')
CHGVAR              VAR(&PARM1)VALUE(' ')
CHGVAR              VAR(&PARM2)VALUE(' ')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

See also

- XLMAXIMIZE

XLOPENFILE command

Opens an EXCEL Workbook.

The Excel application must first have been opened with **EXCELOPEN**.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLOPENFILE')
CHGVAR          VAR(&PARM1) VALUE('File path')
CHGVAR          VAR(&PARM2) VALUE('
                [TEXT=True/False]
                [;ReadOnly=True/False]
                [;Visible=True/False]
                [;New=True/False]
                [;AddToWorkbook=True/False]
                [;Name="Sheet name"]
                [;PWD="password"]
                ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Paramètres

Parameters	
Parm1	**Path and name of the file to open**. If only the file is specified, it will be opened in the default directory of EXCEL documents.
Parm2	**Parm2 allows to set properties of for opening.** Properties : **Text**: to open the file in text mode. The file must have the same format as the one created by LAUNCHER Office (CSV format). The '* TEXT' option is also accepted for compatibility with older versions. **ReadOnly**: to open the file read-only. The option '*ONLY' is also accepted for compatibility with older versions. **AddToWorkbook**: to add the read file to the current workbook, as a new sheet. If no workbook is open, a new workbook is created. **Name**: allows you to name the new worksheet if the **AddToWorkbook** option is also selected.

	New: Creates a blank workbook. **Visible**: makes the instance of Excel visible. **PWD**: A string that contains the password required to open a protected workbook. If this argument is omitted and the workbook requires a password, the user is prompted for the password.

Note

By specifying the **Text** option, LAUNCHER Office will check if there is a LFD file, which would have been created during the file transfer. This repeats the description of the AS/400 file to reproduce it in the workbook.

Examples

```
CHGVAR        VAR(&CMD) VALUE('XLOPENFILE')
CHGVAR        VAR(&PARM1) VALUE('%LNCDIR%\Docs\fichier.xlsx')
CHGVAR        VAR(&PARM2) VALUE(' ')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

In this example, the file will be opened from the Launcher installation directory.

```
CHGVAR        VAR(&CMD) VALUE('XLOPENFILE')
CHGVAR        VAR(&PARM1) VALUE(' ')
CHGVAR        VAR(&PARM2) VALUE('New;Visible')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

In this example, a new workbook will be created and made visible.

```
LNCCMD     CMD(EXCELOPEN)
LNCCMD     CMD(XLOPENFILE) PARM1('C:\A\base.xlsx')
```

In this example, a workbook is opened after having created an instance of the Excel application.

```
LNCCMD     CMD(EXCELOPEN)
LNCCMD CMD(XLOPENFILE) PARM1('%TEMP%\LNC002.TXT') +
PARM2('AddToWorkbook;Name="Customers"')
```

In the example above, the file "LNC002.TXT" is added to the current workbook as "Customers".

```
LNCCMD     CMD(EXCELOPEN)
LNCCMD CMD(XLOPENFILE) PARM1('C:\A\protected_workbook.xlsx')
PARM2('PWD="aura"')
```

In the example above, we open a protected workbook by specifying the password.

See also

- XLCLOSFILE
- XLSAVEAS

XLPASTE command

Paste the content of the clipboard into the active cell.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLPASTE')
CHGVAR          VAR(&PARM2) VALUE(' ')
CHGVAR          VAR(&PARM1) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2+
                &RESULT))
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

See also

- XLCOPY

XLPRINT command

Prints an Excel workbook.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLPRINT')
CHGVAR          VAR(&PARM1) VALUE('
                [Printer="Nom_imprimante"]
                [;Copies="Nombre_de_copies"]
                [;From=Number]
                [;To=Number]
                [;Preview=True/False]
                [;PrintToFile="Chemin fichier"]
                [;Collate=True/False]
                [;Workbook=True/False]
                [;Sheet=True/False]
                [;Selection= True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Printer** is the name of the printer. If **Printer** is not specified, the default printer is used (see Windows Configuration Panel). **Copies** is the number of copies to print. **From** is the number of the first page to print. **From** refers to the printed pages, not overall pages of the sheet or workbook. **To** is the number of the last page to print. **To** refers to the printed pages and not overall pages of the sheet or workbook. **Preview** to preview print. **PrintToFile** prints to a file. You must specify the path and file name. **Collate** allows you to have collated copies. Organizes numbered pages when you print multiple copies of a document. A complete copy of the document is printed before printing the first page of the second copy. **Workbook** or **Sheet** or **Selection** to select the object to print.

Example

```
CHGVAR          VAR(&CMD) VALUE('XLPRINT')
CHGVAR          VAR(&PARM1) VALUE('Printer="Brother HL-1270N series";
                From=2;"To=8;Sheet')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Pages 2 to 8 of the current sheet are printed on the "Brother HL-1270N series" printer.

```
LNCCMD CMD(XLPRINT) PARM1('printer="LBP611C";Workbook')
```

The entire workbook is printed on the printer "LBP611C".

See Also

- WPRINT

XLPROTECT command

Protects the current sheet against changes with a password.

All cells that will be locked (cell format) will not be editable. A write attempt by the user causes a box to be displayed informing him that he can not access the contents of the cell or its properties.

Cells previously unlocked by the user will be editable.

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLPROTECT')
CHGVAR          VAR(&PARM1)VALUE('Password')
CHGVAR          VAR(&PARM2)VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	Password to protect the current sheet. To remember to remove the protection.

See also

- XLDPROTECT

XLREPLACE command

Find and replace text in the current Excel workbook.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLREPLACE')
CHGVAR          VAR(&PARM1) VALUE('
                ;Find="Searched text"
                ;ReplaceWith="Replacement text"
                [;MatchCase]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Find** is the text you are looking for. **ReplaceWith** is the replacement text. **MatchCase** indicates that the search will be case-sensitive.

Example

```
LNCCMD CMD(XLREPLACE) PARM1('Find="Aura";ReplaceWith="Equipements"')
```

Look for cells containing "Aura"/"aura" and replace "Aura"/"aura" with "Equipements".

```
LNCCMD CMD(XLREPLACE) +
PARM1('Find="Aura";ReplaceWith="Equipements";Matchcase')
```

Search (case sensitive) cells containing "Aura" and replacing "Aura" with "Equipements".

See also

- XLFIND

XLRESIZE command

This command inserts or deletes rows and columns, to give a new size to an area of an Excel sheet.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLRESIZE')
CHGVAR          VAR(&PARM1) VALUE('
                [Ref="Référence de cellules"]
                [;RowCount=number]
                [;ColCount=number]
                [;Rowcnt=number]
                [;Colcnt=number]
                [;EntireRow=True/False]
                [;EntireColumn=True/False]
                [;Clear=True/False]
                [;Fill=True/False]
                [;Name="Nouveau nom de zone"]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 + &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Ref**: The reference of the area to resize. 1) **Either we specify the name of the zone**. 2) **Either we specify the reference of the cell at the top left of the area to resize**. The cell reference at the top left can be given as: - Syntax **$B$3**. Example: $B$3 is column B, line 3. - Syntax **CL**. **C** to designate the column and **L** to designate the line. Example: B3 designates line 3, and column 2 (B). - Symbolic name. - "." to designate the active cell. In this case we can use **Rowcnt** and **Colcnt** to specify the reference of the cell at the bottom right of the area to resize. **RowCnt, ColCnt**: **Rowcnt** and **Colcnt** are used to specify the cell reference at the bottom right of the area to be resized, if the reference of the cell in the top left of the area to be resized has been specified with the **Ref** parameter. In this case, **Rowcnt** and **Colcnt** respectively represent the number of rows and columns selected, from the reference given by **Ref**. **RowCount**: Number of rows in the resized zone. If **RowCount** is absent, the number of rows in the field is not

changed.

ColCount: Number of columns in the resized zone.
If **ColCount** is absent, the number of columns in the zone is not changed.

EntireRow : *True* to insert or delete whole lines, or False to move cells up or down the width of the selection.

EntireColumn : *True* to insert or delete entire columns, or False (default) to shift cells to the left or right over the height of the selection.

Fill : *True* to copy the selection to all new inserted cells.
False (default) to leave the new cells empty.
This option allows you to copy the calculation formulas into the new inserted rows or columns.

Clear : *True* clears all cells that do not contain formulas in the new zone.

Name allows you to assign a name to the resized area. If **Ref** refers to a named zone, the same name can be given to Name.

Examples

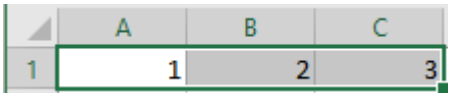
The following example resizes the area named "Orders" to 10 lines.
The number of columns in the zone does not change.
Entire lines are deleted or added.

```
CHGVAR          VAR(&CMD) VALUE('XLRESIZE')
CHGVAR          VAR(&PARM1) VALUE('Ref="Orders";RowCount=10;EntireRow=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

The area named "Orders" is resized to 10 lines.
The formulas are copied to the new lines.
Cells not containing formulas are erased.
Insertions or deletions are done only over the width of the area.

```
CHGVAR          VAR(&CMD) VALUE('XLRESIZE')
CHGVAR          VAR(&PARM1) +
                VALUE('Ref="Orders";RowCount=10; +
                Fill=True;Clear=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

The following area :



is resized with 10 lines, and named "Data":

```
LNCCMD CMD(XLRESIZE) +
PARM1('Ref="$A$1";RowCount=10;name="Data";Fill=false;Rowcnt=3;Colcnt
=3')
```

See also

- XLCELLS

XLSAVEAS command

Saves the current EXCEL workbook.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLSAVEAS')
CHGVAR          VAR(&PARM1) VALUE('[file]')
CHGVAR          VAR(&PARM2) VALUE('
                [Format=constant]
                [;Replace=True/False]
                [;Backup=True/False]
                ')
CHGVAR          VAR(&PARM2) VALUE('Options')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Path and save name of the current workbook.** If this parameter is empty, the document will be saved with the name under which it was opened.
Parm2	**Format** = format for saving the document. The format must be consistent with the extension of the saved file. You can use the Excel syntax: xlAddIn, xlCSV, xlCSVMac, xlCSVMSDOS, xlCSVWindows, xlDBF2, xlDBF3, xlDBF4, xlDIF, xlExcel2, xlExcel3, xlExcel4, xlExcel4Workbook, xlExcel5, xlExcel7, xlExcel9795, xlHTML, xlIntlAddIn, xlIntlMacro ... or the following keywords: - XLS - XLSX - XLSM Note: If you do not set the Format parameter, then the default MS Excel format is used. Example: XLSX for Excel 2016. **Replace** = True/False. False by default. Replace the file if it exists if **Replace**=True.

	Backup = True/False. False by default. True to create a backup file.

Examples

1)

```
LNCCMD CMD(XLSAVEAS) PARM1('c:\temp\res.xlsx') PARM2('format=XLSX')
```

2)

```
LNCCMD CMD(XLSAVEAS) PARM1('c:\temp\res.xlsm')PARM2('format=XLSM')
```

3)

```
CHGVAR        VAR(&CMD) VALUE('XLSAVEAS')
CHGVAR        VAR(&PARM1) VALUE('c:\chemin\sauvegarde.xlsx')
CHGVAR        VAR(&PARM2) VALUE('Replace')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)
```

See also

- XLADDSHEET
- XLDELSHEET
- XLGOTOSH

XLSETLINE command

Assigns values to multiple cells on the same row from the active cell.

Syntax

```
CHGVAR              VAR(&CMD)VALUE('XLSETLINE')
CHGVAR              VAR(&PARM1)VALUE('Values')
CHGVAR              VAR(&PARM2)VALUE('Properties')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**A string containing the values to assign to cells.** The string can contain the following symbolic word: **%SEP%**: To move to the next cell. If we were positioned on the last column of the table, we go to the beginning of the next line.
Parm2	**Character string that can contain cell formats.** The formats of each cell are separated by the word: **%SEP%**. Several formats can be applied to a cell. They are then separated by a semicolon. The formats can be: **NUMFMT(d [DECPOINT=p] [GRPPOINT=g])** with : -**d** is the number of decimals, -**p** represents the character to use for the decimal point. (Point character by default), -**g** is the character to use to separate groups of 3 digits. **PROP** sets the value of a property for the contents of the corresponding cell. Example: *PROP(Font.bold)=True; PROP(Font.italic)=True*

Examples

The example below sends the values for 4 cells, and sets the blue color to the second cell, the value of which is sent in numeric format, with 2 decimals.

```
CHGVAR      VAR(&CMD) VALUE('XLSETLINE')
CHGVAR      VAR(&PARM1) VALUE(&NAME *CAT '%SEP%' *CAT +
            &AMOUNT *CAT '%SEP%' *CAT &ADDRESS *CAT '%SEP%'+ *CAT &CITY)
CHGVAR      VAR(&PARM2) VALUE('%SEP%NUMFMT(2);PROP(Font.Color)= + INT(255) ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 + &PARM2 &RESULT)
```

The following example creates a new workbook and insert 2 rows from cell B2:

```
LNCCMD      CMD(EXCELOPEN) PARM1('visible;new')

LNCCMD      CMD(XLCELLS) PARM1('Ref="$B$2";select=true')

LNCCMD      CMD(XLSETLINE) +
              PARM1('Aura%SEP%Equipements%SEP%2019') +
              PARM2('%SEP%PROP(Font.Bold)=true;PROP(Font.+
              Color)=INT(255)%SEP%PROP(Font.Color)=vbBlue')

LNCCMD      CMD(XLINSERT) PARM1('After=true')

LNCCMD      CMD(XLSETLINE) +
              PARM1('Power8%SEP%IBM%SEP%2019') +
              PARM2('%SEP%PROP(Font.Bold)=TRUE;PROP(Font.+
              Color)=INT(255)%SEP%PROP(Font.Color)=vbBlue')
```

	A	B	C	D	E
1					
2		Aura	Equipeme	2019	
3		Power8	IBM	2019	
4					
5					

Note

To use the **Font.Color** property, it is necessary to use Windows color codes.
Color constants:

Constant	Value	Description
vbBlack	&H0	Black
vbRed	&HFF	Red
vbGreen	&HFF00	Green
vbYellow	&HFFFF	Yellow
vbBlue	&HFF0000	Blue
vbMagenta	&HFF00FF	Magenta
vbCyan	&HFFFF00	Cyan
vbWhite	&HFFFFFF	White

Constants values:

vbBlack	0
vbBlue	16711680
vbCyan	16776960
vbGreen	65280
vbMagenta	16711935
vbRed	255
vbWhite	16777215
vbYellow	65535

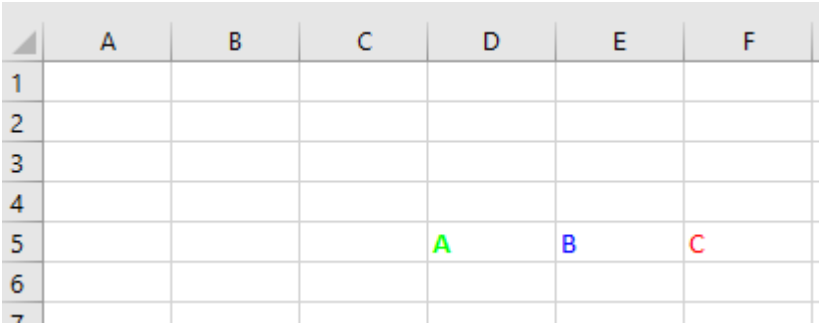
PROP(Font.Color)=vbMagenta
PROP(Font.Color)=INT(255) //red

You can also use BGR code with the RGB function:

PROP(Font.Color)=RGB(255,0,0) //blue
PROP(Font.Color)=RGB(0,255,0) // green
PROP(Font.Color)=RGB(0,0,255) // red

Example :

```
LNCCMD CMD(XLGOTOCELL) PARM1('$D$5')

LNCCMD CMD(XLSETLINE) PARM1('A%SEP%B%SEP%C') +
PARM2('PROP(Font.Color)=RGB(0,255,0);PROP(Font.Bold)=True+
%SEP%PROP(Font.Color)=RGB(255,0,0)%SEP%PROP(Font.Color)=RGB(0,0,255)'
)
```



XLSETPROP command

Sets the value of a property of various objects (workbook, sheet, cell, or selection) on an instance of Excel.

Syntax

```
CHGVAR          VAR(&CMD)VALUE('XLSETPROP')
CHGVAR          VAR(&PARM1)VALUE('Property path')
CHGVAR          VAR(&PARM2)VALUE('Value')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                &PARM1 &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**The full path to the property to edit.** Example: ActiveCell.Font.Color Some examples: ActiveCell.Font.Color: *Change the color* ActiveCell.Font.Bold: *Allows bold/non-bold selection.*
Parm2	**Value to assign to the property.** Examples: TRUE: True value. FALSE: False value. INT(string): When Excel expects an integer value.

Note

The path to the property follows the syntax used in Visual Basic.

You can use the constant values of Excel, such as xlBottom, xlVAlignCenter.

When the path to the property includes an element of a collection (such as ActiveCell.Borders ..., which has 4 borders), the item must be specified.

For example, in the case of borders, in Visual Basic, the property is expressed by:
ActiveCell.Borders(xlBottom).LineStyle = xlDashDot

In Visual Basic, Item is a default property that is not in the syntax.

Example

Red is set as font color of the active cell :

```
CHGVAR              VAR(&CMD)VALUE('XLSETPROP')
CHGVAR              VAR(&PARM1)VALUE('ActiveCell.Font.Bold')
CHGVAR              VAR(&PARM2)VALUE('INT(255)')
CALL                PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                    &PARM1 &PARM2 &RESULT)
```

XLSETTEXT command

Assign text to a cell in the current Excel worksheet.

The fomat cell is set as Text.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLSETTEXT')
CHGVAR          VAR(&PARM1) VALUE('Coordinates')
CHGVAR          VAR(&PARM2) VALUE('Text')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1
&PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Coordinates of the cell to modify.** The coordinates of the cell can be given as following: - Syntax **$B$3**. Example: $B$3 is column B, row 3. - Syntax **CL**. **C** to designate the column and **L** to designate the row. Example: B3 designates row 3, and column 2 (B). - Symbolic name given to a cell or group of cells. **If Parm1 is not specified, the active cell will be modified.**
Parm2	**Text to assign**.

Examples

```
CHGVAR          VAR(&CMD) VALUE('XLSETTEXT')
CHGVAR          VAR(&PARM1) VALUE('$A$1')
CHGVAR          VAR(&PARM2) VALUE('Dépenses')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

or

```
LNCCMD CMD(XLSETTEXT) PARM1('C7') PARM2('Aura')
```

or

```
LNCCMD CMD(XLSETTEXT) PARM1('Name') PARM2('Aura')
```

or (modification of active cell)

```
LNCCMD CMD(XLSETTEXT) PARM2('Aura')
```

See also

- XLCELLNAME
- XLGETVALUE
- XLGOTOCELL
- XLSETVALUE

XLSETVALUE command

Assign a value to a cell in the current Excel worksheet.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLSETVALUE')
CHGVAR          VAR(&PARM1) VALUE('Coordinates']
                ')
CHGVAR          VAR(&PARM2) VALUE('Value')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Coordinates of the cell to modify.** The coordinates of the cell can be given in the form: - Syntax **$B$3**. Example: $B$3 is column B, line 3. - Syntax **CL**. **C** to designate the column and **L** to designate the line. Example: B3 designates line 3, and column 2 (B). - Symbolic name given to a cell or group of cells. **If the coordinates of the cell are not filled in, the active cell will be modified.**
Parm2	**Value to be assigned.**

Example

```
CHGVAR                  VAR(&CMD) VALUE('XLSETVALUE')
CHGVAR                  VAR(&PARM1) VALUE('Sum')
CHGVAR                  VAR(&PARM2) VALUE('1000')
CALL                    PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
```

See also

- XLCELLNAME

- XLGETVALUE
- XLGOTOCELL
- XLSETTEXT
- XLCELLS

XLSHNAME command

Change the name of the current Excel sheet.

Syntax

```
CHGVAR                VAR(&CMD) VALUE('XLSHNAME')
CHGVAR                VAR(&PARM1) VALUE('Name')
CHGVAR                VAR(&PARM2) VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                      &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	New name of the current sheet.

Example

```
CHGVAR                VAR(&CMD) VALUE('XLSHNAME')
CHGVAR                VAR(&PARM1) VALUE('Data')
CHGVAR                VAR(&PARM2) VALUE(' ')
CALL                  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                      &PARM2 &RESULT)
```

or

```
LNCCMD CMD(XLSHNAME) PARM1('data')
```

See also

- XLADDSHEET
- XLDELSHEET
- XLGOTOSH

XLSORT command

Allows you to sort the cells of a sheet according to one or more columns.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLSUBTOTAL')
CHGVAR          VAR(&PARM1) VALUE('
                [;Sheet="Sheet name"]
                [;Key1=cell]
                [;Order1=xlDescending/xlAscending]
                [;Key2=cell]
                [;Order2=xlDescending/xlAscending]
                [;Key3=cell]
                [;Order3=xlDescending/xlAscending]
                [;Header=xlYes/xlNo]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	**Sheet**: Specifies the name of the sheet to process. If the Sheet keyword is missing, the active sheet is taken into account. All cells in the sheet will be sorted. **Key1, Key2, Key3**: Coordinates of the 1st cell of the column from which you want to sort. The coordinates of the cell can be given in the form: - Syntax **$B$3**. Example: $ B $ 3 is column B, line 3. - Syntax **CL**. **C** to designate the column and **L** to designate the line. Example: B3 designates line 3, and column 2 (B). - Symbolic name. **Key1 will be the priority sort order, followed by Key2 and Key3.** **Order1, Order2, Order3**: Sort order. Corresponds to Key1, Key2 and Key3 respectively. The values can be: **xlDescending**: Descending order of values. **xlAscending**: ascending order of values. Default value: xlAscending. **Header**: Specifies whether the sheet contains a row of column

	headers, so that it does not form part of the sort. **xlYes**: the sheet contains a row of column headers, and we want to exclude this line from sorting. **xlNo**: the sheet does not contain header lines, or we want the header line to be part of the sort. Default value.

Example

We want to sort the cells of the active sheet according to the values of column H (decreasing values), and then, the result of this first sorting will be ordered according to the values of column A (increasing values).

The first row of column headers will not be taken into account for sorting.

```
LNCCMD      CMD(EXCELOPEN) +
              PARM1('C:\A\sp_cust_ref.xlsx') +
              PARM2('visible')

LNCCMD      CMD(XLSORT) +
           PARM1('Key2="$A$2";Order2=xlAscending;Key1=+
           "$H$2";Order1=xlDescending;Header=xlYes')
```

XLSUBTOTAL command

Enables you to apply a row grouping function on columns in a selection.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLSUBTOTAL')
CHGVAR          VAR(&PARM1) VALUE('
                TotalList="list"
                [;Ref=" Selected cell reference"]
                [;Sheet="Sheet name"]
                [;RefTo="End of selected area reference"]
                [;GroupBy= Rank of column]
                [;Function=Function code]
                [;Replace=True/False]
                [;PageBreaks=True/False]
                [;SummaryBelowData=xlSummaryAbove/xlSummaryBelow]
                ')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Paramètres	
Parm1	**TotalList**: List of the numbers of the columns on which the function is to be applied (**Function** parameter). The columns are numbered from 1 to n. The numbers are separated by the character "semicolon", in a double-quoted list. **Mandatory**. **Ref**: The reference of the area to be treated. 1) **Either we specify the name of the zone.** 2) **Either we specify the reference of the cell at the top left of the area to resize**. The cell reference at the top left can be given as: - Syntax **$B$3**. Example: $B$3 is column B, row 3. - Syntax **CL**. **C** to designate the column and **L** to designate the row. Example: B3 designates row 3, and column 2 (B). - Symbolic name. In this case, **RefTo** is used to specify the reference of the cell at the bottom right of the area to be treated. **If Ref is absent, all cells in the sheet will be processed.** **Sheet**: Specifies the name of the sheet to be processed. If **Sheet** is missing, the active sheet will be processed. **RefTo**: Reference to the cell at the bottom right of the selected area. This reference is given in the form: - Syntax **$B$3**. Example: $B$3 is column B, row 3.

- Syntax **CL**. **C** to designate the column and **L** to designate the row. Example: B3 designates row 3, and column 2 (B).
- Symbolic name.

GroupBy: This option specifies the rank of the column to use to group summation. The columns are numbered from 1 to n.
Default value = **1**.

Function: Designates the function to apply.
The possible values are:
xlAverage, xlCount, xlCountNums, xlMax, xlMin, xlProduct, xlStDev, xlStDevP, xlSum, xlUnknown, xlVar, xlVarP.
See the Microsoft Excel documentation for more details on these features.
Default value: **xlSum**.

Replace: **False** not to replace the current subtotals.
By default, subtotals are replaced (True).

PageBreaks: To add a page break after each group.
Default value: False.

SummaryBelowData: Allows you to choose the position of subtotals.
The possible values are:
xlSummaryAbove to place subtotals before each data group.
xlSummaryBelow (Default) to place the subtotals after the group hits.

Example

In the following example, subtotals are created for column 12, with groupings made based on columns 1 and 6.

All the cells of the "data" sheet are taken into account.

```
PGM

LNCOPEN

LNCCMD CMD(EXCELOPEN)

LNCCMD CMD(XLOPENFILE) PARM1('C:\A\DataRef.XLSX')

LNCCMD CMD(XLGOTOSH) PARM1('data')

LNCCMD CMD(XLSUBTOTAL) +
PARM1('sheet="data";GroupBy=1;Function=xlSu+
m;TotalList="12"')

LNCCMD CMD(XLSUBTOTAL) +
```

```
PARM1('sheet="data";GroupBy=6;Function=xlSu+
m;TotalList="12";REPLACE=FALSE')

LNCCMD CMD(XLSAVEAS) +
PARM1('C:\A\RES.XLSX;replace=true')

LNCCMD CMD(EXCELCLOSE)

LNCCLOSE

ENDPGM
```

To apply subtotals only on a selection of cells, you can use the following command:

```
  LNCCMD      CMD(XLSUBTOTAL) +
                PARM1('Ref="A1";RefTo="N14";sheet="data";Gr+
                oupBy=1;Function=xlSum;TotalList="12"')
```

XLUNDERLN command

Puts the value in the cell into underlined font.

Syntax

```
CHGVAR                VAR(&CMD) VALUE('XLUNDERLN')
CHGVAR                VAR(&PARM1) VALUE(' ')
CHGVAR                VAR(&PARM2) VALUE(' ')
CALL                  PGM(LNCCMD) +
                      PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)
```

See also

- XLBOLD
- XLITALIC

XLWAIT command

Wait for an EXCEL event.

The AS/400 program waits for the closing of Excel (**EXCELCLOSE**), or an action on the custom menu (**XLMENU** command), to continue its execution.

Syntax

```
CHGVAR          VAR(&CMD) VALUE('XLWAIT')
CHGVAR          VAR(&PARM1) VALUE('[BackPrint=Value]')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

Parameters

Parameters	
Parm1	The **BackPrint** property allows you to wait for a background print to finish. Value is expressed in seconds. 0 = infinite.

Example

```
CHGVAR          VAR(&CMD) VALUE('XLWAIT')
CHGVAR          VAR(&PARM1) VALUE('BackPrint=10')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                &PARM2 &RESULT)
```

See also

- XLMENU
- EXCELCLOSE

CL Commands

General

It is possible to develop applications that only use high-level CL commands, without using API programs.

The conversation between the AS/400 program and the PC is handled by the CL commands, through the **EXESRV** and **ENDOPT** parameters of the CL commands.

The **EXESRV** parameter can take the values:

- ***DEV** to designate the PC used for terminal emulation.
- ***CURRENT** to designate the already open conversation. In this case, a conversation must be opened beforehand, by the **LNCOPEN** program, or by another CL command that kept its conversation open (**ENDOPT = *NONE**).
- **"IP Address/DNS Name"** to designate a PC by its IP address or name.

The **ENDOPT** parameter can take the values:

- ***NONE** to keep everything open: the document, the application and the conversation.
- ***DOC** to close only the document.
- ***APP** to close the document and application.
- ***ALL** to close everything, including the conversation.

Default values (WRKLNCDFT)

With the WRKLNCDFT command, you have access to all the default values used by the Launcher/400 client, when the *DFT value is used in a CL command:

LNCDIR Folder used by WORD by default
LNCFLR Document Path
LNCPRN Path + name of generated document for PRN outputs
LNCPRT Printer used by default by Launcher/400
LNCROOT Default value for the LNCROOT parameter
LNCSAV Path to save documents (LNCPRTDOC)
LNCSEL Path used for ctrl prompt LNCPRTDOC
LNCSRV Server used by default by the LNCPRTDOC command
LNCTFR Path to folder of transferred files
LNCTMP Name and path to the temporary transfer file
MDBPTH Path to documents ACCESS
MDBSRV Server used by default by the CPYTOMDB command
MSGLOG Path and name of the log file for SMTP
MSGRPL Return email address
MSGSMT SMTP server used by the LNCSNDMAIL command
MSGSND Email address used by default for mailings
MSGSRV Launcher server used by the LNCSNDMAIL command
XLSPTH Document path used by the CPYTOXLS command
XLSSAV Path to save spreadsheets
XLSSRV Server used by default for CPYTOXLS

CPYTOMDB - CL Command

The **CPYTOMDB** command is used to transfer data from a file or request from the AS/400 to a table in an ACCESS database.

Parameters

Access Database selection type	**SELMDB**	F
Target .mdb file	**TOMDB**	
Target .mdb path	**MDBPTH**	*DFT
Datasource	**CPYSRC**	*FILE
Source filename	**FROMFILE**	
Library		*LIBL
Source member	**FROMMBR**	*FIRST
SQL query	**FROMSQL**	'SELECT * FROM'
Query	**FROMQRY**	
Library		*LIBL
Target table name	**TOTBL**	
Add/Replace records	**TBLOPT**	*ADD
Show result	**SHOWDOC**	*YES
Unicode transfer	**EXESRV**	*DFT
Ending option	**ENDOPT**	*NONE

Details

The **CPYTOMDB** command is used to launch the **ACCESS** application on a P.C. to make different copies in the tables of the active database.
Data can be added to existing records
in a table or replace the contents of the table.
The ***MAP *DROP** copy technique of CPYF is used every time.

Automatic Access database choice (SELMDB)

Allows to select the Access database with a visual interface on Windows.
This parameter is mandatory.
Possible values are :

F

(FREE) to enter the database name, and path manually.

S

(SELECT) to open the standard Windows explorer, to parse your directories, and select the database.
Notice : The value **S** is only usable in an interactive command line. When writing a CL program with SEU, only value **F** can be used.

Target database (TOMDB)

Specifies the Access database name to create or use.
The database Access file will be create if the database specified in the TOMDB does not exist.
Notice : The .mdb must not be specified
Possible values are :

name-of-database

Specifies a database name to create or use.

***CURRENT**
The database that was opened during last LAUNCHER Office session will be used.
This option is useful when the program needs to make several copies into different tables of the same database.
The *CURRENT value is only applicable if the LAUNCHER Office connection was not closed since the last CPYTOMDB command.

Database access path (MDBPTH)
Specifies the database access path.
Possible values are :
***DFT**
The special value *DFT means that the default LAUNCHER Office value will be used for this parameter.
To display or change the default value, select the **MDBPTH** keyword by calling the **WRKLNCDFT** command.
***NONE**
Means that the full path is given in the **TOMDB** parameter.
access-path
The full access path to find the database, with the following syntax :
C:\Program Files\Launcher400\My databases
Notice : The C:\My Documents\Launcher400 is correct, but the C:\My Documents\Launcher400\ syntax is not correct.

Copy Datasource (CPYSRC)
Indicates the origin of the source to copy.
Possible values are:
***FILE**
The data to transfer into the database are in database file (LF or PF).
***QRY**
The data to transfer into the database are in a query (object type *QRYDFN).
***SQL**
The data to transfer into the database are in selected by a SQL **SELECT**.

Database File (FROMFILE)
Specifies the name of the file and the library to use to get the data.
This parameter is used when parameter **CPYSRC** is set to ***FILE**.
The database file can be physical or logical.
Possible values for the library are :
***LIBL**
The search is performed in all libraries user and system from the list of work libraries until the first occurrence is found.
***CURLIB**
The database file is searched in the library during the work, if it is not specified.

File member (FROMMBR)
Enter the name of the member that contains the data.
This parameter is used when parameter **CPYSRC** is set to ***FILE**.
Possible values are :
***FIRST**

The first member of the database file contains the data.

Member-name

Enter the name of the member that contains the data.

SQL request (FROMSQL)

Enter the SQL request to retrieve the records from the AS/400 database.
This parameter is used when parameter **CPYSRC** is set to ***SQL**.

Query (FROMQRY)

Specifies the name of the query to use to get the data.
This parameter is used when parameter **CPYSRC** is set to ***QRY**.

Query-name

Enter the name of the query.

Possible library values are :

***LIBL**

The library list is searched to find the library where the query file is located.

Library-name

Enter the name of the library that contains the query file.

Destination access table (TOTBL)

Specifies the table name in the Access database file.
If the table does not exists, it will be created.

Add or replace records (TBLOPT)

Specifies how to add the records in the Access Table.

Possible values are :

***ADD**

Simply add the records into the table.

***REPLACE**

Replace all records with the new ones.

Show result (SHOWDOC)

Specifies if we must show the table result or not.

Possible values are :

***NO**

Do not show the result table.

***YES**

Show the result table.

Unicode Transfert

*YES (by default) ou *NO.

LAUNCHER Office server to use (EXESRV)

Specifies the name or IP address of the PC to use to build the document.

Possible values are :

***DFT**

The value from the keyword **MDBSRV** in file **LNCDFTP** is used.

***DEV**

The data transfer is performed on the PC where the terminal for the current job is located.

***CURRENT**

When *CURRENT is specified, the current job must be already connected to a LAUNCHER OfficeServer on a PC. That connection will be used.

IP address or name

The data transfer is performed on the specified PC.
Notice : The LAUNCHER Office server application must be running on the PC.

Ending Option (ENDOPT)

Specifies what object needs to be closed at the end of the process.
Possible values are :

***ALL**

The following elements are closed: Access Database, Access application, LAUNCHER Office connection between the job and the PC.
The next CPYTOMDB command will not be able to use the value ***CURRENT** for any parameter.

***APP**

The following elements are closed: Access Database, Access application.
The LAUNCHER Office connection between the job and the PC is still active.
The next CPYTOMDB command will be able to use the value ***CURRENT** for the parameter EXESRV.

***DOC**

The following elements are closed: - Access Database.
The Access application, and the LAUNCHER Office connection are still active.
The next CPYTOMDB command will be able to use the value ***CURRENT** for the parameter EXESRV.

***CON**

Only the LAUNCHER Office connection between the job and the PC is closed. Word and the document remain in memory.

***NONE**

The connection, Access, and the document are left active.
The next CPYTOMDB command will be able to use the value *CURRENT for the parameters EXESRV and DOC.

Example

Description: COPYING A PF AS/400 FILE TO MS ACCESS
The table named "TABLE1" is used for copying.
Note that the path to the database is indicated at
TOMDB parameter and must not be specified in the **MDBPATH** parameter
If the database already exists, the data in the table is overwritten.
If the database does not exist, it is automatically created and the
TABLE1 table is added.

```
PGM

CPYTOMDB SELMDB(F) TOMDB('%LNCDIR%\SAMPLES\R_MDB') +
MDBPTH(*NONE) FROMFILE(SP_CUST) +
TOTBL('TABLE1') TBLOPT(*ADD) UNICOD(*YES) +
EXESRV(*DEV) ENDOPT(*APP)

LNCSHELL CMD('%LNCDIR%\Samples\S_LNCSQL.MDB') +
VISIBLE(*YES) ACTION(OPEN) EXESRV(*CURRENT)

ENDPGM
```

CRTPRNSPLF - CL Command

The **CRTPRNSPLF** command creates an AS/400 spool file from a PRN file that is on the IFS.
A PRN file is a file that can be generated from any Windows program using the print command of that program and checking 'into file' in the print dialog box.
This PRN file can also be a result of LAUNCHER Office commands (for example LNCPRTDOC).
The PRN file can be transferred into the IFS using the IFSPUT LAUNCHER Office command.

Parameters

Name of the PRN file in IFS	**FILE**	
Spool filename	**SPLFNAME**	Value alpha
User name	**USER**	Value alpha, *CURRENT
Output queue	**OUTQ**	Name, *JOB
Library	***LIBL**	Name, *LIBL, *CURLIB
From type	**FORMTYPE**	Value alpha, *STD
Copies	**COPIES**	1-255
Hold spooled file	**HOLD**	*NO, *YES
Save spooled file	**SAVE**	*NO, *YES
Output priority (on OUTQ)	**OUTPTY**	1-9, *JOB
User reference	**USRDTA**	Value alpha
User text		
Type de données imprimante		*USERASCII
Rotation des pages		*NO

Details

Name of the PRN file in the IFS (FILE)

Specifies the name of the PRN file in the IFS that will be used to create the spool file.
This parameter is mandatory.

Spool file name (SPLFNAME)

Specifies the spool file name to create on the AS/400.
Possible values are:
LNCPRNPRT
The LNCPRNPRT spool file name is used by default.
spool-file-name
Enter a spool file name to create (10 maximum characters).

User name (USER)

Specifies the creator of the spool file.
Possible values are :
***CURRENT**

Use the active user profile.
user-name
Choose an AS/400 user profile.

Output queue (OUTQ)
Specifies the full qualified name of the output queue.
The possible values are :
***JOB**
The output queue associated with the job will be used.
output-queue
Enter an output queue.
The possible library values are :
***LIBL**
The library list is searched to find the library where the output queue is located.
***CURLIB**
The output queue is searched in the current library of the job.
Library-name
Enter the name of the library that contains the output queue.

Form type (FORMTYPE)
Specifies the paper type.
Possible values are:
***STD**
The standard form type is used.
form-type
Any form type.

Copies (COPIES)
Specifies the number of copies for the spool file.
This value must be between 1 and 255.

Hold spooled file (HOLD)
Specifies to hold the spool file or not.
Possible values are:
***NO**
Do not hold the spool file.
***YES**
Hold the spool file. See RLSSPLF to release the spool file.

Save spooled file (SAVE)
Specified to keep the spool file (in the output queue) after print or not.
Possible values are :
***NO**
Do not keep the spool file after print.
***YES**
Keep the spool file in the output queue after print.

Output priority (on OUTQ)
Specifies the output priority for the spooled file.

The highest priority level is 1, and the lowest priority level is 9.
Possible values are :
***JOB**
Use the current job output priority.
output-priority
The output priority value, between 1 (highest priority level) and 9 (lowest priority level).

User data (USRDTA)
Specifies a user-defined reference to identify the spool file.
This reference is a 10 characters text.

User text (USRTXT)
Specifies a 100 characters text to comment the spool file.

Printer data type (DEVTYP)
*USERASCII (default)
*SCS
*LINE
*AFPDS
*AFPDSLINE
*IPDS

Rotating pages
*YES
*NO (default)

Example

The following example prints a document to a PRN file, sends the PRN file to the IFS, and then forwards it to a spool file.

```
LNCPRTDOC     DOC('C:\Modeles\layout.doc')
              MRGTYPE(*FILE)
              FROMFILE(SP_LAYOUT)
              OUTPUT(*OUTPRN)
              OUTPRN('C:\temp\layout.prn')
              ENDOPT(*APP)
CHGVAR        VAR(&CMD) VALUE('IFSPUT')
CHGVAR        VAR(&PARM1) VALUE('C:\Temp\layout.prn')
CHGVAR        VAR(&PARM2) VALUE('/QDLS/IMAGES/Layout.prn')
CALL          PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
              &PARM2 &RESULT)

CRTPRNSPLF    FILE('/QDLS/IMAGES/Layout.prn')
              SPLFNAME(PRNSPLF)
              USRTXT('Document Word')

RMVLNK        OBJLNK('/QDLS/IMAGES/Layout.prn')
```

```
CALL          PGM(LNCCLOSE)
```

LNCCHK - CL Command

LNCCHK is used to test the connection between the AS400 and the LAUNCHER Office server on the Windows side.

It also checks the launch of Word and Excel, and displays all versions of the products in a window.

This command does not work with Launcher in Windows service mode.

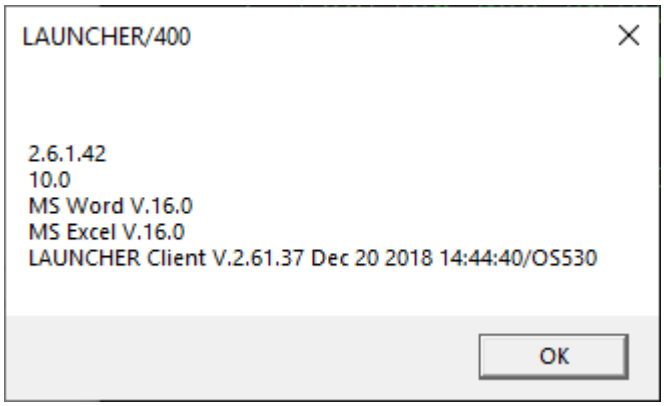
Parameters

```
Name of PC hosting LAUNCHER server : *DEV by default
```

Example

```
LNCCHK EXESRV(*DEV)
```

Result:



LNCPRTDOC - CL Command

The **LNCPRTDOC** command is used to compose a document from a Word template, and data from a file or request from the AS/400.

The following operations can be processed with this command :

- Template and data merging
- Merged document saving
- Merged document printing
- Document display

Using Word documents templates (*.dot, *.dotx) is recommended.

The online help available on the AS400 can be more complete for the LNCPRTDOC command.

Parameters

Document Name for the template	**DOC**	
Folder of the template	**FLR**	*DFT
Root Access path	**ROOT**	*DFT
Data source type for the merge	**MRGTYPE**	*NONE
Source filename	**FROMFILE**	
Library		*LIBL
Source member	**FROMMBR**	*FIRST
SQL query	**FROMSQL**	'SELECT * FROM'
Query	**FROMQRY**	
Library		*LIBL
Output option	**OUTPUT**	*
Save the resolved document	**OUTSAVE**	*NO
Show the resolved document	**SHOWDOC**	*YES
LAUNCHER PC Server to use	**EXESRV**	*DFT
Ending Option	**ENDOPT**	*NONE
Word Default directory	**WORDPATH**	*DFT
Exécuter le publipostage	**EXECMRG**	*YES
Suppression des champs	**NLINK**	*YES

Details

The **LNCPRTDOC** command merges a Word document template with data from the AS/400 database.

Document (DOC)

Specifies the document template name.

Possible values are :

Document_name

Specifies the document name.

Note : It is possible to set the full path to the document in this parameter.

In that case, parameter FLR and ROOT must be set to *NONE.

A full path to a network directory can be set.
(Example : \\MyServer\MyShare\MyDir\MyDoc.docx).

***NEW**
A new empty Word document will be created.

***CURRENT**
The actual call to LNCPRTDOC will use the current document.
The current document was opened by a previous call to LNCPRTDOC,
or any other LAUNCHER command that did not close the document.

Folder (FLR)

Specifies the folder or directory path, where the document is located.
Note : When the path is specified in the parameter
DOC this parameter must be set to *NONE.
The character '\' must not appear at the end.

Root path (ROOT)

Specifies the path to the model directories or saved resolved documents.
Note: This parameter is optional. If specified, this parameter is used to build the path to the model document (DOC) and to the document to be backed up (SAVDOC).
The '\' character must not appear at the end of the entered path.

A network path in the form \\Server\Share\Directory can be specified for the ROOT parameter.

Merge Type (MRGTYPE)

Specifies the type of the data source for the mail merge.
Possible values are :

***NONE**
No data source is specified. The document will be displayed or printed, without any merge of data.

***QRY**
The data to merge with the template are in a query (object type *QRYDFN).

***FILE**
The data to merge with the template are in database file (LF or PF).

***SQL**
The data to merge with the template are selected by a SQL request.

***DOC**
The data source is defined in the Word template.

Query (QRYDFN)

Specifies the name of the query to use to get the data.
This parameter is used when parameter MRGTYPE is set to *QRY.

Query-name

Enter the name of the query.
Possible library values are :

***LIBL**
The library list is searched to find the library where the query file is located.

Library Name
Enter the name of the library that contains the query file.

Database File (FROMFILE)
Specifies the name of the file and the library to use to get the data.
This parameter is used when parameter **MRGTYPE** is set to ***FILE**.
The database file can be physical or logical.
Possible values for the library are :
***LIBL**
The search is performed in all libraries
user and system from the list of work libraries
until the first occurrence is found.
***CURLIB**
The database file is searched in the library
during the work, if it is not specified.

File member (FROMMBR)
Enter the name of the member that contains the data.
This parameter is used when parameter **MRGTYPE** is set to ***FILE**.
Possible values are :
***FIRST**
The first member of the database file contains the data.
Member-name

SQL request (FROMSQL)
Enter the SQL request to retrieve the records from the AS/400 database.
This parameter is used when parameter **MRGTYPE** is set to *SQL.

Output unit (OUTPUT)
Specifies if the resolved document will be printed or displayed.
Possible values are :
*****
The document will not be printed. It will be displayed if the parameter SHOWDOC is set to the value *YES.
***PRINT**
The document will be printed.
***OUTPRN**
The document will not be printed to a file (PRN).
*** PDF**
The document will be converted to PDF. The **OUTPDF** parameter must be populated with the path and the name of the PDF file to generate.
***E-MAIL**
Each letter will be sent to an email recipient. Fill in the MAILOPT parameter to indicate the subject and the name of the column of the file containing the addresses of the recipients.
This feature is only available with an OUTLOOK messaging client.

Printer file (OUTPRN)
Specifies the path and name of the file to print to, when the parameter **OUTPUT** is set to the value *OUTPRN

Save the resolved document (OUTSAVE)

Specifies if the resolved document is saved.
Possible values are :
***NO**
The resolved document is not saved.
***YES**
The resolved document will be saved. The parameter **SAVDOC** must be set.

Show the document (SHOWDOC)
Specifies if the document is made visible on the Windows screen
Possible values are :
***NO**
The resolved document is not visible.
***YES**
The resolved document is visible.

Launcher server to use (EXESRV)
Specifies the name or IP address of the PC to use to build the document.
Possible values are :
***DFT**
The value from the keyword LNCSRV in file LNCDFTP is used.
***DEV**
The mail merge is processed on the PC where the terminal for the current job is located.
***CURRENT**
When *CURRENT is specified, the current job must be already connected to a LAUNCHER Server on a PC. That connection will be used.
IP address or name
The document is built on the specified PC.
Note : The LAUNCHER server application must be running on the PC.

Ending Option (ENDOPT)
Specifies what object needs to be closed at the end of the process.
Possible values are :
***ALL**
The following elements are closed: Word Document, Word application, LAUNCHER Connection between the job and the PC.
The next LNCPRTDOC command will not be able to use the value *CURRENT for any parameter.
***APP**
The following elements are closed: Word Document, Word application.
The LAUNCHER Connection between the job and the PC is still active.
The next LNCPRTDOC command will be able to use the value *CURRENT for the parameter EXESRV.
***DOC**
The following elements are closed: Word Document.
The Word application, and the LAUNCHER Connection are still active.
The next LNCPRTDOC command will be able to use the value *CURRENT for the parameter EXESRV.
***CON**
Only the LAUNCHER connection between the job and the PC is closed.
Word and the document remain in memory.
***NONE**

The connection, Word and the document are left active.
The next LNCPRTDOC command will be able to use the value *CURRENT for the parameters EXESRV and DOC.

Document name to save (SAVDOC)
Specifies the name for the new document to save after the mail merge.
Note : This parameter can contain the full directory path.
The file extension must be set.
Example : MyResultLetter.docx

Folder for the saved document. (SAVFLR)
Specifies the folder or directory where the resolved document will be saved.
Possible values are :
***DFT**
The default value from the file LNCDFTP, for the keyword LNCSAV will be used.
Folder_name
The path to the folder or directory is specified.
Note : The character '\' must not be typed at the end of the string.
***NONE**
Specifies that the parameter **SAVDOC** contains the directory path.

WORD merge type (MRGSEL)
Specifies the merge type that Word will process.
Possible values are :
***LTR**
The word merge is a letter.
A letter is built for each record in the data source. The letters are separated from each other by a "Form Feed".
***CAT**
The Word merge is a catalog.
Only one document is built, with all the records from the data source.
***NONE**
The merge type is set in the Word template.

Default Word directory (WORDPATH)
Specifies the default directory for Word during the mail merge process.
Possible values are :
***DFT**
The value from the keyword LNCDIR in file LNCDFTP is used.
Default_directory
Set the full path to the default directory.
***NONE**
The default directory is not changed. It is set in the Word options.

Temporary file for transfer (DOCTMP)
The data are transferred to a temporary file on the PC. The file name and path for this file can be set.
Possible values are :
***DFT**
The value from the keyword LNCTMP in file LNCDFTP is used.

Temporary_file_name
Set the path and name for the temporary file to use.

Word merge options (WORDMRG))
Several options are available.
Header file
Specifies the name of a file containing the fields names for the data source.
Possible values are :
***FILE**
The fields names come from the AS/400 database file.
Header-file-name
Enter a PC file name containing the names of the fields.
Suppress blank lines
In the document, when a line containing merge fields is blank after the mail merge, Word can suppress it.
Possible values are :
***NO**
The blank lines are kept in the resolved document.
***YES**
The blank lines are deleted from the resolved document.

Run the mail merge (EXECMRG)
Specifies if the mail merge process is to be run or not.
Possible values are :
***YES**
The mail merge is processed. The data and the template are merged together. A resolved document is built.
***NO**
The mail merge is not processed. The final document is attached to the data source. The document is ready for editing.

Example

```
LNCPRTDOC SELDOC(F) DOC('%LNCDIR%\Samples\sp_cust.docx') FLR(*NONE)
ROOT(*NONE) MRGTYPE(*FILE) FROMFILE(SP_CUST) SHOWDOC(*YES)
EXESRV(*DEV)
```

LNCSHELL - CL Command

The command **LNCSHELL** executes a DOS command, opens a document, or executes a program on a Windows station.

Parameters

Command, Document or Exe file.	**CDM**	
Directory for the Doc. ou Exe	**FLR**	*NONE
Parameters for the exe prog.	**PARMEXE**	*NONE
Wait for the end of execution	**WAITCMD**	*NO
Return back the ERRORLEVEL	**ERRORLVL**	*YES
Run a visible mode	**VISIBLE**	*NO
Placer la fenêtre en 1er plan	**FOCUS**	*YES
Minimiser la fenêtre active	**MINCURWIN**	*NO
Action on the document or Exe.	**ACTION**	*NONE
Default directory.	**WRKFLR**	*NONE
LAUNCHER PC Server to use	**EXESRV**	*DFT
End options	**ENDOPT**	*ALL

Details

Command, Document or Exe (CMD)

Enter : A DOS command, a document to open, or an executable program name.

Directory for the Doc. or Exe (FLR)

If the parameter **CMD** is a document or an executable program, then **FLR** can be the path to that file.

Notice : This parameter is optional if the full path is given with the parameter **CMD**.

A network path is valid: \\Serveur\Partage\repertory.

Parameters for the executable prog (PARMEXE)

If the parameter **CMD** is an executable program, then parameters to pass to the program can be specified.

Wait for the end execution (WAITCMD)

Possible values are :

***NO**

The program does not wait for the command to be completed.

***YES**

The program waits for the command to be completed on the PC.

Value

The job waits for the closing of the called program on the PC, up to a maximum delay expressed in milliseconds.

If the timeout has expired, the message LNC0701 is sent to the job, with the value 99999 in the first 5 characters.

Several programs, like Internet Explorer, does not allow to wait for the completion.

Return back ERRORLEVEL (ERRORLVL)

This option is valid only when **WAITCMD** is set to ***YES**.

Possible values are :

***YES**

If a return code other than 0 is returned, the message LNC0701 is sent to the program.

The first 5 characters of the message represent the error number.
The value 99999 is returned when the delay expressed by **WAITCMD** has been exceeded.

***NO**

The return of the command call is not tested.

Run in visible mode (VISIBLE)

Possible values are :

***NO**

The command execution is hidden.

***YES**

The command execution is visible.

Place the window in the foreground (FOCUS)

The possible values are:

***YES**

The application window launched on the PC will be placed in the foreground of windows active on Windows.

***NO**

The active window will remain unchanged.

This keeps the terminal emulation window always active.

Minimize the active window (MINCURWIN)

The possible values are:

***YES**

The active window is minimized.

If WAITCMD is not ***NO**, the window will be restored at the end of the execution of the called program.

***NO**

The active window remains displayed in its normal size.

Action on the document or Exe (ACTION)

This parameter is not used when **CMD** is a DOS command.

Possible values are :

OPEN

This value must be set when **CMD** is an executable program name.

When **CMD** is a document, the value OPEN allows to open the document on Windows, using its default application.

PRINT

When **CMD** is a document, the value PRINT allows to print the document on Windows.

Default directory (WRKFLR)

Set the default directory during the execution of the command or program.

Launcher server to use (EXESRV)

Specifies the name of the server on which the command is to run.

This name may correspond to another workstation.

The host name or the IP address are valid choices.

Possible values are :

***DFT**

The value from the keyword **LNCSRV** in file **LNCDFTP** is used.

***DEV**

The command is processed on the PC where the terminal for the current job is located.

***CURRENT**

When *CURRENT is specified, the current job must be already connected to a LAUNCHER Server on a PC.

That connection will be used.

adresse IP or hôte name

The command is processed on the specified PC.

Notice : The LAUNCHER server application must be running on the PC.

Ending Option (ENDOPT)

Specifies what object needs to be closed at the end of the process.

Possible values are :

***ALL**

The LAUNCHER communication between the job and the PC server will be closed.

The program will not be able to reuse keyword *CURRENT for parameter EXESRV for the next call.

***NONE**

LAUNCHER communication remains active.

The program will be able to reuse keyword *CURRENT for parameter EXESRV for the next call.

Example

Creating and opening a PDF file:

```
PGM

LNCOPEN

LNCCMD CMD(WORDOPEN) PARM1(NEW)

LNCCMD CMD(WTYPETEXT) PARM1('CECI EST UN FICHIER QUI A ETE +
CONVERTI EN FICHIER PDF')

LNCCMD CMD(PDFPRINTER) +
PARM1('FILE="C:\TEMP\TEMP.PDF"')

LNCCMD CMD(WPRINT) PARM1('PRINTER="LAUNCHER_PDF"')

LNCCMD CMD(WORDCLOSE)

LNCSHELL CMD('C:\TEMP\TEMP.PDF') WAITCMD(*NO) VISIBLE(*YES) +
ACTION(OPEN) EXESRV(*DEV)

LNCCLOSE

ENDPGM
```

LNCSNDMAIL - CL Command

LNCSNDMAIL allows to send electronic messages and controls the main features :

recipients (TO, CC, BCC), object, message text, attachments , priority level, acknowledgement.
Messaging system must be MAPI compatible (as Outlook or Exchange), otherwise Lotus Notes or a SMTP server can be used.

Notice : It is not necessary that the messaging system be currently running on the station to send messages.

Parameters

Attached files + if others values	**DOC**	*NONE
Attached files folder	**FLR**	*NONE
To recipient(s):	**TOINTNET**	
Messaging address		
Recipient type		*TO
+ if others values		
Subject	**SUBJECT**	
Message	**MSG**	*NONE
Priority	**PTY**	*NORMAL
Reception confirmation	**CFMDEL**	*NO
Launcher server name	**EXESRV**	*DFT
Ending option		*ALL
Messaging system type	**MSGTYPE**	*SMTP
SMTP Parameters:		
Messaging address of sender	**CFGSMTP**	*DFT
Return address		*DFT
SMTP server adress or name		*DFT
SMTP port number		*DFT
Trace of message being sent		*NO
SMTP user profile		*NONE
User password		*NONE
Enable SSL		*NO
Timeout		*NONE
Number of trials		*NONE
Logfile path		*DFT
SSL protocol		TLS
Signature		*NO
Encrypt email		*NO
Certificate path		*NONE
Password certificate		*NONE
Secure headers		*NO

Wrapper subject	*NONE
Check email	*NO

Details

Attached files (DOC)
Specifies the PC filename of each attachment.
This parameter allows multiple values.
To enter more values, enter a '+' sign in the "+ if other values" line, and type ENTER.

Attached files folder (FLR)
Specifies the main folder for attached files.
If a full path is specified in the **DOC** parameter this parameter must be set to ***NONE**.

Recipient(s) (TOINTNET)
Specifies the messaging address of the destination address(es).
Up to 100 messaging addresses can be specified.
This parameter allows multiple values.
To enter more values, enter a '+' sign in the "+ if other values" line, and type ENTER.

Messaging address
Specify a messaging address of a recipient, in the messaging system syntax.
Can contain up to 253 characters.
Type of recipient
Specifies the kind of recipient that was specified.
Possible values are :
***TO**
This is a main recipient.
***CC**
This is a CC (Carbon Copy) recipient.
***BCC**
This is a BCC (Blind Carbon Copy) recipient.
The other recipients will not see the existence of this recipient.

Subject (SUBJECT)
Message subject. Can contain up to 44 characters.
This parameter is mandatory and must not begin with blanks.

Message (MSG)
Main message text.
Possible values are:
message
The message text. Can contain up to 5000 characters.
To put line breaks, use %PARA%.
***NONE**
Use *NONE special value to generate a message with no body.

***DOCMSG**
The *DOCMSG special value indicates that the body will be a document specified in the DOCMSG parameter.
Notice : The *DOCMSG value is only valid when using the SMTP messaging system.

Priority (PTY)
Specifies the priority level of the message (low, normal or high).
Possible values are:
***NORMAL**
The message has a normal priority level.
***HIGH**
The message has a high priority level.
***LOW**
The message has a low priority level.

Reception confirmation (CFMDEL)
Specifies if the sender wants to receive a return receipt.
Possible values are :
***NO**
The sender will not ask a return receipt.
***YES**
The sender asks a return receipt.
This return receipt will be sent by the recipients if they receive the message and accept to return receipts.

LAUNCHER server name (EXESRV)
Specifies the name or IP address of the PC to use to build the document.
Possible values are :
***DFT**
The value from the keyword MSGSRV in file LNCDFTP is used.
***DEV**
PC where the terminal for the current job is located.
***CURRENT**
When *CURRENT is specified, the current job must be already connected to a LAUNCHER Office Server on a PC.
That connection will be used.
IP address or name
Processing is performed on a network PC, referred to here as its hostname, or TCP / IP address.
Notice : The LAUNCHER Office server application must be running on the PC.

Messaging system type (MSGTYPE)
Speficies the messaging system type.
***SMTP**
Messaging system is SMTP (Simple Mail Transfer Protocol).
***MAPI**
Messaging system is MAPI (For example Exchange or Outlook).
***LOTUS**
Messaging system is Lotus Notes.

SMTP parameters SMTP (CFGSMTP)

These are mandatory parameters when using SMTP messaging type.

Email address of sender

Specifies the email address of the sender.

Possible values are:

***DFT**

The sender value is specified in the MSGSND keyword of LNCDFTP file.

email-address

Choose an email address for the sender.

Return address

Specifies the email return address.

Name of SMTP server

Specifies the IP address or the name of the SMTP server.

SMTP port number

Port number of the SMTP server. This is 25 by default.

Trace of sent emails

Default is ***NO**.

***YES**: Ensures that the message is sent.

The file is formatted by delimiters (;), to be imported into a database or Excel.

Use WRKLNCDFT to specify the path and name of the logfile.

SMTP user profile

SMTP user profile.

SMTP user password

SMTP user password.

Enable SSL

Default value: *NO.

SSL is used to access the specified SMTP mail server.

TLS is used by default. Change the SSL Protocol setting if you want to use the SMTPS protocol.

Timeout

Defines the waiting time for sending an email. By default, no delay is specified (*NONE). It is the processing time of the SMTP server once the connection is established. Specify a delay in seconds.

Number of trials

Indicate the number of attempts to send an e-mail if the SMTP server is not reachable. By default, no new attempts (*NONE).

Indicate a number of retries.

Logfile path

If the trace of the sent emails is activated (*YES), by default the path of the logfile (*DFT) is defined with the command WRKLNCDFT. You can specify a path here other than *DFT.

SSL protocol

By default: the TLS protocol (STARTTLS command) is used.

Port #587 is generally used with the TLS protocol.

You can specify the SMTPS protocol, also called SMTP/SSL, SMTP over SSL, or SMTPS, and typically uses port #465.

Sign email

Put *YES to sign the email. The certificate path and password are required.

Default: *NO.

Encrypt (encrypt) email

The body of the email is encrypted according to PKCS #7. The certificate path and password are required. Default: *NO.

Certificate path

Path of the certificate (.p12 file) used to sign and/or encrypt the email.

Certificate password
Password of the certificate (.p12 file) used to sign and/or encrypt the email.

Secure headers
Protection of headers through the use of a RFC 822 message.
A message/rfc822 wrapper is used to apply S/MIME security services to headers.
See RFC 8551 for more details. The email will be received as an attachment by the recipient.

Wrapper subject
If SecureHeaders=*YES, by default the recipient will receive an email without subject and an attachment containing the specified email.
You can specify a subject here in order to prevent the email from going into spam.

Check email
Put *YES to proceed to a verification of the recipients' email addresses, according to the RFC 5322 standard, before sending. Default: *NO.

Example

```
LNCSNDMAIL TOINTNET(('tech@easycom-aura.com' *CC) +
('INFO@EASYCOM-AURA.COM')) SUBJECT('ENVOI DE MAIL') +
MSG('Ceci est un test d''envoi de mail avec LAUNCHER Office') +
CFGSMTP('TECH@EASYCOM-AURA.COM''TECH@EASYCOM-AURA.COM' +
'183.145.89.12')
```

Other example

HTML file as body mail, sent with SMTP. Delivy report and email sent logfile activated.

```
LNCSNDMAIL TOINTNET(('tech@easycom-aura.com') +
             ('sales@easycom-aura.com' *BCC)) +
             SUBJECT(&OBJET) MSG(*DOCMSG) CFMDEL(*YES) +
             EXESRV(*CURRENT) +
             CFGSMTP('info@easycom-aura.com' +
             'info@easycom-aura.com' +
             'smtp.aura.fr' 0 *DFT *YES) +
             DOCMSG('C:\A\R_11WORD.HTML')
```

LNCTOXLS - CL Command

The **LNCTOXLS** command is designed to transfer data from the AS/400 into an Excel sheet.

LNCTOXLS command requires the following parameters:

Parameters

Workbook used as template	**TOXLS**	
Workbook Folder/Directory	**XLSPTH**	*DFT
Data source type	**CPYSRC**	*FILE, *QRY, *SQL, *NONE
File name	**FROMFILE**	
Library		Nom, *LIBL, *CURLIB
File member	**FROMMBR**	Nom, *FIRST
Columns selection	**COLUMNS**	*ALL
Data types conversions	**TYPECAST**	*NONE
Name of Excel sheet	**TOSHEET**	*ONLY
Replace/Add/Insert rows	**TONAME**	*NONE
	XLSOPT	*REPLACE, *ADD, *INSERT
Output type	**OUTPUT**	*, *PRINT
Saved file name	**SAVDOC**	*NONE
Afficher le document	**SHOWDOC**	*YES, *NO
LAUNCHER server name	**EXESRV**	*DFT
Ending options	**ENDOPT**	*ALL, *APP, *DOC, *CON, *NONE
Ajuster largeur des colonnes	**AUTOFIT**	*YES, *NO
Appliquer Format aux cellules	**FMTCELLS**	*YES, *NO
Insérer entêtes de colonnes	**ADDCOLH**	*NONE, *COLHDG, *NAME, ALIAS
Appliquer Format automatique	**AUTOFMT**	*NONE

Details

Excel workbook name (TOXLS)

Specifies the excel workbook name to use as a model.
This parameter is mandatory.
Possible values are:
name-of-Excel-workbook
Designates an Excel workbook name.
***CURRENT**
The command will use the last workbook opened by LAUNCHER Office.
***NEW**
The command will use a new empty workbook.

Workbook access path (XLSPTH)

Speficies the acces path for opening the Excel workbook.
Possible values are:
***DFT**
The special value *DFT means that the default LAUNCHER will be used for this parameter.
To display or change the default value, select the **XLSPTH** keyword (**WRKLNCDFT** command).
***NONE**
Means that the full path is given in the **TOXLS** parameter.
Path
The full access path to find the workbook.

Copy Datasource (CPYSRC)
Specifies the type of the data source for the mail merge.
The possible values are:
***FILE**
The data to transfer into the workbook are in database file (LF or PF).
***QRY**
The data to transfer into the workbook is a query (object type *QRYDFN).
***SQL**
The data to transfer into the workbook comes from a SQL request.

Database File (FROMFILE)
Specifies the name of the file to use to get the data.
This parameter is used when parameter **CPYSRC** is set to ***FILE**.
The possible values for library are :
***LIBL**
The library list is searched to find the library where the file is located.
***CURLIB**
Library-name
Enter the name of the library that contains the file.

File member (FROMMBR)
Enter the name of the member that contains the data.
This parameter is used when parameter MRGTYPE is set to *FILE.
Possible values are :
***FIRST**
The first member of the database file contains the data.
Member-name
Enter the name of the member that contains the data.

SQL request (FROMSQL)
Enter the SQL request to retrieve the records from the AS/400 database.
This parameter is used when parameter **MRGTYPE** is set to ***SQL**.

Query (FROMQRY)
Specifies the name of the query to use to get the data.
This parameter is used when parameter **MRGTYPE** is set to ***QRY**.
Query-name
Enter the name of the query.
The possible library values are :

***LIBL**
The library list is searched to find the library where the query file is located.
Library-name
Enter the name of the library that contains the query file.

Name of the Excel sheet (TOSHEET)
Specifies the name of the Excel sheet used in this command.
The possible values are:
***ONLY**
The *ONLY special value means that the command will use the first available and active sheet of the Excel workbook.
Remarque : Prefer this option when using a one-sheet Excel workbook.
***AUTOCRT**
The *AUTOCRT special value means that the command will insert a new sheet into the workbook and transfer the data into it.
Excel-sheet-name
Specifies a named destination sheet.

Name of excel result (TONAME)
Specifies the name of the new range created by the command.
A name can be a standard string or a range with the following form:
$Columnletter$linenumber
Example : TONAME($D$3) designates the cell of column 4, line 3.

Add / Replace or Insert rows (XLSOPT)
Specifies how to add the records in the Excel workbook.
Possible values are :
***REPLACE**
Replace all records with the new ones.
***ADD**
Simply add the records into the sheet.
***INSERT**
Rows are inserted to receive the new data from the data source.
Parameter **ENTIRER** indicates if entire rows are inserted or not.
***RESIZE**
The new data replaces the old data.
The area in the workbook is resized.
Lines will be deleted or inserted.
***MAP**
Column values in the data source are assigned to cells
with the same names in the Excel table.
Only the first recording of the data source is used.

Output device (OUTPUT)
Specifies to view or print the result.
Possible values are:
*****
Shows the result.
***PRINT**
Prints the result into the default printer of the workstation.

Saved file name (SAVDOC)
Specifies the name of the final saved workbook.
Notice : If there is no access path, the file will be saved into the default saving directory.
Possible values are:
***NONE**
The result will not be saved.
Excel-filename
The result will be saved into a specified name.

Workbook saving path (SAVFLR)
Speficies the saving folder for the final result.
Possible values are:
***DFT**
The default value from the file LNCDFTP, for the keyword **XLSSAV** will be used.
Folder_name
The path to the folder or directory is specified.
Notice : The character '\' must not be typed at the end of the string.
***NONE**
Specifies that the parameter **SAVDOC** contains the directory path.

Saved file format (SAVFMT)
Specifies in the saving format.

Show document (SHOWDOC)
Specifies to show the result or not.
Possible values are:
***NO**
The result is not shown.
***YES**
The result is shown.

Launcher server to use (EXESRV)
Specifies the name or IP address of the PC to use to build the document.
Possible values are :
***DFT**
The value from the keyword **XLSSRV** in file **LNCDFTP** is used.
***DEV**
The data transfer is performed on the PC where the terminal for the current job is located.
***CURRENT**
When *CURRENT is specified, the current job must be already connected to a LAUNCHER Server on a PC.
That connection will be used.
IP address or name
The data transfer is performed on the specified PC.
Notice : The LAUNCHER server application must be running on the PC.

Ending Option (ENDOPT)
Specifies what object needs to be closed at the end of the process.
Possible values are :

***ALL**
The following elements are closed: Excel workbook, Excel application, LAUNCHER Connection between the job and the PC.
The next LNCTOXLS command will not be able to use the value *CURRENT for any parameter.
***APP**
The following elements are closed: Excel workbook, Excel application.
The LAUNCHER Connection between the job and the PC is still active.
The next **LNCTOXLS** command will be able to use the value ***CURRENT** for the parameter **EXESRV**.
***DOC**
The following elements are closed: Excel workbook.
The Excel application, and the LAUNCHER Connection are still active.
The next LNCTOXLS command will be able to use the value *CURRENT for the parameter EXESRV.
***CON**
Only the LAUNCHER Connection between the job and the PC is closed.
Excel and the document remain in memory.
***NONE**
The connection, Excel, and the document are left active.
The next **LNCTOXLS** command will be able to use the value ***CURRENT** for the parameters **EXESRV** and **DOC**.

Adjust the columns width (AUTOFIT)
Adjust the column width with the size of the contents.
Possible values are:
***YES**
Adjust the column width with the maximum size of values.
***NO**
Do not change the column width.

Apply an automatic format (AUTOFMT)
Set the automatic format to apply to the destination area.
Possible values are:
***NONE**
No automatic format is applied.
Name
The specified format is applied.
Read the Excel technical guides to know the existing automatic formats.

Set the cells format (FMTCELLS)
Indicates if the format of the cell is to be set depending on the column data type of the database.
Possible values are :
***YES**
The format of the cell is set depending on the database data types.
***NO**
The format is set by Excel depending on the type of value.

Insert column headings (ADDCOLH)
Insert the column headings or field names in the first row.
Possible values are:

***NONE**
No column heading inserted.
***NAME**
The field names are inserted as the first row.
***COLHDG**
The column heading from the database are inserted.
***ALIAS**
The aliases (long names) are inserted.

Insert entire rows (ENTIRER)
If **XLSOPT** is set to value ***INSERT**, this option indicates if entire rows are inserted or not.
Possible values are :
***YES**
Entire rows are inserted to receive the new data.
***NO**
The cells behind the range selected (parameter **TONAME)** are moved dows.

Example

```
LNCTOXLS TOXLS(MODELE.XLSX) +
XLSPTH('%LNCDIR%\SAMPLES') CPYSRC(*SQL) +
FROMSQL('SELECT * FROM SP_ORD') +
TOSHEET(FEUIL1) +
TONAME($A$2) XLSOPT(*ADD) +
SAVDOC(R_RESULTATS) +
SAVFLR('%LNCDIR%\SAMPLES') ENDOPT(*DOC) +
AUTOFIT(*YES)
```

LNCXFER - CL Command

LNCXFER allows to transfer data from a file or an AS/400 request to a fine on PC (ASCII or Unicode format).

Parameters

Target PC file path	**PCFILE**	
Folder/Directory of file	**FLR**	*NONE
Data source type	**CPYSRC**	*FILE
Source file name	**FROMFILE**	
Library		*LIBL
Source member	**FROMMBR**	*FIRST
Columns to select from file		*ALL
Data type conversions		*NONE
Maximum number of records	**MAXRCD**	*NOMAX
Generate UNICODE data	**UNICOD**	*NONE
Generate column headings	**COLHDG**	*NAME
Decimal point	**DECPT**	*LOCALE
Generate desc. file (lfd)	**DESCF**	*NO
Format of the destination file	**OUTFMT**	*CSV
Column delimiter character.	**DELIMITER**	*DFT
Double quotes around values	**QUOTETXT**	*YES
LAUNCHER server name	**EXESRV**	*DFT
Close communication	**CLOSECOM**	*YES

Example

```
LNCXFER PCFILE(TRANFERT.TXT) FLR('c:\temp\') +
FROMFILE(LAUNCHER/SP_CUST) EXESRV(*DEV)
```

LNCFTP – CL Command

The "FTP" license extension is mandatory.

The LNCFTP command "allows to address a FTP server from a CL client program, on AS400 side.

It calls the various functions of the LNCFTP DLL.

The command performs the following actions :

- open connection with the Launcher server
- management of the SSL connection or not
- send commands to the FTP server
- get data from the FTP server
- close connection with the Launcher server

An application on IBM I commands receiving/sending files from/to an FTP server, or send commands to the FTP server securely.

For this, the Secure FTP protocol (FTP over TLS/SSL) is used.

The use of SSL allows to require the use of a Certificate to secure communication between client and server.

Parameters

FTP server	**RMTSYS**
Port	**PORT**
Secure connection	**SECCNN**
FTP user	**USER**
FTP password	**PWD**
FTP command	**CMDFTP**
Remote path	**SRVPATH**
Local path	**LOCALPATH**
LAUNCHER server	**EXESRV**
End option	**ENDOPT**

Example

```
LNCFTP      RMTSYS('milan01') PORT(990) SECCNN(*SSL) +
               USER(aura) PWD(aura) CMDFTP(*MKDIR) +
               SRVPATH(CED) EXESRV('192.168.15.84') +
               ENDOPT(*NONE)
```

Java classes

The class DataSource

Attributes	Description
String Type	Peut prendre les valeurs suivantes : • csv: data source is a CSV file. Complete path of the CSV file is filled into Path attribute. • oracle: data source from Oracle database • mysql: data source from MySQL database • postgresql: data source from PostgreSQL database • sybase: data source from Sybase database • as400db2: data source from IBM DB2 for iSeries database • sqlserver: data source from Microsoft SQL Server database • saptable: data source from table of SAP system • sapbapi: data source from BAPI result table
String SrvAddr	The IPv4 address of the server hosting the database : Oracle, MySQL or PostgreSQL. Example: 192.168.1.7
String Port	The port number of the server hosting the database : Oracle, MySQL or PostgreSQL. Example: 1521 for Oracle 3306 for MySQL 6078 for PostgreSQL
String User	Name of the database user.
String Password	Password of the database user.
String DBName	Database name. Example: XE for Oracle test for MySQL newold for PostgreSQL
String Query	SQL query. Example: select * from test.marci where title > 'g'
String Path	Complete path of the CSV file when Type = csv. Example: c:\\temp\\source.csv
String ClientSAP	SAP client. Example: 001
String LangSAP	Language of SAP system.
String AshostSAP	Host name of a specific SAP application server. Example: 192.168.1.7
String GwhostSAP	SAP gateway host on which the server should be registered Example: 192.168.1.7

String GwservSAP	SAP gateway port. Example: 3300
String SysnrSAP	SAP instance number of the server. Example: 00
String TableNameSAP	Name of SAP system table. Example: SBOOK
String RowskipsSAP	Number of records from the SAP table selection, which is not taken into account Example: If 50 records are returned from the search in the SAP table, and that RowskipsSAP = 10, then the output CSV file only contains 40 lines. The first 10 records are not taken into account.
String RowcountSAP	Maximum number of potential records returned from a selection in a SAP table. Example 1: If RowcountSAP = 100, and 200 records correspond to the selection, then the result CSV file will contain only 100 lines. Example 2: If RowcountSAP = 100, and 70 records match the selection, then the CSV file will contain 70 lines.
String FoiSAP	Table fields selected in the query. Corresponds to SELECT within SQL query. Example: CITYFROM,CITYTO,DISTANCE,ARRTIME,DEPTIME
String QuerySAP	Selection condition in the query.Correspond to WHERE within SQL query. The value to compare has to be between single ribs. Example: CITYFROM = 'ROME'
String BAPIName	BAPI name (Business API (Application Program Interface)) or "RFC enabled functions module." If BAPIName is entered, the following fields are not taken into account when executing the BAPI: TableNameSAP, RowskipsSAP, RowcountSAP, FoiSAP, QuerySAP. Example: BAPI_FLBOOKING_GETLIST
String BAPIResultTable	The result of the execution of the BAPI is in different tables. If BAPIName is specified, it must also be specified BAPIResultTable. Example: For the BAPI "BAPI_FLBOOKING_GETLIST", one result table is: BOOKING_LIST.
Map<String, String> ParameterMap	The field is optional. It is a map containing the name and the value of the parameter of the BAPI Example: For the BAPI "BAPI_FLBOOKING_GETLIST", one of the parameters is: MAX_ROWS. In the Java code, a hm map is created and parameters are put into hm. Then this collection is assigned to the ParameterMap parameter. Map<String,String> hm = new HashMap<String,String>();

	`hm.put("MAX_ROWS", "70");` `myDataSource.setParameterMap(hm);`

All attributes are accessed and modified through Getters and Setters methods.

Attributes	Getters(read)	Setters (write)
`String Type`	getType()	setType(String)
`String SrvAddr`	getSrvAddr()	setSrvAddr(String)
`String Port`	getPort()	setPort(String)
`String User`	getUser()	setUser(String)
`String Password`	getPassword()	setPassword(String)
`String DBName`	getDBName()	setDBName(String)
`String Query`	getQuery()	setQuery(String)
`String Path`	getPath()	setPath(String)
`String ClientSAP`	getClientSAP()	setClientSAP(String)
`String LangSAP`	getLangSAP()	setLangSAP(String)
`String AshostSAP`	getAshostSAP()	setAshostSAP(String)
`String SysnrSAP`	getSysnrSAP()	setSysnrSAP(String)
`String TableNameSAP`	getTableNameSAP()	setTableNameSAP(String)
`String RowskipsSAP`	getRowskipsSAP()	setRowskipsSAP(String)
`String RowcountSAP`	getRowcountSAP()	setRowcountSAP(String)
`String FoiSAP`	getFoiSAP()	setFoiSAP(String)
`String QuerySAP`	getQuerySAP()	setQuerySAP(String)
`String BAPIName`	getBAPIName()	setBAPIName(String)
`String BAPIResultTable`	getBAPIResultTable()	setBAPIResultTable(String)
`Map<String, String> ParameterMap`	getParameterMap()	setParamterMap(Map<String, String>)

The class LNCPRTDOC

Attributes	Description
String LNCSrvAddr	The IPv4 address of the Launcher server Example: 192.168.1.3
String Doc	Template name. Example: C:\\temp\\template.docx
String WordPath	Default directory of Word.
String MrgType	Can take the following values : • *NONE : no merge • *FILE : data source to merge comes from CSV file or database.
String Unlink	Can take the following values : • True : deleting blank fields • False : none
String OutSave	Can take the following values : • True : save the generated document • False : none
String MrgSel	Merge format : • *LTR : letter format • *CAT : catalog format
String OutPdf	Complete path of the PDF file to save.
String OutPrn	Complete path of the PRN file to save.
String ExecMrg	Can take the following values : • True : exécution of the mail merge • False : the document is only prepared for the mail merge
String Printer	Printer name
String Copies	Number of generated copies
String Tray	Name of the tray
String FirstPAgeTray	Number or name of the tray
String OtherPageTray	Number or name of the tray
String DirectPrt	Can take the following values : • True : direct printing • False : none
String OneDoc	Can take the following values : • True : merging temporary document • False : none
String DltLgn	Can take the following values : • True : delete blank lines after merging • False : none
String MailTo	A field of a CSV file or a database table, containing email addresses
String MailSubj	Email subject
String HeaderSrc	Name of the file that contains the headers.
String TmpPdf	PDF temporary file.
String TmpPrn	PRN temporary file.
String DataSource	Complete path of the CSV file if you do not use DataSource class.
DataSource sourceData	An instance of the DataSource class.
String Output	Can take the following values : • *PDF : if the generated document is in PDF

	format • *EMAIL : if the result is sent by email • *OUTPRN : file Print PRN • *PRINT : the document is sent to print
String ShowDoc	Can take the following values : • True : to show the document • False : document not visible
String SavDoc	Complete path of the result document Example: C:\\temp\\result.docx
String SavFmt	Can take the following values : • *NORMAL : format of the template file • *DOC : Word format • *HTM : HTML format
String EndOpt	Can take the following values : • *NONE : no end processing option • *DOC : close Word document • *ALL : end of the command session and closing Word application • *APP : close Word application

All attributes are accessed and modified through Getters and Setters methods.

Attributes	Getters (read)	Setters (write)
String LNCSrvAddr	getLNCSrvAddr()	setLNCSrvAddr()
String Doc	getDoc()	setDoc()
String WordPath	getWordPath()	setWordPath()
String MrgType	getMrgType()	setMrgType()
String Unlink	getUnlink()	setUnlink()
String OutSave	getOutsave()	setOutSave()
String MrgSel	getMrgSel()	setMrgSel()
String OutPdf	getOutPdf()	setOutPdf()
String OutPrn	getOutPrn()	setOutPrn()
String ExecMrg	getExecMrg()	setExecMrg()
String Printer	getPrinter()	setPrinter()
String Copies	getCopies()	setCopies()
String Tray	getTray()	setTray()
String FirstPageTray	getFirstPageTray()	setFirstPageTray()
String OtherPageTray	getOtherPageTray()	setOtherPageTray()
String DirectPrt	getDirectPrt()	setDirectPrt()
String OneDoc	getOneDoc()	setOneDoc()
String DltLgn	getDltLgn()	setDltLgn()
String MailTo	getMailTo()	setMailTo()
String MailSubj	getMailSubj()	setMailSubj()
String HeaderSrc	getHeaderSrc()	setHeaderSrc()
String TmpPdf	getTmpPdf()	setTmpPdf()
String TmpPrn	getTmpPrn()	setTmpPrn()
String DataSource	getDataSource()	setDataSource()
String Output	getOutput()	setOutput()

`String ShowDoc`	getShowDoc()	setShowDoc()
`String SavDoc`	getSavDoc()	setSavDoc()
`String SavFmt`	getSavFmt()	setSavFmt()
`String EndOpt`	getEndOpt()	setEndOpt()
`DataSource sourceData`	getSourceData()	setSourceData()

The method **execute**() allows to execute the command LNCPRTDOC.

The class LNCTOXLS

Attributes	Description
String LNCSrvAddr	The IPv4 address of the Launcher server Example: 192.168.1.3
String ToXls	Template name Exemple: C:\\temp\\template.xlsx
String ToSheet	Excel sheet name
String ToName	Area name into Excel sheet
String XlsOpt	Can take the following values : • *REPLACE : new data overwrites the old data. • *RESIZE : new data overwrites the old data. Area in the workbook is resized. • *ADD : new data is added to the old data. • *INSERT : rows are inserted to receive new data. The parameter ENTIRER determines whether entire lines are inserted or not. • *MAP : column values from the data source are assigned to cells with the same names in the Excel spreadsheet. Only the first entry of the data source is used.
String NoData	Can take the following values : • *BLANK : if no data is present in the data source, the field is left empty • *DEL : if no data is present in the data source, the cells are removed
String XlsMap	Can take the following values : • *MAPNAME : column values from the data source are assigned to cells with the same names in the Excel spreadsheet. • *MAPVALUE : values for columns in the data source are assigned to cells whose initial value is the same
String ColPref	Can take the following values : • *NONE : no prefix for columns • Prefix value for columns
String DataSource	The full path of the CSV file if the DataSource class is not used.
String RecCnt	If using a CSV file directly as a source of data, it is mandatory to specify the number of rows in the CSV file. In all other cases, this attribute is not used.
String Entirer	Can take the following values : • True : entire lines are inserted • False : not inserting entire rows
String AutoFmt	Can take the following values : • *NONE : no automatic formatting is applied. • Format name : le specified format is applied.
String AutoFit	Can take the following values : • True : automatically adjust the column width. • False : no change of the column width

String FmtCells	Can take the following values : • True : the cell format is set according to the type Database. • False : the format of the cells is determined by Excel, according to the values.
String AddColH	Can take the following values : • True : column names are inserted in the first line. • False : column names are not inserted in the first line.
String Freeze	Can take the following values : • True : replaces the terms by their calculated value • False : none
String LocateSheet	Specifies the active sheet.
String LocateCell	Specifies the active cell.
DataSource sourceData	An instance of the DataSource class.
String Output	Can take the following values : • *PRINT : the document will be printed on the default printer.
String ShowDoc	Can take the following values : • True : to show the document • False : document not visible
String SavDoc	Complete path of the result document Example: C:\\temp\\result.xlsx
String SavFmt	Can take the following values : • *NORMAL : format of the template file • *XLS : Excel format • *TXT : text format
String EndOpt	Can take the following values : • *NONE : no end processing option • *DOC : close Excel document • *ALL : end of the command session and closing Excel application • *APP : close Excel application

All attributes are accessed and modified through Getters and Setters methods.

Attributes	Getters (read)	Setters (write)
String LNCSrvAddr	getLNCSrvAddr()	setLNCSrvAddr()
String ToXls	getToXls()	setToXls()
String ToSheet	getToSheet()	setToSheet()
String ToName	getToName()	setToName()
String XlsOpt	getXlsOpt()	setXlsOpt()
String NoData	getNoData()	setNoData()
String XlsMap	getXlsMap()	setXlsMap()
String ColPref	getColPref()	setColPref()
String DataSource	getDataSource()	setDataSource()

String RecCnt	getRecCnt()	setRecCnt()
String Entirer	getEntirer()	setEntirer()
String AutoFmt	getAutoFmt()	setAutoFmt()
String AutoFit	getAutoFit()	setAutoFit()
String FmtCells	getFmtCells()	setFmtCells()
String AddColH	getAddColH()	setAddColH()
String Freeze	getFreeze()	setFreeze()
String LocateSheet	getLocateSheet()	setLocateSheet()
String LocateCell	getLocateCell()	setLocateCell()
DataSource sourceData	getSourceData()	setSourceData()
String Output	getOutput()	setOutput()
String ShowDoc	getShowDoc()	setShowDoc()
String SavDoc	getSavDoc()	setSavDoc()
String SavFmt	getSavFmt()	setSavFmt()
String EndOpt	getEndOpt()	setEndOpt()

The method **execute**() allows to execute the command LNCTOXLS.

Use examples

These examples are dedicated to Java programs run on PC/server hosting Launcher Server.

The Launcher Server address (attribute : LNCSrvAddr) is **: 192.168.1.9**.

Java code should necessarily contain the following lines:

```
import fr.aura.launcher.wrapper.DataSource ;
import fr.aura.launcher.wrapper.LNCTOXLS ;
import fr.aura.launcher.wrapper.LNCPRTDOC ;
```

1) LNCTOXLS : CSV file as data source

In this example, the data source is a CSV file.

The command **LNCTOXLS** is used to generate Excel file.

```
String Destination=c:\\temp\\destination.xls";

DataSource myDataSource = new DataSource();
myDataSource.setType("csv");
myDataSource.setPath("c:\\temp\\source.csv");

LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setToSheet("*NONE");
myLNCTOXLS.setToName("*NONE");
myLNCTOXLS.setColPref("*NONE");
myLNCTOXLS.setAutoFmt("*NONE");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("true");
myLNCTOXLS.setSourceData(myDataSource);
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*XLS");
myLNCTOXLS.setSavDoc(Destination);
myLNCTOXLS.setRecCnt(55); //Mandatory if Type = csv

myLNCTOXLS.execute();
```

---

2) LNCTOXLS : Oracle

In this example, the database has the following characteristics :

- Type DB : **Oracle**
- Database server address : 192.168.1.7
- Port number of the database server : 1521
- Database user : hr
- Database user password : hr
- Database name : XE
- SQL query : select * from countries

The command **LNCTOXLS** is used to generate the following Excel file:
C:\temp\test2.xls

```
String Destination="C:\\temp\\test2.xls";

DataSource myDataSource = new DataSource();
myDataSource.setType("oracle");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setPort("1521");
myDataSource.setUser("hr");
myDataSource.setPassword("hr");
myDataSource.setDBName("XE");
myDataSource.setQuery("select * from countries");

LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setToSheet("*NONE");
myLNCTOXLS.setToName("*NONE");
myLNCTOXLS.setColPref("*NONE");
myLNCTOXLS.setAutoFmt("*NONE");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("true");
myLNCTOXLS.setSourceData(myDataSource);
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*XLS");
myLNCTOXLS.setSavDoc(Destination);

myLNCTOXLS.execute();
```

---

3) LNCTOXLS : MySQL

In this example, the database has the following characteristics :

- Type DB : **MySQL**
- Database server address : 192.168.1.7
- Port number of the database server : 3306
- Database user : root
- Database user password : (none)
- Database name : test
- SQL query : select * from test.marci

The command **LNCTOXLS** is used to generate the following Excel file:
C:\temp\test3.xls

```
String Destination="C:\\temp\test3.xls";

DataSource myDataSource = new DataSource();
myDataSource.setType("mysql");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setPort("3306");
myDataSource.setUser("root");
myDataSource.setPassword("");
myDataSource.setDBName("test");
myDataSource.setQuery("select * from test.marci");

LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setToSheet("*NONE");
myLNCTOXLS.setToName("*NONE");
myLNCTOXLS.setColPref("*NONE");
myLNCTOXLS.setAutoFmt("*NONE");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("true");
myLNCTOXLS.setSourceData(myDataSource);
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*XLS");
myLNCTOXLS.setSavDoc(Destination);

myLNCTOXLS.execute();
```

---

4) LNCTOXLS : PostgreSQL

In this example, the database has the following characteristics :

- Type DB : **PostgreSQL**
- Database server address : 192.168.1.7
- Port number of the database server : 6078
- Database user : postgres
- Database user password : aura
- Database name : newold
- SQL query : select * from marci where title < 'b'

The command **LNCTOXLS** is used to generate the following Excel file:
C:\temp\lnctoxls_test4_dbget.xls

```
String Destination="C:\\temp\\lnctoxls_test4_dbget.xls";

DataSource myDataSource = new DataSource();
myDataSource.setType("postgresql");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setPort("6078");
myDataSource.setUser("postgres");
myDataSource.setPassword("aura");
myDataSource.setDBName("newold");
myDataSource.setQuery("select * from marci where title < 'b'");

LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setToSheet("*NONE");
myLNCTOXLS.setToName("*NONE");
myLNCTOXLS.setColPref("*NONE");
myLNCTOXLS.setAutoFmt("*NONE");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("true");
myLNCTOXLS.setSourceData(myDataSource);
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*XLS");
myLNCTOXLS.setSavDoc(Destination);

myLNCTOXLS.execute();
```

---

5) LNCTOXLS : Sybase

In this example, the database has the following characteristics :

- Type DB : **Sybase**
- Database server address : 192.168.1.7
- Port number of the database server : 2638
- Database user : dba
- Database user password : sql
- Database name : newold
- SQL query : SELECT ID, GivenName, Surname FROM Customers

The command **LNCTOXLS** is used to generate the following Excel file:
C:\temp\test5.xls

```
String Destination="C:\\temp\\test2.xls";

DataSource myDataSource = new DataSource();
myDataSource.setType("sybase");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setPort("2638");
myDataSource.setUser("dba");
myDataSource.setPassword("sql");
myDataSource.setDBName("newold");
myDataSource.setQuery("SELECT ID, GivenName, Surname FROM
Customers ");

LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setToSheet("*NONE");
myLNCTOXLS.setToName("*NONE");
myLNCTOXLS.setColPref("*NONE");
myLNCTOXLS.setAutoFmt("*NONE");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("true");
myLNCTOXLS.setSourceData(myDataSource);
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*XLS");
myLNCTOXLS.setSavDoc(Destination);

myLNCTOXLS.execute();
```

---

6) LNCTOXLS : DB2 for iSeries

In this example, the database has the following characteristics :

- Type DB : **DB2 for iSeries**
- AS400 address : 192.168.1.7
- Port number of the database server : not necessary
- Database user : qpgmr
- Database user password : aura
- Library name : easycomxmp
- SQL query : select * from easycomxmp.s_customer

The command **LNCTOXLS** is used to generate the following Excel file:
C:\temp\test2.xls

```
String Destination="C:\\temp\\test2.xls";

DataSource myDataSource = new DataSource();
myDataSource.setType("as400db2");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setUser("qpgmr");
myDataSource.setPassword("aura");
myDataSource.setDBName("easycomxmp");
myDataSource.setQuery("select * from easycomxmp.s_customer");

LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setToSheet("*NONE");
myLNCTOXLS.setToName("*NONE");
myLNCTOXLS.setColPref("*NONE");
myLNCTOXLS.setAutoFmt("*NONE");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("true");
myLNCTOXLS.setSourceData(myDataSource);
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*XLS");
myLNCTOXLS.setSavDoc(Destination);

myLNCTOXLS.execute();
```

---

7) LNCTOXLS : Microsoft SQL Server

In this example, the database has the following characteristics :

- Type DB : **Microsoft SQL Server**
- Database server address : 192.168.1.7
- Port number of the database server : 1433
- Database user : sa
- Database user password : aura
- Database name : master
- SQL query : select * from dbo.spt_values

The command **LNCTOXLS** is used to generate the following Excel file:
C:\temp\test2.xls

```
String Destination="C:\\temp\\test2.xls";

DataSource myDataSource = new DataSource();
myDataSource.setType("sqlserver");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setPort("1433");
myDataSource.setUser("sa");
myDataSource.setPassword("aura");
myDataSource.setDBName("master");
myDataSource.setQuery("select * from dbo.spt_values");

LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setToSheet("*NONE");
myLNCTOXLS.setToName("*NONE");
myLNCTOXLS.setColPref("*NONE");
myLNCTOXLS.setAutoFmt("*NONE");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("true");
myLNCTOXLS.setSourceData(myDataSource);
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*XLS");
myLNCTOXLS.setSavDoc(Destination);

myLNCTOXLS.execute();
```

---

8) LNCTOXLS : SAP

In this example, the database has the following characteristics :

- Type DB: **SAP**
- SAP server address: 192.168.1.7
- SAP user: DDIC
- SAP user password: aura
- SAP table name: SPFLI
- SQL query: select **CITYFROM,CITYTO,DISTANCE,DEPTIME** from **SBOOK** where **CITYFROM = 'ROME'**

The command **LNCTOXLS** is used to generate the following Excel file:
C:\temp\sap_test.xlsx

```
String Destination="C:\\temp\\sap_test.xlsx";

DataSource myDataSource = new DataSource();
myDataSource.setType("saptable");
myDataSource.setUser("DDIC");
myDataSource.setPassword("aura");
myDataSource.setAshostSAP("192.168.1.7");
myDataSource.setClientSAP("001");
myDataSource.setLangSAP("");
myDataSource.setSysnrSAP("00");
myDataSource.setGwhostSAP("192.168.1.7");
myDataSource.setGwservSAP("3300");
myDataSource.setQuerySAP("CITYFROM = 'ROME'");
myDataSource.setRowcountSAP("5");
myDataSource.setRowskipsSAP("0");
myDataSource.setFoiSAP("CITYFROM,CITYTO,DISTANCE,DEPTIME");
myDataSource.setTableNameSAP("SPFLI");


LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("true");
myLNCTOXLS.setSourceData(myDataSource);
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*NORMAL");
myLNCTOXLS.setSavDoc(Destination);


myLNCTOXLS.execute();
```

9) LNCTOXLS : SAP – Using BAPI

In this example, the database has the following characteristics :

- Type DB: **SAP**
- SAP server address: 192.168.1.7
- SAP user: DDIC
- SAP user password: aura
- BAPI name: **BAPI_FLBOOKING_GETLIST**
- Result table used to export to Excel: BOOKING_LIST
- Parameter of the BAPI: MAX_ROWS=7

The command **LNCTOXLS** is used to generate the following Excel file:
C:\temp\sap_test.xlsx

```
String Destination="C:\\temp\\sap_test.xlsx";

String template="C:\\temp\\SBOOK_template_BAPI.xlsx";


 // Create a hash map
 Map<String,String> hm = new HashMap<String,String>();

// Put elements to the map
hm.put("MAX_ROWS", "70");

DataSource myDataSource = new DataSource();
myDataSource.setType("sapbapi");
myDataSource.setUser("DDIC");
myDataSource.setPassword("aura");
myDataSource.setAshostSAP("192.168.1.7");
myDataSource.setClientSAP("001");
myDataSource.setLangSAP("");
myDataSource.setSysnrSAP("00");
myDataSource.setGwhostSAP("192.168.1.7");
myDataSource.setGwservSAP("3300");
myDataSource.setBAPIName("BAPI_FLBOOKING_GETLIST");
myDataSource.setBAPIResultTable("BOOKING_LIST");
myDataSource.setParameterMap(hm);


LNCTOXLS myLNCTOXLS = new LNCTOXLS();
myLNCTOXLS.setToXls("*new");
myLNCTOXLS.setAutoFit("true");
myLNCTOXLS.setFmtCells("false");
myLNCTOXLS.setAddColH("false");
myLNCTOXLS.setShowDoc("false");
myLNCTOXLS.setSourceData(myDataSource);
```

```
myLNCTOXLS.setLNCSrvAddr("192.168.1.9");
myLNCTOXLS.setSavFmt("*NORMAL");
myLNCTOXLS.setSavDoc(Destination);


myLNCTOXLS.execute();
```

10) LNCPRTDOC : CSV file as data source

In this example, the data source is a CSV file (C:\temp\LNC002tmp.csv).

The command **LNCPRTDOC** is used to generate the following Word document.
C:\temp\TEST1_LNCPRTDOC_csv.DOC

The template used for mail merging is :
C:\temp\SP_CUST.DOC

```
String source = "C:\\temp\\LNC002tmp.csv";
String template = "C:\\temp\\SP_CUST.DOC";
String destination = "C:\\temp\\TEST1_LNCPRTDOC_csv.DOC";

DataSource myDataSource = new DataSource();
myDataSource.setType("csv");
myDataSource.setPath(source);

LNCPRTDOC myLNCPRTDOC = new LNCPRTDOC();
myLNCPRTDOC.setDoc(template);
myLNCPRTDOC.setSavFmt("*NORMAL");
myLNCPRTDOC.setShowDoc("true");
myLNCPRTDOC.setMrgType("*FILE");
myLNCPRTDOC.setUnlink("true");
myLNCPRTDOC.setOutSave("true");
myLNCPRTDOC.setSavDoc(destination);
myLNCPRTDOC.setMrgSel("*LTR");
myLNCPRTDOC.setExecMrg("true");
myLNCPRTDOC.setEndOpt("*CON");
myLNCPRTDOC.setHeaderSrc("*FILE");
myLNCPRTDOC.setSourceData(myDataSource);
myLNCPRTDOC.setLNCSrvAddr("192.168.1.9");

myLNCPRTDOC.execute();
```

---

11) LNCPRTDOC : Oracle

In this example, the database has the following characteristics :

- Type DB : **Oracle**
- Database server address : 192.168.1.7
- Port number of the database server : 1521
- Database user : hr
- Database user password : hr
- Database name : XE
- SQL query : select * from employees

The command **LNCPRTDOC** is used to generate the following Word document:
C:\temp\TEST2_oracle.doc

The template used for mail merging is :
C:\temp\SP_CUST_ORACLE.DOC

```
String template = "C:\\temp\\SP_CUST_ORACLE.DOC";
String destination = "C:\\temp\\TEST2_oracle.DOC";

DataSource myDataSource = new DataSource();
myDataSource.setType("oracle");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setPort("1521");
myDataSource.setUser("hr");
myDataSource.setPassword("hr");
myDataSource.setDBName("XE");
myDataSource.setQuery("select * from employees");

LNCPRTDOC myLNCPRTDOC = new LNCPRTDOC();
myLNCPRTDOC.setDoc(template);
myLNCPRTDOC.setSavFmt("*NORMAL");
myLNCPRTDOC.setShowDoc("true");
myLNCPRTDOC.setMrgType("*FILE");
myLNCPRTDOC.setOutSave("true");
myLNCPRTDOC.setSavDoc(destination);
myLNCPRTDOC.setMrgSel("*LTR");
myLNCPRTDOC.setExecMrg("true");
myLNCPRTDOC.setEndOpt("*NONE");
myLNCPRTDOC.setHeaderSrc("*FILE");
myLNCPRTDOC.setSourceData(myDataSource);
myLNCPRTDOC.setSourceData(myDataSource);
myLNCPRTDOC.setLNCSrvAddr("192.168.1.9");

myLNCPRTDOC.execute();
```

---

12) LNCPRTDOC : MySQL

In this example, the database has the following characteristics :

- Type DB : **MySQL**
- Database server address : 192.168.1.7
- Port number of the database server : 3306
- Database user : root
- Database user password : (none)
- Database name : test
- SQL query : select * from test.marci

The command **LNCPRTDOC** is used to generate the following Word document:
C:\temp\TEST2_MySQL.doc

The template used for mail merging is :
C:\temp\SP_CUST_MYSQL.DOC

```
String template = "C:\\temp\\SP_CUST_MYSQL.DOC";
String destination = "C:\\temp\\TEST2_MySQL.DOC";

DataSource myDataSource = new DataSource();
myDataSource.setType("mysql");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setPort("3306");
myDataSource.setUser("root");
myDataSource.setPassword("");
myDataSource.setDBName("test");
myDataSource.setQuery("select * from test.marci");

LNCPRTDOC myLNCPRTDOC = new LNCPRTDOC();
myLNCPRTDOC.setDoc(template);
myLNCPRTDOC.setSavFmt("*NORMAL");
myLNCPRTDOC.setShowDoc("true");
myLNCPRTDOC.setMrgType("*FILE");
myLNCPRTDOC.setOutSave("true");
myLNCPRTDOC.setSavDoc(destination);
myLNCPRTDOC.setMrgSel("*LTR");
myLNCPRTDOC.setExecMrg("true");
myLNCPRTDOC.setHeaderSrc("*FILE");
myLNCPRTDOC.setSourceData(myDataSource);
myLNCPRTDOC.setDltLgn("true");
myLNCPRTDOC.setSourceData(myDataSource);
myLNCPRTDOC.setLNCSrvAddr("192.168.1.9");

myLNCPRTDOC.execute();
```

---

13) LNCPRTDOC : PostgreSQL

In this example, the database has the following characteristics :

- Type DB : **PostgreSQL**
- Database server address : 192.168.1.7
- Port number of the database server : 6078
- Database user : postgres
- Database user password : aura
- Database name : newold
- SQL query : select * from marci where title < 'b'

The command **LNCPRTDOC** is used to generate the following Word document:
C:\temp\TEST4_POSTGRESQL.doc

The template used for mail merging is :
C:\temp\SP_CUST_POSTGRESQL.DOC

```
String template = "C:\\temp\\SP_CUST_POSTGRESQL.DOC";
String destination = "C:\\temp\\TEST4_POSTGRESQL.DOC";

DataSource myDataSource = new DataSource();
myDataSource.setType("postgresql");
myDataSource.setSrvAddr("192.168.1.7");
myDataSource.setPort("6078");
myDataSource.setUser("postgres");
myDataSource.setPassword("aura");
myDataSource.setDBName("newold");
myDataSource.setQuery("select * from marci where title < 'b'");

LNCPRTDOC myLNCPRTDOC = new LNCPRTDOC();
myLNCPRTDOC.setDoc(template);
myLNCPRTDOC.setSavFmt("*NORMAL");
myLNCPRTDOC.setShowDoc("true");
myLNCPRTDOC.setMrgType("*FILE");
myLNCPRTDOC.setUnlink("true");
myLNCPRTDOC.setOutSave("true");
myLNCPRTDOC.setSavDoc(destination);
myLNCPRTDOC.setMrgSel("*LTR");
myLNCPRTDOC.setExecMrg("true");
myLNCPRTDOC.setEndOpt("*NONE");
myLNCPRTDOC.setHeaderSrc("*FILE");
myLNCPRTDOC.setSourceData(myDataSource);
myLNCPRTDOC.setDltLgn("true");
myLNCPRTDOC.setSourceData(myDataSource);
myLNCPRTDOC.setLNCSrvAddr("192.168.1.9");

myLNCPRTDOC.execute();
```

14) LNCPRTDOC : SAP- Using BAPI

In this example, the database has the following characteristics :

- Type DB: **SAP**
- SAP server address: 192.168.1.7
- SAP user: DDIC
- SAP user password: aura
- BAPI name: **BAPI_FLBOOKING_GETLIST**
- Result table used to export to Excel: BOOKING_LIST
- Parameter of the BAPI: MAX_ROWS=7

The command **LNCPRTDOC** is used to generate the following Word file:
C:\temp\result.docx

```
String template = "C:\\temp\\template.docx";
String destination = "C:\\temp\\result.docx";

 // Create a hash map
 Map<String,String> hm = new HashMap<String,String>();

// Put elements to the map
hm.put("MAX_ROWS", "70");

DataSource myDataSource = new DataSource();
myDataSource.setType("sapbapi");
myDataSource.setUser("DDIC");
myDataSource.setPassword("aura");
myDataSource.setAshostSAP("192.168.1.7");
myDataSource.setClientSAP("001");
myDataSource.setLangSAP("");
myDataSource.setSysnrSAP("00");
myDataSource.setGwhostSAP("192.168.1.7");
myDataSource.setGwservSAP("3300");
myDataSource.setBAPIName("BAPI_FLBOOKING_GETLIST");
myDataSource.setBAPIResultTable("BOOKING_LIST");
myDataSource.setParameterMap(hm);

LNCPRTDOC myLNCPRTDOC = new LNCPRTDOC();
myLNCPRTDOC.setDoc(template);
myLNCPRTDOC.setSavFmt("*NORMAL");
myLNCPRTDOC.setShowDoc("false");
myLNCPRTDOC.setMrgType("*FILE");
myLNCPRTDOC.setOutSave("true");
myLNCPRTDOC.setSavDoc(destination);
myLNCPRTDOC.setMrgSel("*LTR");
myLNCPRTDOC.setExecMrg("true");
myLNCPRTDOC.setEndOpt("*NONE");
myLNCPRTDOC.setHeaderSrc("*FILE");
```

```
myLNCPRTDOC.setSourceData(myDataSource);
myLNCPRTDOC.setLNCSrvAddr("192.168.1.9");

myLNCPRTDOC.execute();
```

Webservices

Invoking service methods

Once the Gateway web service is started, the service methods can be accessed at **https://<IP address>:9090/launcher/method_name** (or the port specified by the user in the beans.xml file or using the Service Manager).

The primary service methods are:

- open
- ldapauth
- cmd
- close
- filetransfer
- lncprtdoc
- lnctoxls

All these services can be invoked from any client that is connected to the same network as the one the Launcher REST Gateway is hosted on. Most of these service methods can be invoked using either **POST or GET** HTTPS request methods. If parameters must be passed to the service methods, they are included in the POST **message body in JSON format**. Certain parameters such as the **Launcher connexion ID can be included in the service URL**.

Open

The **open** service method **establishes a connection with the Launcher server**.

If « UseLdap=**Yes » into the launcher.ini file,** this method is also used to **authenticate the REST Gateway web service client**. He must include his username and password to authenticate (LDAP – Active Directory) his connection to the web service.

Here is an example of calling the **open** service method that connects to the Launcher server on localhost at port 6078 (which are also default values):

```
POST /launcher/open HTTP/1.1
Host: localhost:9090
Content-Type: application/JSON

{ "open": { "server":"localhost", "port":"6078", "username":"aura", "password":"aura" } }
```

The above request is sent to the URL **https://<IP address>:9090/launcher/open** using the POST HTTP method.

If the authentication is successful, and the Launcher REST Gateway manages to connect to the Launcher Server, an integer representing **the client id will be returned to the client**. The client must **include this id in all subsequent requests** to the Launcher REST Gateway.

Ldapauth

The **ldapauth** web method is used to **authenticate the client to the Launcher/i Server, that is configured to use LDAP authentication**. This method is reachable at the following URL: **https://localhost:9090/launcher/ldapauth**. The POST request made to this service method should contain a JSON message in the body with the Active Directory username and password. Here is an example:

```
POST /launcher/ldapauth HTTP/1.1

Host: localhost:9090
Content-Type: application/JSON
Cache-Control: no-cache

{ "ldapauth": { "id":"1", "username":"john", "password":"secretpassword" }
}
```

Cmd

The **cmd** web method is used to **execute Launcher CMD verbs**. The **cmd** method can be called using HTTP POST requests because the parameters of the command, **parm1** and **parm2**, are sent in the POST message body.

To call the **cmd** method, it is necessary to include the client id, the verb, parm1 and parm2 in the JSON message in the POST body as shown in the following example:

```
POST /launcher/cmd HTTP/1.1
Host: localhost:9090
Content-Type: application/json
Cache-Control: no-cache

{"cmd":{"verb":"WORDOPEN","parm1":"new","parm2":"visible","id":"1"}}
```

When the command is executed, **the value returned by the Launcher Server will be forwarded by the Launcher REST Gateway to the client** which made the request:

- *ACK or *MSG if the command is successful
- *REJ if the command fails. Further details about the error are available into the Launcher Server trace.

Filetransfer

The **filetransfer** web method is used to **transfer files from the client machine to the Launcher server**. When calling this web method, the data that needs to be sent is not written in a JSON message, instead it is included as **form-data**. The data that must be included when calling the filetransfer method are the **client id**, the **destination file name**, the **destination directory**, and **the file itself** that is to be uploaded. Here is an example:

```
POST /launcher/filetransfer HTTP/1.1
Host: localhost:9090
Cache-Control: no-cache
----WebKitFormBoundaryE19zNvXGzXaLvS5C
Content-Disposition: form-data; name="id"
1
```

```
----WebKitFormBoundaryE19zNvXGzXaLvS5C
Content-Disposition: form-data; name="destFileName"
Example_doc.pdf

----WebKitFormBoundaryE19zNvXGzXaLvS5C
Content-Disposition: form-data; name="destDir"
C:\Temp


----WebKitFormBoundaryE19zNvXGzXaLvS5C
Content-Disposition: form-data; name="file"
C:\Temp\test.pdf
```

The above example will transfer the attached file to the server, where it will be saved in C:\Temp with the file name Example_doc.pdf. Note that the complete file path including the directory and filename can be passed in the “destFileName” field of the form data, in which case the “destDir” field is not required.

Attaching the file as form data can be achieved in different ways depending on the language that the client is written in. See the examples into the installation directory

Close

The **close** web method is used to close the connection to the Launcher server. The only argument that needs to be passed when calling this method is the **client id**. This can be done using a HTTP GET request with the client id in the URL as follows:

```
GET /launcher/close/1 HTTP/1.1
Host: localhost:9090
Content-Type: application/json
Cache-Control: no-cache
```

The above example will close the connection of the client with the id number 1.

Alternatively, the **close** method can be requested by POST, with the message body containing the client id:

```
POST /launcher/close HTTP/1.1
Host: localhost:9090
Content-Type: application/json
Cache-Control: no-cache

{ "close": { "id":"1" } }
```

Lncprtdoc

The `lncprtdoc` web method is used to **execute Word mailmerge from a model and using data source (CSV file)**. The `lncprtdoc` method is called using HTTP POST requests because the parameters of the command are sent in the POST message body.

The following URL is used: `/launcher/lncprtdoc`

This is the different parameters :

```
<command name="LNCPRTDOC">
  <param1>
    <param name="Doc" default="*New"/>
    <param name="WordPath" default="notrequired"/>
    <param name="MrgType" default="*FILE"/>
    <param name="UnLink" default="true"/>
    <param name="DataSource" default="" type="string"/>
    <param name="OutSave" default="false"/>
    <param name="MrgSel" default="*LTR"/>
    <param name="SavFmt" default="*NORMAL"/>
    <param name="ExecMrg" default="true"/>
    <param name="DltLgn" default="true"/>
    <param name="ShowDoc" default="false"/>
    <param name="SavDoc" default="" type="string"/>
  </param1>
</command>
```

Parameters	Description
String Doc	Template name. Example: C:\\temp\\template.docx
String WordPath	Default directory of Word.
String MrgType	Can take the following values : • *NONE : no merge • *FILE : data source to merge comes from CSV file or database.
String Unlink	Can take the following values : • True : deleting blank fields • False : none
String OutSave	Can take the following values : • True : save the generated document • False : none
String MrgSel	Merge format : • *LTR : letter format • *CAT : catalog format
String ExecMrg	Can take the following values : • True : exécution of the mail merge • False : the document is only prepared for the mail merge
String DltLgn	Can take the following values : • True : delete blank lines after merging • False : none
String DataSource	Complete path of the CSV file.
String ShowDoc	Can take the following values : • True : to show the document • False : document not visible
String SavDoc	Complete path of the result document Example: C:\\temp\\result.docx

String SavFmt	Can take the following values : • *NORMAL : format of the template file • *DOC : Word format • *HTM : HTML format

Lnctoxls

The **lnctoxls** web method is used to **export data (CSV file) to an Excel model**. The **lnctoxls** method is called using HTTP POST requests because the parameters of the command are sent in the POST message body.

The following URL is used: **/launcher/lnctoxls**

This is the different parameters :

```
<command name="LNCTOXLS">
  <param1>
    <param name="ToXls" default="*New"/>
    <param name="ToSheet" default=""/>
    <param name="ToName" default="*NONE"/>
    <param name="ColPref" default="*NONE"/>
    <param name="DataSource" default="" type="string"/>
    <param name="RecCnt" default=""/>
    <param name="AutoFmt" default="*NONE"/>
    <param name="AutoFit" default=""/>
    <param name="FmtCells" default=""/>
    <param name="AddColH" default="*NONE"/>
    <param name="ShowDoc" default="false"/>
    <param name="SavDoc" default="" type="string"/>
    <param name="SavFmt" default="*XLS"/>
    <param name="XlsOpt" default=""/>
    <param name="XlsMap" default=""/>
  </param1>
</command>
```

Parameters	Description
String ToXls	Template name Exemple: C:\\temp\\template.xlsx
String ToSheet	Excel sheet name
String ToName	Area name into Excel sheet
String XlsOpt	Can take the following values : • *REPLACE : new data overwrites the old data. • *RESIZE : new data overwrites the old data. Area in the workbook is resized. • *ADD : new data is added to the old data. • *INSERT : rows are inserted to receive new

	data. The parameter ENTIRER determines whether entire lines are inserted or not. • *MAP : column values from the data source are assigned to cells with the same names in the Excel spreadsheet. Only the first entry of the data source is used.
String XlsMap	Can take the following values : • *MAPNAME : column values from the data source are assigned to cells with the same names in the Excel spreadsheet. • *MAPVALUE : values for columns in the data source are assigned to cells whose initial value is the same
String ColPref	Can take the following values : • *NONE : no prefix for columns • Prefix value for columns
String DataSource	The full path of the CSV file.
String RecCnt	If using a CSV file directly as a source of data, it is mandatory to specify the number of rows in the CSV file. In all other cases, this attribute is not used.
String AutoFmt	Can take the following values : • *NONE : no automatic formatting is applied. • Format name : le specified format is applied.
String AutoFit	Can take the following values : • True : automatically adjust the column width. • False : no change of the column width
String FmtCells	Can take the following values : • True : the cell format is set according to the type Database. • False : the format of the cells is determined by Excel, according to the values.
String AddColH	Can take the following values : • True : column names are inserted in the first line. • False : column names are not inserted in the first line.
String ShowDoc	Can take the following values : • True : to show the document • False : document not visible
String SavDoc	Complete path of the result document Example: C:\\temp\\result.xlsx
String SavFmt	Can take the following values : • *NORMAL : format of the template file • *XLS : Excel format • *TXT : text format

Authentication

Clients that want to consume the web service methods provided by the Launcher REST Gateway first have to **authenticate themselves using the open web method**. This authentication uses Active Directory.

Once the user is authenticated to the Launcher REST Gateway after calling the open method, they will receive a **client id**, allowing them to call the other web service methods.

The Launcher can also requires LDAP authentication (if configuration is done), so the client will receive a message saying "LDAP authentication required" when calling the other service methods.

In order to authenticate with the Launcher Server, the client must call the ldapauth web method after calling the open method.

More details, including how to configure the authentication method of the REST Gateway, are in the Configuration section.

Consuming webservices from a PHP client

A few examples of using web service are given in the **« Examples\PHP** »folder.

The code used for preparing the messages to be sent to the web service can be encapsulated in a PHP class (**Examples\PHP\Classe\LauncherClient.PHP**), simplifying the service method calls (refer to testClient.PHP, testClient_LNCDBGET_LNCPRTDOC.php et testClient_LNCDBGET_LNCTOXLS.php).

Other examples

Windev21

A full example and a user guide are provided in to the "**C:\Program Files (x86)\LAUNCHER XPRESS\Exemples\Windev**" folder.

Visual Studio 2015 – C#

A full example and a user guide are provided in to the "**C:\Program Files (x86)\LAUNCHER XPRESS\Exemples\VisualStudio_CSharp**" folder.

Advanced programming

Special values

When using LAUNCHER Office commands or API programs, the parameters accept special values LAUNCHER Office processed.

These special values allows to define Windows or LAUNCHER Office environment properties.

- **%LNCDIR%**

  This special value indicates the directory from which LAUNCHER Office server was launched on the PC.
  Default directory is « C:\Program Files\LAUNCHER<X> ».
  Example :
  CHGVAR VAR(&PARM1) VALUE('%LNCDIR%\Samples\Model.dot')

- **%CHR(n)**

  Allows to define a character with its ASCII value.
  n is a decimal number from 0 to 255.
  Example :
  CHGVAR VAR(&PARM1) VALUE('Ligne 1%CHR(10)Ligne2')
  %CHR(10) inserts a character « section jump ».

- **%PARA% or %LF%**

  Allows to insert a *skip a line* in a Word text or electronic message.
  %LF%, allow to insert a return to the line in the same Excel cell.

  NB : %LF%, use for LAUNCHER Office version > to v 2.0.41
  Example :
  CHGVAR VAR(&PARM1) VALUE('Paragraphe 1%PARA%Suite')

- **%EURO%**

  Inserts EURO (€) symbol in text.
  Example :
  CHGVAR VAR(&PARM1) VALUE(&VAL *CAT ' %EURO%')

- **%TAB%**

  Inserts **Tab** character in text.

- **%CONID%**

  Inserts current connection LAUNCHER ID number.
  This is a sole number from 1 to simultaneous accepted connections maximum number.
  This ID is useful in order to generate temporary and sole file names.

- **%NONE%**

  Initialises PARM2 as %NONE% if one want to clear a bookmark content

- **%TEMP%**

If the word within % characters is Windows environment variable, it will be replaced with its value.

Example:
CHGVAR VAR(&PARM1) VALUE('%TEMP%\file.txt')
%TEMP% indicate the temporary Windows directory, as define in its environment.

- **%SEP%**

  To pass to the following cell.

  If one were positioned on the last column of the table, one passes at beginning of the following line.

  Example:
  CHGVAR VAR(&PARM1) VALUE('%INS%Value column 1%SEP%Value Column 2 & 3%MRG%%SEP%12300%SEP%')

- **%INS%**

  To insert a new line and to positon at the beginning of the new line.

  Example:
  CHGVAR VAR(&PARM1) VALUE('%INS%Value column 1%SEP%Value Column 2 & 3%MRG%%SEP%12300%SEP%')

- **%MRG%**

  To merge the cuurent cell with the following one.

  Exemple:
  CHGVAR VAR(&PARM1) VALUE('%INS%Value column 1%SEP%Value Column 2 & 3%MRG%%SEP%12300%SEP%')

- **%CRLF%**

  Allows to insert a return to the line in the same Word cell.

Word

LAUNCHER Office and Word

LAUNCHER Office supports all Word versions from Office 2010 up to Office 2019.

Using the command **LNCPRTDOC** is the easiest way to insert external database data in a Word document.

If this command do not allow to meet your requested detail level, use **LNCOPEN**, **LNCCMD** and **LNCCLOSE** commands.

To use Word from an AS/400 program, the following procedure must be applied :

```
/* State and initialise the program variables */
 DCL VAR(&HANDLE) TYPE(*CHAR) LEN(50) VALUE('*ONLY')
 DCL VAR(&SVRADD) TYPE(*CHAR) LEN(30) VALUE('*DEV')
 DCL VAR(&CCSID) TYPE(*CHAR) LEN(10) VALUE('*JOB')
 DCL VAR(&OPT) TYPE(*CHAR) LEN(1)
 DCL VAR(&CMD) TYPE(*CHAR) LEN(10)
 DCL VAR(&PARM1) TYPE(*CHAR) LEN(512)
 DCL VAR(&PARM2) TYPE(*CHAR) LEN(1024)
 DCL VAR(&RESULT) TYPE(*CHAR) LEN(512)

/* Open communication between AS/400 PGM and PC */
 CALL PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)

/* Open Word, and a document template is necessary */
 CHGVAR VAR(&CMD) VALUE('WORDOPEN')
 CHGVAR VAR(&PARM1) VALUE(' ')
 CHGVAR VAR(&PARM2) VALUE(' ')
 CALL PGM(LNCCMD) PARM(&HANDLE &OPT &CMD &PARM1 &PARM2 &RESULT)

/* Proceed with the document lay out */
/* . . . */

/* Save, Print, Display the document (Optional) */
/* . . . */

/* Close document and Word (Optional) */
 CHGVAR VAR(&CMD) VALUE('WORDCLOSE')
 CHGVAR VAR(&PARM1) VALUE(' ')
 CHGVAR VAR(&PARM2) VALUE(' ')
 CALL PGM(LNCCMD) PARM(&HANDLE &OPT &CMD &PARM1 &PARM2 &RESULT)

/* Fermer la communication */
 CALL PGM(LNCCLOSE) PARM(&HANDLE)
```

To lay out the document, "**bookmarks**" or "**Mailing**" procedure can be used.

Document lay out with bookmarks

Bookmarks procedure allows Word document lay out or modification, addressing to document positions to insert text, object or picture.

With Word, Bookmarks must be inserted first in the template using "Inserting – Bookmarks" menu.

A bookmark is named. It refers to a position or a selection within the text.

LNCCMD program WBOOKMARK key word allows setting Word pointer at a bookmark position and inserts text.
If the bookmark stands for a selection, the new text will replace it all.

WTYPETEXT key word allows to insert text to the current pointer position.

While inserting text with WBOOKMARK or WTYPETEXT, Word will readjust document lay out, taking care of borders, lines and page jumps as well as alignments.

A bookmark can also be used to set the pointer in view of inserting another document (WINSERTF) or image (WINSERTIMG).

Special values for bookmark names allows scrolling through charts, to include cells. See WBOOKMARK detail.

Word offers to the programmer a set of pre-defined bookmarks in order to reach some identified document parts. See Bookmarks preset.

Bookmark can be found anywhere within a document including headers, footers.

Before launching the mailing part of the final document, a mailing template can be prepared with bookmarks procedure.

It allows template enhancement adding, as example, a logo.

There is no more bookmark in the final document resulting of a mailing !

Pre-defined Bookmarks

Microsoft Word have some pre-defined bookmarks.

They can be used with all LAUNCHER Office commands where a bookmark name is useful.

Bookmarks	Descriptions
\Sel	Current selection or inserting point.
\PrevSel1	Latest selection where editing take place.
\PrevSel2	Second latest selection where editing take place.
\StartOfSel	Beginning of current selection.
\EndOfSel	End of current selection.
\Line	Current line or current selection first line. If inserting point is set at line end but not the last section line, the bookmark includes the next line completely.

\Char	Current character. It can be the character next to the inserting point if there is no selection, or the first selection character.
\Para	Current paragraph. That is the paragraph including the inserting point, or, if several paragraphs are selected, the selection first paragraph. Note that if the inserting point, or the selection is in the document last paragraph the " \Para " bookmark does not include the paragraph mark.
\Section	Current section, including end of section break. Current section includes the inserting point or the selection. If the selection includes more than one section, the " \Section " bookmark is the selection first section.
\Doc	Active document entire content, excepted paragraph final mark.
\Page	Current page, including end of page end break. Current page includes the inserting point. If the selection includes more than one page, the " \Page " bookmark is the selection first page. Note that if the inserting point, or the selection is in the document last page the " \Page " bookmark does not include the page final mark.
\StartOfDoc	Document beginning.
\EndOfDoc	Document ending.
\Cell	Table current cell. This cell contains the inserting point. If one or more table cells are included in the current selection, the "\Cell " bookmark is the selection first cell.
\Table	Current table. This table contains the inserting point or the selection. If the selection includes more than one table, the "\Table " bookmark is the selection first table, even if the entire table has not been chosen.
\HeadingLevel	Header containing the inserting point or selection, and some more subsidiary headers and texts. If the current selection has a text body, the " \HeadingLevel " bookmark includes the previous header and any others subsidiary headers and texts.

Excel

LAUNCHER Office and Excel

LAUNCHER Office support all Excel versions from Office 2010 up to Office 2019.

LAUNCHER Office must be running on the PC in order to use Excel, further "Start Excel invisible" option should have been selected in configuration tab if Excel has to stay hidden.

Do not forget to close Excel when ending the session (EXCELCLOSE)

Do not forget to close Excel at the end of the session (EXCELCLOSE).

Mails

Sending e-mails

LAUNCHER Office allows you to send e-mail messages and control the main features:

recipients (TO, CC, BCC),

object, message text, attachment(s),

priority level, acknowledgement.

Messaging systems must be **MAPI** compatibles (this is the case for Outlook or Exchange). Lotus notes or a SMTP server can also be used.

Notice : Sending messages do not require messaging system running on the station.

E-mails support functions

Some functions are compatible with all interfaces, others are dedicated to particular interface.

- : MAPI and Lotus Notes interfaces compatible Functions.
- : SMTP interface compatible Functions.
- : All interfaces compatible Functions.

- MAILPREP, to prepare a message.
- MAILATT, to add a file as message attachment.
- MAILTEXT, to define a text body.
- MAILTO, to add recipients to a message.
- MAILPRTY, to set message priority.
- MAILSUBJ, to add an object to a message.
- MAILREPORT, to activate a message follow-up.
- MAILSEND, to send a message.
- MAILEND, to release all message allocated structures.
- MAILSMTP, to set connections with a SMTP server.
- MAILBODYF, to allow a message to be sent from an HTML page.
- MAILCC, to define message secondary recipients.

The MAPI technology

To send messages, **LAUNCHER Office** uses a technology called MAPI (Messaging Application Program Interface) developed by Microsoft collaboration with a number of other messaging system producers.

The aim of this technology is to furnish developers with a "gateway" for their applications. These applications can then access any messaging system, and not just one application designed for one system.

By installing specific plugs-ins, you can send faxes through your e-mails.

For more information, we advise you to consult the Microsoft web-site (**msdn.microsoft.com)**, and take a look at the articles concerning **Microsoft Fax.**

When using Exchange, different products are used to send faxes through e-mails.

To use Outlook or Exchange services, you have to configure LAUNCHER Office to use them.

So, in the configuration tab, you will find a combo box that allows you to choose you messaging system type : choose **MAPI**.

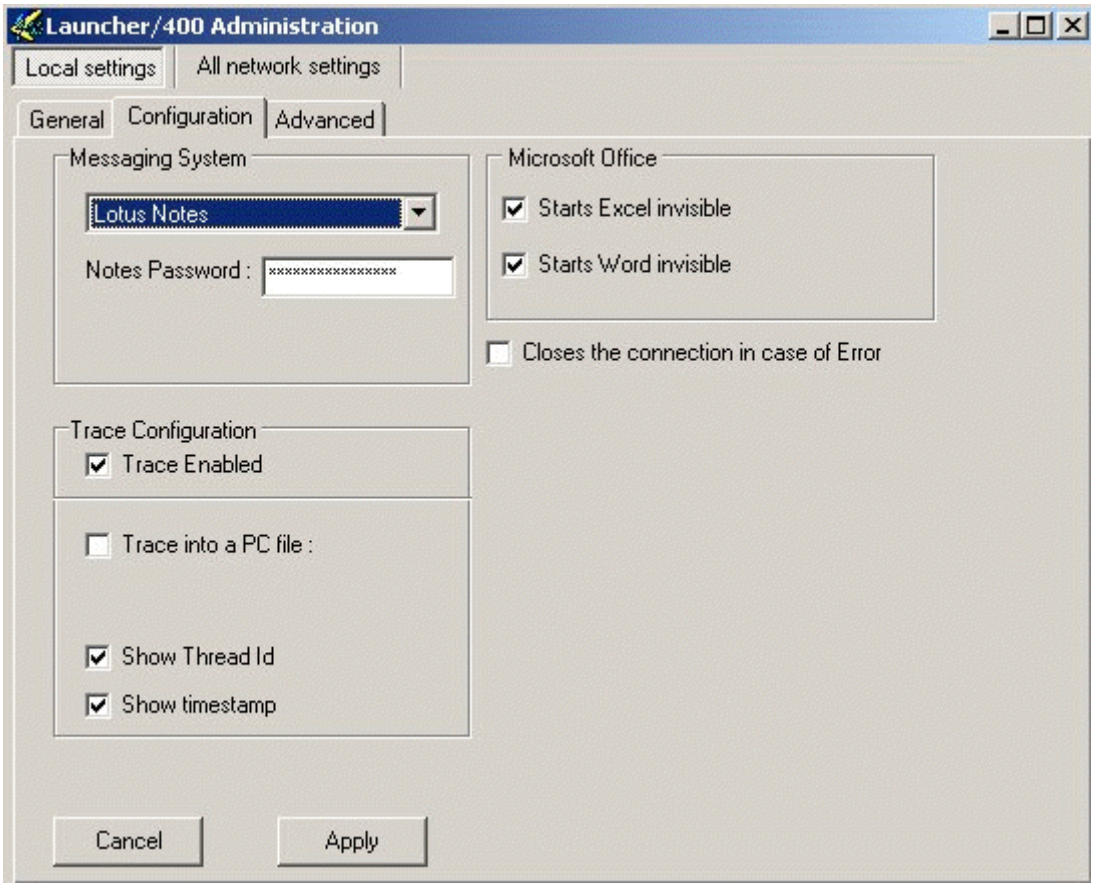
You can also choose the profile you want to use to send your e-mails.

Using Lotus notes

Up to version 5 Lotus Notes is not MAPI compatible.

That is why a dedicated Notes versions 4.5 and 4.6 interface was developed. This interface allows use of the same **LAUNCHER Office** functions as available with MAPI.

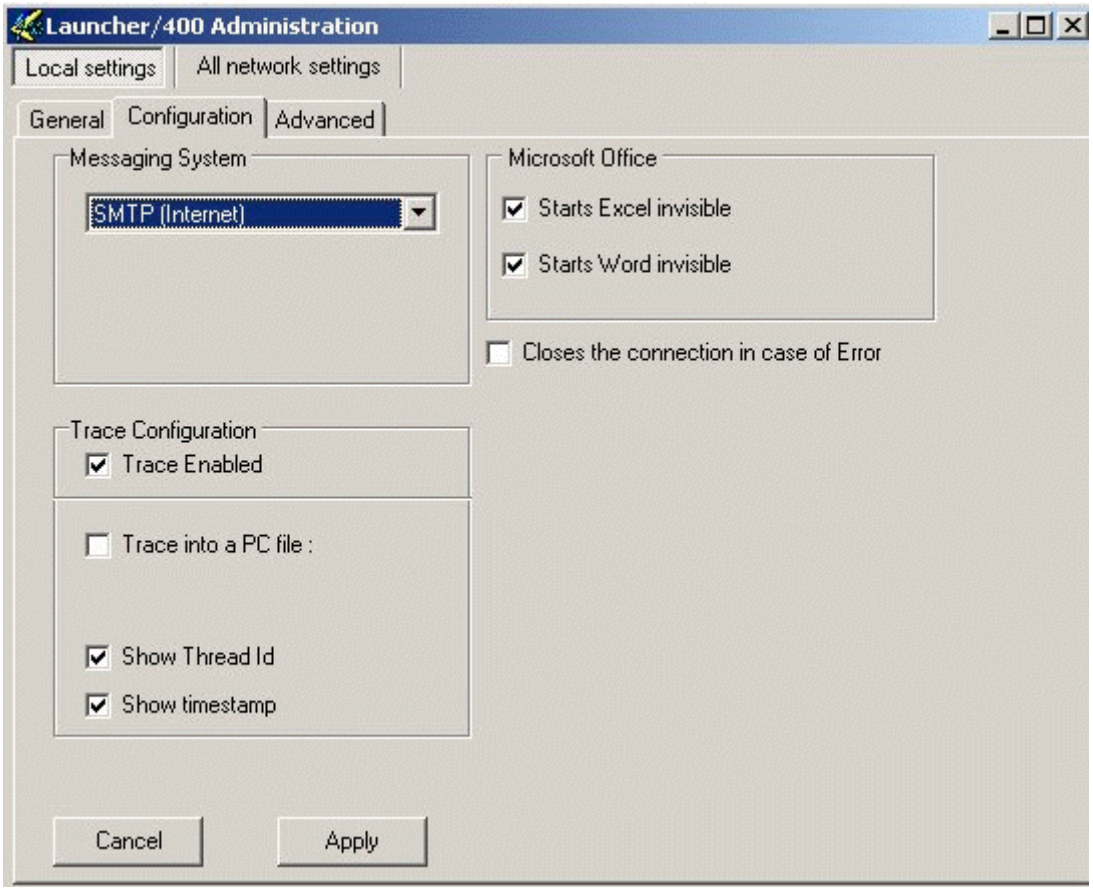
To use Notes with **LAUNCHER Office**, select Lotus Notes as messaging system type, and enter your Notes user password. Station profile is necessary.



Using a SMTP server

If there is no MAPI compatible messaging system, but a SMTP server, or if the use of SMTP sending functions are chosen, for those cases, a dedicated SMTP interface has been developed.

To use SMTP with **LAUNCHER Office**, select SMTP as messaging system type. No profile are required, all is programming.



S/MIME

Since the 2.7.1.1 version, S/MIME functionalities have been introduced for sending SMTP mails with Launcher.

We used S/MIME Version 4.0, RFC 8551.

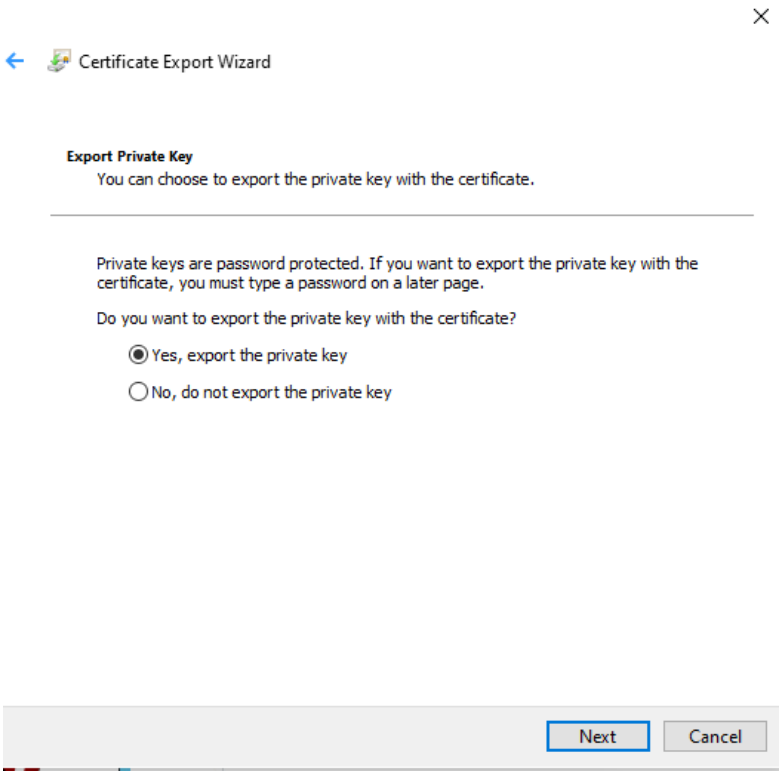
S/MIME (Secure/Multipurpose Internet Mail Extensions) is used to send and secure MIME data.

Based on the popular Internet MIME standard, S/MIME provides the following cryptographic security services:

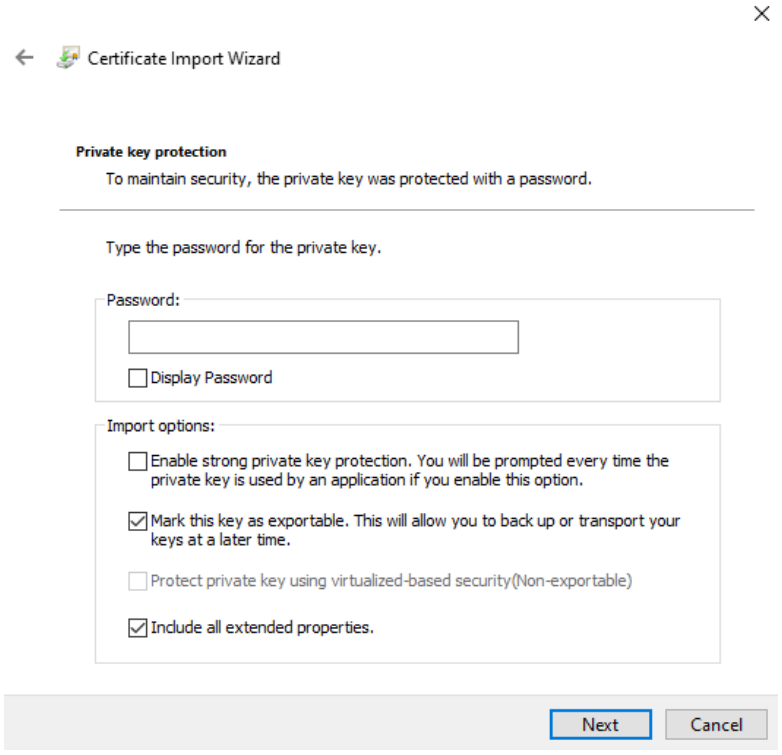
- authentication
- message integrity and non-repudiation of origin (using digital signatures)
- privacy and data security (using encryption).

Certificate on client side

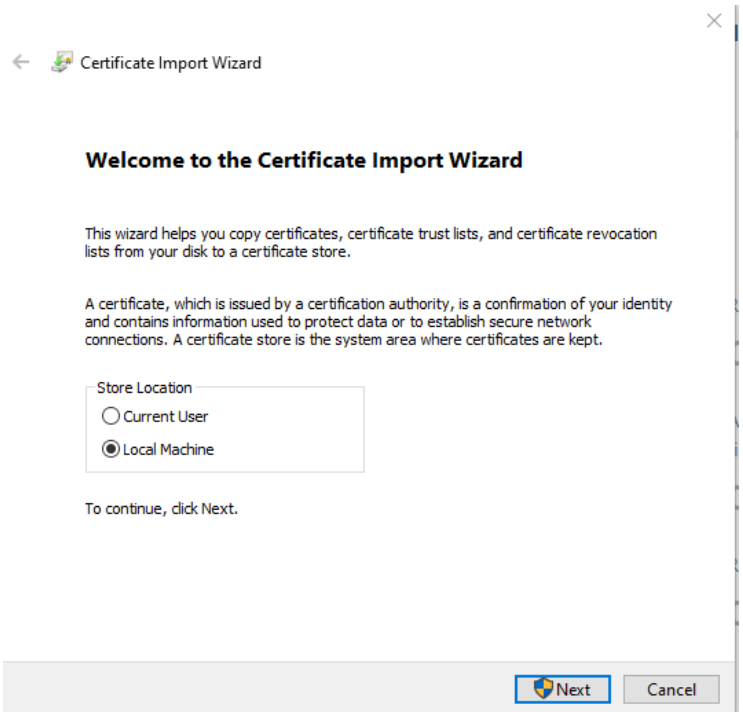
On the Windows PC/Server sending emails it is necessary to have a certificate in the .p12 format. It is a binary format used to save the certificate and its private key. The .p12 file contains the certificate and private key, and it is password protected. It can be obtained with an export of the certificate:



If the .p12 file must be installed on a Windows PC/Server, and you want to send signed emails with Launcher, then when importing the .p12 file, "Mark this key as exportable" must be selected:



In addition, the certificate must be saved in "Local Computer", in "Trusted Root Certification":



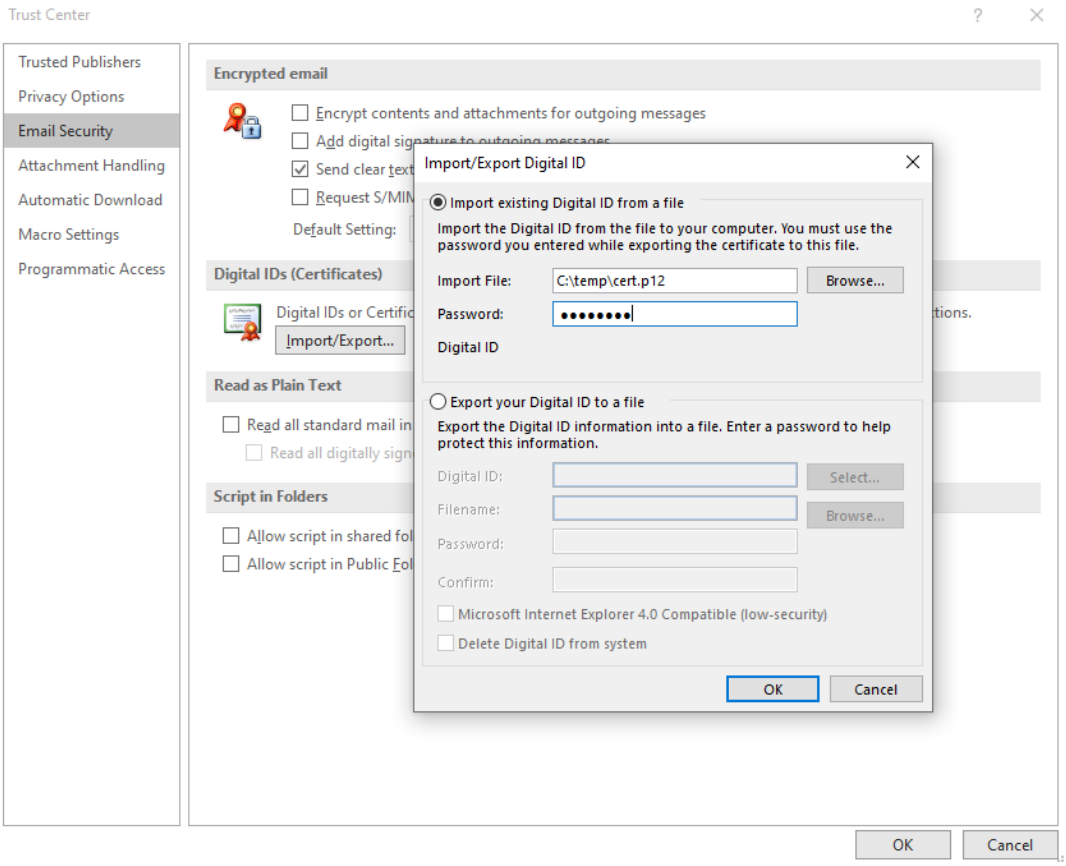
The certificates installed in "Local machine" can be managed with: certlm.msc.

Certificate on recipient side

On the Windows PC/Server of the email recipient, the certificate must be installed.

To install certificate on Outlook:

1. Open Outlook. From the File tab, choose Options, then Trust Center, and then Trust Center Settings.
2. Click Email Security, and then Import/Export.
3. Click Browse.... Locate your certificate file (.p12) and click Open.
4. Enter the password used to secure the private key, and click OK.



API Programs with AS400 client

Variables Statement

To use LAUNCHER Office API programs, the programs parameters must be strictly declared.

AS/400 programs variables statement :

LAUNCHER Office programs require input and output parameters.

Name	Type	Size	Description
HANDLE	CHAR	50	Conversation ID between job and PC.
SOVRAN	CHAR	30	Target PC Name or IP address.
CCSID	CHAR	10	AS/400 job CCSID.
CMD	CHAR	10	Command send to PC Verb.
OPT	CHAR	1	Command Option.
PARM1	CHAR	512	Command Parameters.
PARM2	CHAR	1024	Command Parameters (Complements)
RESULT	CHAR	512	PC returned Result.

Also see : Command PROPERTY

Statements examples :

CL :

```
 DCL VAR(&HANDLE) TYPE(*CHAR) LEN(50) VALUE('*ONLY')
 DCL VAR(&SVRADD) TYPE(*CHAR) LEN(30) VALUE('*DEV')
 DCL VAR(&CCSID) TYPE(*CHAR) LEN(10) VALUE('*JOB')
 DCL VAR(&CMD) TYPE(*CHAR) LEN(10)
 DCL VAR(&OPT) TYPE(*CHAR) LEN(1)
 DCL VAR(&PARM1) TYPE(*CHAR) LEN(512)
 DCL VAR(&PARM2) TYPE(*CHAR) LEN(1024)
 DCL VAR(&RESULT) TYPE(*CHAR) LEN(512)
```

RPG ILE :

```
 DHANDLE S 50 inz('*ONLY')
 DSVRADD S 30 inz('*DEV')
 DCCSID S 10 inz('*JOB')
 DCMD S 10 inz('*DEV')
 DOPT S 1 inz(*blanks)
 DPARM1 S 512 inz(*blanks)
 DPARM2 S 1024 inz(*blanks)
 DRESULT S 512 inz(*blanks)
```

All commands may return errors messages through the standard AS/400 errors managing process. These messages are identified with a MSGID : ***LNCnnnn***.

Exemple :

The following example opens a single connection (&HANDLE value is *ONLY) on PC housing the calling emulator (using *DEV device) and the current job CCSID.

```
DCL                     VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL                     VAR(&SVRADDR) TYPE(*CHAR) LEN(30)
DCL                     VAR(&CCSID) TYPE(*CHAR) LEN(10)

CHGVAR                  VAR(&HANDLE) VALUE('*ONLY')
CHGVAR                  VAR(&SRVADDR) VALUE('*DEV')
CHGVAR                  VAR(&CCSID) VALUE('*JOB')

CALL                    PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
MONMSG                  MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

ILE Programming

ILE procedures for LAUNCHER Office are defined in service program "LAUNCHER" in library LAUNCHER.

To create ILE programs using LAUNCHER Office, create the modules with the compiler, and then, create the program bound to the service program :

```
CRTPGM PGM(MYPROG) MODULE(MYMODULE) … BNDSRVPGM(LAUNCHER) …
```

In your source code, when calling a LAUNCHER Office procedure, choose the calling methods from the programming language, that send the "***Operational Descriptor***" to the called procedure.

In **CL**, "*Operational descriptor*" is always sent:

```
 CALLPRC PRC'(LNCCMD) PARM(&HANDLE 'WORDOPEN' +
*OMIT +
'New;Visible=True' *OMIT &RESULT)
```

With **CALLB**, in RPG ILE, use option (D):

```
CALLB(D) 'LNCCMD'
PARM Handle
PARM Cmd
PARM Opt
PARM Parm1
PARM Parm2
PARM Result
```

With **CALLP**, in RPG ILE:

```
D lnccmd PR ExtProc('LNCCMD') OPDESC
D handle 50 const options(*varsize)
D cmd 10 const options(*varsize)
D opt 1 const options(*varsize:*omit)
D parm1 16000 const options(*varsize:*omit)
D parm2 16000 const options(*varsize:*omit)
D result 16000 options(*varsize:*omit)
c callp lnccmd(handle:'SHELL':'0':
c parm1:parm2:result)
```

See the samples sources in files **QCLSRC** and **QRPGSRC**, library LAUNCHER.

BATCH job for an AS/400 client

LAUNCHER Office can process BATCH job.

The special value ***DEV** cannot be used in BATCH job as name for target PC.

PC name or IP address must be used to open conversation.

Current interactive job PC associated to terminal IP address can be found with API program LNCOPEN :

See example: How to display the name of the user profile ?

Interactive job for an AS/400 client

In interactive job, AS/400 program may use *DEV value to name the target PC for « PC Server address» parameter.

LAUNCHER Office will then look for screen IP address associated to the current job.

In an interactive job, Word or Excel documents may be displayed to the user. In this case, opened application overlays the currently used terminal emulation.

LAUNCHER Office commands allow to create interactivity between user and AS/400 program.

Display of : customised menus, message boxes, Windows directories browser screens, and PC programs launching can be performed.

AS/400 program can set itself in standby mode waiting for a user action, before further processing.

LNCOPEN - API Program

Open communication between AS/400 and a PC.

LAUNCHER Office demon must be launched on the target PC.

Syntax

```
DCL                 VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL                 VAR(&SVRADDR)TYPE(*CHAR) LEN(30)
DCL                 VAR(&CCSID) TYPE(*CHAR) LEN(10)

CHGVAR              VAR(&HANDLE) VALUE('*ONLY')
CHGVAR              VAR(&SRVADDR) VALUE('*DEV')
CHGVAR              VAR(&CCSID) VALUE('*JOB')

CALL                PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
MONMSG              MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
HANDLE (E/S)	Descriptor returned by LNCOPEN function. This variable will be used with all **LAUNCHER Office** functions. If a single communication is opened to the PC, &HANDLE variable ***ONLY** will be used to initialise. Special value ***GETDOT** allows to recover PC TCP/IP address in SVDADDR parameter.
SVRADDR (E/S)	PC network name or TCP/IP address. Use ***DEV** value to name the terminal emulated PC where the current interactive session is opened**.** If HANDLE parameter value is *GETDOT, SVRADDR will return the PC TCP/IP address.
CCSID (E)	CCSID is used by AS/400 programs. Set this value to ***JOB** to use current job CCSID.

Note

From OS/400 version V3R7 (CUMUL C8069370) AS/400 DEVICE may be used to recover emulation IP address.

Specifying ***DEV** as target address, allows conversation between the program and emulated PC.

In order to use LAUNCHER Office in a « Batch » job communicating with user's PC, the interactive program must :

- Recover PC address, using HANDLE=*GETDOT, and SVRADDR=*DEV
- Submit the « Batch » job by transferring the received SVRADDR parameter value to the program.

Example

Open a conversation with the interactive user's PC.

```
DCL VAR(&SVRADDR) TYPE(*CHAR) LEN(30)
DCL VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL VAR(&CCSID) TYPE(*CHAR) LEN(10) VALUE('*JOB')
. . .
CHGVAR VAR(&SVRADD) VALUE('*DEV')
CHGVAR VAR(&HANDLE) VALUE(*ONLY')
CALL PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Open a conversation with the PC to address "182.12.13.14"

```
. . .
CHGVAR VAR(&SVRADD) VALUE('182.12.13.14')
CHGVAR VAR(&HANDLE) VALUE(*ONLY')
CALL PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Recover interactive user's PC IP address.

```
. . .
CHGVAR VAR(&SVRADD) VALUE(*DEV')
CHGVAR VAR(&HANDLE) VALUE(*GETDOT')
CALL PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
/*&SVRADDR contains the IP address in return. */
```

LNCCMD - API Program

When a conversation is opened with LNCOPEN, **LNCCMD program allows command to be sent to PC.**

This is the main program call that application will repeat until connection is closed with LNCCLOSE.

Syntax

```
DCL                     VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL                     VAR(&CMD) TYPE(*CHAR) LEN(10)
DCL                     VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL                     VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL                     VAR(&PARM2) TYPE(*CHAR) LEN(1024)
DCL                     VAR(&RESULT) TYPE(*CHAR) LEN(512)

CHGVAR                  VAR(&CMD) VALUE('Command')
CHGVAR                  VAR(&OPT) VALUE('Option')
CHGVAR                  VAR(&PARM1) VALUE('First parameter')
CHGVAR                  VAR(&PARM2) VALUE('Second parameter')
CALL                    PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1+
                        &PARM2 &RESULT)
MONMSG                  MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters		
HANDLE (E)		Descriptor returned by LNCOPEN function or *ONLY.
CMD	E	Command to run on PC key word. See : Commands list
OPT	E	Option value according to the referenced &CMD command.
PARM1	E	These parameters are assigned according to specified key word in &CMD parameter. See each key word detail in Commands list
PARM2	E	
RESULT	S	Returned result, if the requested command returns a value or a message.

LNCCLOSE - API Program

Close communication with the PC.

Syntax

```
DCL                 VAR(&HANDLE) TYPE(*CHAR) LEN(50)

CALL                PGM(LNCCLOSE) PARM(&HANDLE)
```

Parameters

Parameters	
HANDLE (E)	Descriptor returned by LNCOPEN function.

This command closes network communication between AS/400 and PC.

Example

```
DCL                 VAR(&HANDLE) TYPE(*CHAR) LEN(50)

CALL                PGM(LNCCLOSE) PARM(&HANDLE)
```

If several connections are supported their ID will be contained in the &Handle variable. Use *ONLY for a single connection.

LNCWBM - API Program

LNCWBM program allows to send, in a Word document, several bookmarks values in a single call.

Syntax

```
DCL             VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL             VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL             VAR(&VAR1) TYPE( . . .
DCL             VAR(&VAR2) TYPE( . . .
DCL             VAR(&VARN) TYPE( . . .

CHGVAR          VAR(&PARM1) VALUE('bookmark names, and variables
                formats')

CALL            PGM(LNCWBM) PARM(&HANDLE &PARM1 +
                &VAR1 &VAR2 . . . &VARN)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
&PARM1	Each bookmark value set by the command is displayed in brackets, as to syntax : **(Name1:Fnn.dd)(Name2:Fnn.dd)** **Name 1 and Name 2** are the bookmark names. F is AS/400 program variable type. A = Character Type. P = Condensed decimal Type (PACK). S = Extended decimal Type. **nn** is the variable figures or characters number. **dd** is the decimals number. If the bookmark values are in a DS variable, then they are showed as follow : **DS:((Name 1:Fnn.dd)(Name 2:Fnn.dd)…)**
&VAR…	Variables including values to assign to bookmark are over. Variables number and variables order match with what &PARM1 describes.

Examples

In the bellow example, program variables :

```
&ZPOLNV, &INTD, &NAMED, &IDADR1, &ZDVILL, &ACTIVITE
```

Contain themselves a dedicated bookmark value.

&DS variable is at least a 60 characters variable. The variable contains the values to be assigned to two bookmarks : NAME (position 1 to 20) and ADQUERAB (position 21 to 60).

```
CHGVAR VAR(&PARM1) +
VALUE('(ZPOLNV:A10)(ZINTD:A20)(Z NAMED:A20) +
(IDADR1:A20)(ZDVILL:A20)DS:(( NAME:A20)(ADQUERAB:A40)) +
(ACTIVITE:P8.2)')
CALL PGM(LNCWBM) PARM(&HANDLE &PARM1 &ZPOLNV &INTD &NAMED +
&IDADR1 &ZDVILL &DS +
&ACTIVITE)
```

LNCWFF - API Program

LNCWFF program allows to send, in a Word document, several forms values in a single call.

Syntax

```
DCL             VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL             VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL             VAR(&VAR1) TYPE( . . .
DCL             VAR(&VAR2) TYPE( . . .
DCL             VAR(&VARN) TYPE( . . .

CHGVAR          VAR(&PARM1) VALUE('forms names, and formats
                variables')

CALL            PGM(LNCWBM) PARM(&HANDLE &PARM1 +
                &VAR1 &VAR2 . . . &VARN)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
&PARM1	Each form value set by the command is displayed in brackets, as to syntax : **(Name1:Fnn.dd)(Name2:Fnn.dd)** **Name 1 and Name 2** are the forms names. F is AS/400 program variable type. A = Character Type. P = Condensed decimal Type (PACK). S = Extended decimal Type. **nn** is the variable figures or characters number. **dd** is the decimals number. If the forms values are in a DS variable, then they are showed as follow : **DS:((Name 1:Fnn.dd)(Name 2:Fnn.dd)...)**
&VAR...	Variables including values to assign to forms are over. Variables number and variables order match with what &PARM1 describes.

Exemples

In the bellow example, program variables :

```
&ZPOLNV, &INTD, &NAMED, &IDADR1, &ZDVILL, &ACTIVITE
```

Contain themselves a dedicated form value.

&DS variable is at least a 60 characters variable. The variable contains the values to be assigned to two forms : NAME (position 1 to 20) and ADQUERAB (position 21 to 60).

```
CHGVAR VAR(&PARM1) +
VALUE('(ZPOLNV:A10)(ZINTD:A20)(ZNAMED:A20) +
(IDADR1:A20)(ZDVILL:A20)DS:((NAME:A20)(ADQUERAB:A40)) +
(ACTIVITE:P8.2)')
CALL PGM(LNCWBM) PARM(&HANDLE &PARM1 &ZPOLNV &INTD &NAMED +
&IDADR1 &ZDVILL &DS +
&ACTIVITE)
```

LNCWVA - API Program

LNCWVA program allows to send, in a Word document, several variables values in a single call.

Syntax

```
DCL             VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL             VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL             VAR(&VAR1) TYPE( . . .
DCL             VAR(&VAR2) TYPE( . . .
DCL             VAR(&VARN) TYPE( . . .

CHGVAR          VAR(&PARM1) VALUE('names of variables, and formats
                variables')

CALL            PGM(LNCWBM) PARM(&HANDLE &PARM1 +
                &VAR1 &VAR2 . . . &VARN)
MONMSG          MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

Parameters

Parameters	
&PARM1	Each variable value set by the command is displayed in brackets, as to syntax : **(Name1:Fnn.dd)(Name2:Fnn.dd)** **Name 1 and Name 2** are the variables names. F is AS/400 program variable type. A = Character Type. P = Condensed decimal Type (PACK). S = Extended decimal Type. **nn** is the variable figures or characters number. **dd** is the decimals number. If the variable values are in a DS variable, then they are showed as follow : **DS:((Name 1:Fnn.dd)(Name 2:Fnn.dd)…)**
&VAR…	Variables including values to assign to variable are over. Variables number and variables order match with what &PARM1 describes.

Examples

In the bellow example, program variables :

```
&ZPOLNV, &INTD, &NAMED, &IDADR1, &ZDVILL, &ACTIVITE
```

Contain themselves a dedicated form value.

&DS variable is at least a 60 characters variable. The variable contains the values to be assigned to two variables : NAME (position 1 to 20) and ADQUERAB (position 21 to 60).

```
CHGVAR VAR(&PARM1) +
VALUE('(ZPOLNV:A10)(ZINTD:A20)(ZNAMED:A20) +
(IDADR1:A20)(ZDVILL:A20)DS:((NAME:A20)(ADQUERAB:A40)) +
(ACTIVITE:P8.2)')
CALL PGM(LNCWBM) PARM(&HANDLE &PARM1 &ZPOLNV &INTD &NAMED +
&IDADR1 &ZDVILL &DS +
&ACTIVITE)
```

LNCCVTWCS - API Program

The API program LNCCVTWCS converts the contain of a character variable from or to the Unicode Windows character set.

Syntax

```
DCL             VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL             VAR(&DIRECTION) TYPE(*CHAR) LEN(10)
DCL             VAR(&WCS_VAR) TYPE(*CHAR) LEN(...)
DCL             VAR(&WCS_SIZE) TYPE(*DEC) LEN(5)
DCL             VAR(&CHAR_VAR) TYPE(*CHAR) LEN(...)
DCL             VAR(&CHAR_SIZE) TYPE(*DEC) LEN(5)
DCL             VAR(&CCSID) TYPE(*DEC) LEN(5)
DCL             VAR(&RES_SIZE) TYPE(*DEC) LEN(5)
DCL             VAR(&OPTION) TYPE(*CHAR) LEN(...)

CALL            PGM(LNCCVTWCS) PARM(&HANDLE &DIRECTION +
                &WCS_VAR &WCS_SIZE +
                &CHAR_VAR &CHAR_SIZE +
                &CCSID &RES_SIZE &OPTION)
```

Parameters

Parameters	
&DIRECTION	This parameter contains : ***TO** to convert the value of parameter &CHAR_VAR to the Windows Unicode character set. ***FROM** to convert the value of parameter &CHAR_VAR from the Windows Unicode character set, to another character set.
&WCS_VAR	Input or Output : The string value, coded with the Windows Unicode character set.. When **&DIRECTION='*TO'**, &WCS_VAR contains the result of the conversion. When **&DIRECTION='*FROM'**, &WCS_VAR contains the string to convert.
&WCS_SIZE	Size in bytes of parameter &WCS_VAR. When **&DIRECTION='*TO'**, &WCS_SIZE is the size allocated to variable &WCS_VAR. When **&DIRECTION='*FROM'**, &WCS_SIZE is the size to convert in &WCS_VAR. If &WCS_SIZE is equal to –1, then, the string value in &WCS_VAR must be ended by a null character..
&CHAR_VAR	Input or Output : The character value, coded with the character set

	specified by parameter &CCSID. When **&DIRECTION='*TO'**, &CHAR_VAR is the string to convert. When **&DIRECTION='*FROM'**, &CHAR_VAR will be the result of the conversion.
&CHAR_SIZE	Size in bytes of parameter &CHAR_VAR. When **&DIRECTION='*TO'**, &CHAR_SIZE is the number of characters to convert in &WCS_VAR. When **&DIRECTION='*FROM'**, &CHAR_SIZE is the size allocated to variable &CHAR_VAR.
&CCSID	Character set for the value of &CHAR_VAR. The Unicode value of &WCS_VAR will be converted to or from the character set specified in &CCSID. If &CCSID is 0, the current job CCSID is used.
&RES_SIZE	Output. Size in bytes of the resulting string in &WCS_VAR or &CHAR_VAR.
&OPTION	Possible values are : **NZSTR** : (Non zero string) When **&DIRECTION='*TO'**, the Unicode result will not be terminated by a null character. By default, a null character is added at the end of the Unicode string, if there is not already one in the source string. When **&DIRECTION='*FROM'**, if the Unicode string was terminated by a null character, it will be removed in the converted string By default, if a null character exists at the end of the Unicode string, it is kept.

The Unicode character set handled by LNCCVTWCS
is not the UCS2 character on OS/400.

When used with XLCELLS and WBOOKMARK in LAUNCHER Office, this API program allows to read and write text in Excel and Word, using foreign character sets (Chinese, Korean, …), and use DBCS character sets on AS/400.

Examples

```
                /* Read value of cell B2 in Unicode Windows */
CHGVAR          VAR(&CMD) VALUE('XLCELLS')
CHGVAR          VAR(&PARM1) VALUE('Ref="$B$2";GetText=True;Unicode=True')
CHGVAR          VAR(&PARM2) VALUE(' ')
CALL            PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
```

```
                &PARM2 &RESULT)
                /* &RESULT contains value of B2 in Unicode */
                /* ended by a null character */

                /* Convert &RESULT to Chinese CCSID */
CHGVAR          VAR(&WCS_LEN) VALUE(-1)
CHGVAR          VAR(&CCSID) VALUE(835)
CHGVAR          VAR(&CHINA_SIZE) VALUE(1000)
CALL            PGM(LNCCVTWCS) PARM(&HANDLE '*FROM ' &RESULT &WCS_LEN +
                &CHINA &CHINA_SIZE &CCSID &RES_SIZE)
```

LNCSETKW - API Program

Program LNCSETKW inserts into variable &PARMx, an expression in the form: 'KeyWord=Value'.

With LNCSETKW make easier the construction of the parameters need for program LNCCMD.

If the keyword already exists in &PARMx, its value is replaced.

Syntax

```
DCL             VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL             VAR(&PARMX) TYPE(*CHAR) LEN(512)
DCL             VAR(&KEYW) TYPE(*CHAR) LEN(...)
DCL             VAR(&TYPE) TYPE(*CHAR) LEN(10)
DCL             VAR(&VALUE) TYPE( . . .
DCL             VAR(&FORMAT) TYPE(*CHAR) LEN(10)

CALL            PGM(LNCSETKW) PARM(&HANDLE &PARMX +
                &KEYW &TYPE &VALUE &FORMAT)
```

Parameters

Parameters	
&PARMX	&PARMX is the variable to change. This variable will be used in a call to LNCCMD. When programming in OPM mode (non ILE), variable &PARMX 512.
&KEYW	&KEYW contains the keyword to insert in &PARMX.. The keyword can be inside doubles quotes ("), or, it must be ended by a punctuation character or a blank. Example: **CHGVAR VAR(&KEYW) VALUE('Pattern;')** If &KEYW is blank, value of parameter &VALUE will be added to &PARMX, with a semicolon before (;) if &PARMX is not empty.
&TYPE	&TYPE is the type of the value given to the keyword. Possible values are: ***STR**: Character string type; Double quotes ("") will be added to the value given in parameter &VALUE. ***DEC**: Numeric; ***BOO**: Boolean; &VALUE contain value : '1' or 'T' for True, '0' or 'F' for False. ***CAT**: Value of &VALUE will be concatenated to the value already set in &PARMX for the keyword. ***CLR**: The keyword and its value are deleted from &PARMX.
&VALUE	Contains the value to affect to Keyword
&FORMAT	Format of parameter &VALUE, in the form : **Tn.d** **T** is the data type:

A for character.
P for packed decimal
S for zoned decimal.
n is the number of digits or characters in &VALUE.
d is the number of digits after the decimal point for a numeric value.

Example

After the execution of the following code :

```
CHGVAR  VAR(&PARPM1) VALUE(' ')
CHGVAR  VAR(&KEYW) VALUE('Title ')
CHGVAR  VAR(&TYPE) VALUE('*STR')
CHGVAR  VAR(&VALUE) VALUE('Open a document')
CHGVAR  VAR(&FORMAT) VALUE('A50')
CALL  PGM(LNCSETKW) PARM(&HANDLE &PARM1 +
      &KEYW &TYPE &VALUE &FORMAT)

CALL  PGM(LNCSETKW) PARM(&HANDLE &PARM1 +
      'FileMustExist' '*BOO ' '1')
```

Variable &PARM1 has value:

```
'Title="Open a document";FileMustExist=True'
```

LNCGETKW - API Program

Program LNCGETKW gets the value associated with a keyword on return of a call to LNCCMD.

This program can also convert a number stored in character format in a variable, into packed or zoned decimal format.

Syntax

```
DCL             VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL             VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL             VAR(&KEYW) TYPE(*CHAR) LEN(...)
DCL             VAR(&VALUE) TYPE( . . .
DCL             VAR(&FORMAT) TYPE(*CHAR) LEN(10)

CALL            PGM(LNCGETKW) PARM(&HANDLE &PARM1 +
                &KEYW &VALUE &FORMAT)
```

Parameters

Parameters	
&PARMX	Variable &PARMX contains the value returned by a previous call to LNCCMD.
&KEYW	&KEYW contains the keyword for witch we want to get the value. The keyword can be inside doubles quotes ("), or, it must be ended by a punctuation character or a blank. Example: **CHGVAR VAR(&KEYW) VALUE('Pattern;')** If the first character of &KEYW is an asterix (*), then the next characters represent a data format, in the form : **Tn.d .** The text stored in &PARMX will be then converted into a decimal format in &VALUE. **T** is the data type: A for character. P for packed decimal S for zoned decimal. **n** is the number of digits or characters in &VALUE. **d** is the number of digits after the decimal point for a numeric value. Example: **CHGVAR VAR(&KEYW) VALUE('*P8.2 ')**
&VALUE	Variable where the value associated to the keyword must be stored.
&FORMAT	Format of parameter &VALUE, in the form : **Tn.d** **T** is the data type: A for character. P for packed decimal

	S for zoned decimal. **n** is the number of digits or characters in &VALUE. **d** is the number of digits after the decimal point for a numeric value. If &FORMAT is absent, variable &VALUE is considered as character type, and its size will be not controlled. .

Example

```
DCL  VAR(&SBJ) TYPE(*CHAR) LEN(50)
DCL  VAR(&SND) TYPE(*CHAR) LEN(50)
...
CALL  PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RES)

/* Here &RES contains: */
/* 'Subject="My message";SenderName="Accounting service"' */
CALL PGM(LNCGETKW) PARM(&HANDLE &RES 'Subject ' &SBJ 'A50')
CALL PGM(LNCGETKW) PARM(&HANDLE &RES 'SenderName ' &SND 'A50')

/* After the calls: */
/* &SBJ = 'My message' */
/* &SND = 'Accounting service' */
```

Programs to connect PC and AS400

Communication between AS/400 And PC

LAUNCHER Office offers a functions library called LNCMsg allowing all PC applications to send messages to AS/400.

All of those functions are operated in CALLEXE example.

SHELL : to call a program on a PC.
LNCMessage, LNCRcvMessage, LNCSetPostMessage : to exchange messages.

LNCMessage

To send a message to the AS/400. Also enables you to warn the AS/400 that the PC application has terminated.

Prototype

```
long LNCMessage (long lRetCode, char *szRetString);
```

Parameters

Parameters	
lRetCode (I)	Return code interpreted by **LAUNCHER Office**.
szRetString (I)	Message to send to AS/400

Note

With **lRetCode** set to 0 or 1, set **szRetString** to "***END**".

See also

- LNCRcvMessage
- LNCSetPostMessage

LNCRcvMessage

The application waits for a message from the AS/400.

Using this function, the input parameters sent by the AS/400 will be available when the application is called for the second time.

Prototype

```
Long LNCRcvMessage          (long lRetCode, char *szRetString);
```

Parameters

Parameters	
lRetCode (O)	Return code
szRetString (O)	Message from the AS/400

See also

- LNCMessage
- LNCSetPostMessage

LNCSetPostMessage

Enables an event usable by **LAUNCHER Office**.

Prototype

```
Long LNCSetPostMessage              (long hHwnd,
                                    long lMessage,
                                    long wParam,
                                    long lParam);
```

Parameters

Parameters	
HHwnd	Target window
LMessage	Message to send to the window
Wparam	First message parameter
Lparam	Second message parameter

See also

- LNCMessage
- LNCRcvMessage

CL programs samples

CallExe

This example shows how to send and receive messages to and from the AS/400.

Description

When the message *EXE* is received, the server executes a program containing the dialog box shown below. LAUNCHER Office then just waits for an event sent by the application.

A click on the appropriate button and the message is sent to the AS/400 :

When the "Message to AS/400" button is clicked, the program uses the LNCMessage function found in the LNCMSG.DLL library, and thus allows messages to be sent to the AS/400.

Pressing the "Back to Terminal" button ends the program. The AS/400 then passes back into command mode, while the PC is placed in the 'waiting for connection' mode.

On AS/400 :

Parameters or command

= = = >Call callexec '194.206.160.97'

Message To send to the AS/400

Message send by CALLEXE program

Source of the CALLEXE program (Visual Basic source)

Declarations :

```
' LAUNCHER/400 functions from LNCMSG.DLL
' ---------------------------------
' Send a message to AS/400 and give hand to it
Declare Function LNCMessage Lib "lncmsg.dll" (ByVal nRetCode As Long,
ByVal szRetString As String) As Long
' Receive a message from AS/400
Declare Function LNCRcvMessage Lib "lncmsg.dll" (piRetCode As Long,
ByVal szRetString As String) As Long
' Set the reference to event we want to receive from AS/400
Declare Function LNCSetPostMessage Lib "lncmsg.dll" (ByVal hwnd As
Long, ByVal Msg As Long, ByVal wParam As Long, ByVal lParam As Long)
As Long


' Windows function
' -----------------
```

```
' Get the ID of a control in the form.
Declare Function GetDlgCtrlID Lib "user32" (ByVal hwnd As Long) As
Long
Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal
hwnd As Long, ByVal wMsg As Long, ByVal wParam As Long, ByVal lParam
As Long) As Long

Global Const WM_COMMAND As Long = &H111
Global Const BN_CLICKED As Long = 0

Global Const RET_REJECT = 0
Global Const RET_ACKNOWLEDGE = 1
Global Const RET_MESSAGE = 2
```

On the Message To AS/400 button :

```
Dim ret As Long
Dim RetCode As Long
Dim RetString As String * 100
Dim DlgId As Integer
'LAUNCHER/400 will simulate a click on button 'Command3'
DlgId = GetDlgCtrlID(Command3.hwnd)
ret = LNCSetPostMessage(Form1.hwnd, WM_COMMAND, DlgId, Command3.hwnd)
'Send a message to the AS/400 application.
ret = LNCMessage(2, Text1.Text)
```

On the Back To Terminal button :

```
Dim ret As Long

'Tell to AS/400 that we stop to talk
ret = LNCMessage(2, "*END")
```

On the Command3 button (hidden) :

```
Dim ret As Long
Dim str As String * 200

' This button was clicked by LAUNCHER/400,
' when the AS/400 call the application for
' the 2nd time and more.
' We can receive a message from the AS/400
' application.
ret = LNCRcvMessage(ret, str)
Text1.Text = str
```

For the first execution, after clicking on the "**Message to AS/400**" button, **LAUNCHER Office** will simulate the "**Click on the Command3 button**" event. Then the application will be waiting for a callback message.

When the event is simulated, the application sends the Text1 control's text to the AS/400.

When the AS/400 calls the application the second time, the application will run from the Command3 code, and will retrieve the calling parameters furnished by the AS/400 to LAUNCHER.

Source of the CALLEXE program (CL language)

```
/* Variables used for opening the communication */
PGM         PARM(&SVRADDR)
DCL         VAR(&SVRADDR) TYPE(*CHAR) LEN(30)
DCL         VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL         VAR(&CCSID) TYPE(*CHAR) LEN(10)

/* Variables used for sending the commands */
DCL         VAR(&CMD) TYPE(*CHAR) LEN(10)
DCL         VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL         VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL         VAR(&PARM2) TYPE(*CHAR) LEN(1024)
DCL         VAR(&RESULT) TYPE(*CHAR) LEN(512)
DCL         VAR(&MSGID) TYPE(*CHAR) LEN(7)
DCL         VAR(&MSG1) TYPE(*CHAR) LEN(80)
DCL         VAR(&MSG2) TYPE(*CHAR) LEN(80)

/* Opening of only a single communication with the PC*/
CHGVAR      VAR(&HANDLE) VALUE('*ONLY')

/* Translates the CCSID depending on the job CCSID */
CHGVAR      VAR(&CCSID) VALUE('*JOB')

/* Connection to the PC */
CALL        PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* Save the active window on the PC */
CHGVAR      VAR(&CMD) VALUE('STO')
CHGVAR      VAR(&PARM1) VALUE(' ')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
            &PARM2 &RESULT)
```

```
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* Minimizes the active window(Emulation AS400) */
CHGVAR      VAR(&CMD)VALUE('MIN')
CHGVAR      VAR(&PARM1)VALUE(' ')
CHGVAR      VAR(&PARM2)VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
            &PARM2 &RESULT)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* Begin communication using the application CALLEXE.EXE, OPT=
/* Then the demon sends a message to the AS 400 following an   */
/* application event */

CALLIT:
CHGVAR      VAR(&CMD) VALUE('EXE')
CHGVAR      VAR(&OPT) VALUE('2')
CHGVAR      VAR(&PARM1) VALUE('%LNCDIR%\SAMPLES\CALLEXE.EXE')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
            &PARM2 &RESULT)

MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
SNDPGMMSG   MSGID(CPF9898) MSGF(QSYS/QCPFMSG) MSGDTA(&RESULT) TOPGMQ
+
(*EXT) MSGTYPE(*STATUS)
IF          COND(%SST(&RESULT 1 4) =
GOTO        CMDLBL(CALLIT)

FINISH:

/* Minimize the active window (Application CALLEXE) */
CHGVAR      VAR(&CMD) VALUE('MIN')
CHGVAR      VAR(&PARM1) VALUE(' ')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* Restore the saved window (Emulation AS/400)*/
CHGVAR      VAR(&CMD) VALUE('RCL')
CHGVAR      VAR(&OPT) VALUE('0')
CHGVAR      VAR(&PARM1) VALUE(' ')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)
```

```
/* End of communication with the PC */
CLOSE:
CHGVAR      VAR(&CMD) VALUE('END ')
CHGVAR      VAR(&PARM1) VALUE(' ')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 &PARM2 &RESULT)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

CALL        PGM(LNCCLOSE) PARM(&HANDLE)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
GOTO END

ERROR:
CALL        PGM(LNCCLOSE) PARM(&HANDLE)
RCVMSG      MSG(&MSG1) SECLVL(&MSG2) MSGID(&MSGID)
SNDPGMMSG   MSGID(CPF9898) MSGF(QSYS/QCPFMSG) MSGDTA('Error !!!!! ' +
            *CAT &MSGID *CAT ' ' *CAT &MSG1 *CAT ' ' *CAT &MSG2)+
            MSGTYPE(*ESCAPE)
END:
ENDPGM
```

See also the EVENTTEST example in the AS/400 installation library, illustrating the exchange of synchronization messages between the PC and the AS/400.

Merge example

This is a small sample of Launcher's mail merge functions.

The main document is named sp_cust.doc and is located in the Samples directory of LAUNCHER Office. The data source is dynamically transferred from the AS/400 to the PC by the MERGE program.

First the DBFXFER command transfers the SP_CUST file from the LAUNCHER Office library to the PC.

Then the WMAILMERGE command will launch the merge into a new document.

Type CALL LAUNCHER/MERGE to execute it.

Source

```
            PGM
/*  DECLARATIONS */
            DCL         VAR(&HANDLE) TYPE(*CHAR) LEN(50)
            DCL         VAR(&SVRADR) TYPE(*CHAR) LEN(30)
            DCL         VAR(&CCSID) TYPE(*CHAR) LEN(10)
            DCL         VAR(&CMD) TYPE(*CHAR) LEN(10)
            DCL         VAR(&OPT) TYPE(*CHAR) LEN(1)
            DCL         VAR(&PARM1) TYPE(*CHAR) LEN(512)
            DCL         VAR(&PARM2) TYPE(*CHAR) LEN(1024)
            DCL         VAR(&RESULT) TYPE(*CHAR) LEN(512)
            DCL         VAR(&MSGID) TYPE(*CHAR) LEN(7)
            DCL         VAR(&MSG1) TYPE(*CHAR) LEN(80)
            DCL         VAR(&MSG2) TYPE(*CHAR) LEN(80)
/* INITIALISATION */
            CHGVAR      VAR(&HANDLE) VALUE('*ONLY')
            CHGVAR      VAR(&SVRADR) VALUE('*DEV')
            CHGVAR      VAR(&CCSID) VALUE('*JOB')
/* OUVERTURE DE LA SESSION - OPEN THE SESSION  */
            CALL        PGM(LAUNCHER/LNCOPEN) PARM(&HANDLE &SVRADR &CC
SID)
            MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* FILE TRANSFERT - TRANSFERT DU FILE */
            CHGVAR      VAR(&CMD) VALUE('DBFXFER')
            CHGVAR      VAR(&PARM1) VALUE('%LNCDIR%\SAMPLES\SP_CUST.TX
T')
            CHGVAR      VAR(&PARM2) VALUE('LAUNCHER/SP_CUST')
            CALL        PGM(LAUNCHER/LNCCMD) PARM(&HANDLE &CMD &OPT &P
ARM1 +
                        &PARM2 &RESULT)
            MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

```
/*  OUVERTURE WORD - OPEN WORD */
            CHGVAR     VAR(&CMD) VALUE('WORDOPEN')
            CHGVAR     VAR(&PARM1) VALUE(' ')
            CHGVAR     VAR(&PARM2) VALUE(' ')
            CALL       PGM(LAUNCHER/LNCCMD) PARM(&HANDLE &CMD &OPT &P
ARM1 +
                       &PARM2 &RESULT)
            MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* FUSION - MERGE */
            CHGVAR     VAR(&CMD) VALUE('WMAILMERGE')
            CHGVAR     VAR(&PARM1) VALUE('DOCUMENT=
                       .DOC";DATASOURCE=
            CHGVAR     VAR(&PARM2) VALUE(' ')
            CALL       PGM(LAUNCHER/LNCCMD) PARM(&HANDLE &CMD &OPT &P
ARM1 +
                       &PARM2 &RESULT)
            MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* WORDWAIT */
            CHGVAR     VAR(&CMD) VALUE('WORDWAIT')
            CHGVAR     VAR(&PARM1) VALUE(' ')
            CHGVAR     VAR(&PARM2) VALUE(' ')
            CALL       PGM(LAUNCHER/LNCCMD) PARM(&HANDLE &CMD &OPT &P
ARM1 +
                       &PARM2 &RESULT)
            MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* FERMETURE SESSION LAUNCHER - CLOSE LAUNCHER SESSION */
            CHGVAR     VAR(&CMD) VALUE('END')
            CHGVAR     VAR(&PARM1) VALUE(' ')
            CHGVAR     VAR(&PARM2) VALUE(' ')
            CALL       PGM(LAUNCHER/LNCCMD) PARM(&HANDLE &CMD &OPT &P
ARM1 +
                       &PARM2 &RESULT)
            MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
            CALL       PGM(LAUNCHER/LNCCLOSE) PARM(&HANDLE)
            MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
            GOTO       END
ERROR:
            RCVMSG     MSG(&MSG1) SECLVL(&MSG2) MSGID(&MSGID)
            SNDPGMMSG  MSGID(CPF9898) MSGF(QSYS/QCPFMSG) +
                       MSGDTA('ERROR !!!!! ' *CAT &MSGID *CAT ' +

                       ' *CAT &MSG1 *CAT ' ' *CAT &MSG2) +
                       MSGTYPE(*ESCAPE)
END:
            ENDPGM
```

Mail example

The MAILSEND is a CL program that shows how to send a mail with an attached file and a high priority.

You can edit the source code from the QCLSRC file in the LAUNCHER Office library and change the parameters with you own.

Here are the default values :

```
            CHGVAR      VAR(&ADDR) VALUE('INFO@THECOMPAGNY.COM')
            CHGVAR      VAR(&SUBJ) VALUE('THE CUSTOMER LIST')
            CHGVAR      VAR(&MESG) VALUE('Hello,%PARA%%PARA%Here +
                        is the list of the customers you +
                        required. The file comes from our AS/400. +
                        Isn''t this great ?%PARA%%PARA%See you +
                        soon !')
```

Source

```
             PGM
/* VARIABLES USED TO OPEN COMMUNICATION AND SEND COMMANDS ...*/
             DCL        VAR(&HANDLE) TYPE(*CHAR) LEN(50) VALUE('*ONLY
')
             DCL        VAR(&SVRADDR) TYPE(*CHAR) LEN(30) VALUE('*DEV
')
             DCL        VAR(&CCSID) TYPE(*CHAR) LEN(10)  VALUE('*JOB'
)
             DCL        VAR(&CMD) TYPE(*CHAR) LEN(10)
             DCL        VAR(&OPT) TYPE(*CHAR) LEN(1)
             DCL        VAR(&PARM1) TYPE(*CHAR) LEN(512)
             DCL        VAR(&PARM2) TYPE(*CHAR) LEN(1024)
             DCL        VAR(&RESULT) TYPE(*CHAR) LEN(512)
             DCL        VAR(&MSGID) TYPE(*CHAR) LEN(7)
             DCL        VAR(&MSG1) TYPE(*CHAR) LEN(80)
             DCL        VAR(&MSG2) TYPE(*CHAR) LEN(80)
/* VARIABLES FOR THE MAIL */
             DCL        VAR(&ADDR) TYPE(*CHAR) LEN(64)
             DCL        VAR(&SUBJ) TYPE(*CHAR) LEN(64)
             DCL        VAR(&MESG) TYPE(*CHAR) LEN(512)

             CHGVAR     VAR(&ADDR) VALUE('INFO@THECOMPAGNY.COM')
             CHGVAR     VAR(&SUBJ) VALUE('THE CUSTOMER LIST')

             CHGVAR     VAR(&MESG) VALUE('Hello,%PARA%%PARA%Here +
```

```
                          is the list of the customers you +
                          required. The file comes from our AS/400. +
                          Isn''t this great ?%PARA%%PARA%See you +
                          soon !')
/* OPEN THE SESSION */
             CALL       PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* CONNECT TO THE SERVER ...*/

/* MESSAGE IS SENT TO RECIPIENT &ADDR */
/* SUBJECT IS &SUBJ AND MESSAGE IS &MESG */
/* */
             CHGVAR     VAR(&CMD) VALUE('MAILPREP')
             CHGVAR     VAR(&PARM1) VALUE(&ADDR)
             CHGVAR     VAR(&PARM2) VALUE(&SUBJ)
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                                        &PARM2 &RESULT)
             IF         COND(%SST(&RESULT 1 5) =
             SNDMSG     MSG(&RESULT) TOUSR(QPGMR)
 SUI:
 /* TRANSFER THE SP_CUST FILE AND ATTACH IT TO THE MAIL  */
             CHGVAR     VAR(&CMD) VALUE('DBFXFER')
             CHGVAR     VAR(&PARM1) VALUE('%LNCDIR%\SAMPLES\SP_CUST.T
XT')
             CHGVAR     VAR(&PARM2) VALUE('LAUNCHER/SP_CUST')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                        &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

 /* ATTACH THE FILE */
             CHGVAR     VAR(&CMD) VALUE('MAILATT')
             CHGVAR     VAR(&PARM1) VALUE('%LNCDIR%\SAMPLES\SP_CUST.T
XT')
```

```
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
                        &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
 /* SET THE MESSAGE BODY */
             CHGVAR     VAR(&CMD) VALUE('MAILTEXT')
             CHGVAR     VAR(&PARM1) VALUE(&MESG)
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +

                        &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* SET THE PRIORITY */
             CHGVAR     VAR(&CMD) VALUE('MAILPRTY')
             CHGVAR     VAR(&PARM1) VALUE('*HIGH')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +

                        &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* SEND THE MESSAGE */
             CHGVAR     VAR(&CMD) VALUE('MAILSEND')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +

                        &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
             CHGVAR     VAR(&CMD) VALUE('MAILEND')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +

                        &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

CLOSE:
/* END OF COMMUNICATION WITH THE SERVER STATION ... */
             CHGVAR     VAR(&CMD) VALUE('END ')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

```
                GOTO END

ERROR:
                CALL        PGM(LNCCLOSE) PARM(&HANDLE)
                RCVMSG      MSG(&MSG1) SECLVL(&MSG2) MSGID(&MSGID)

                SNDPGMMSG   MSGID(CPF9898) MSGF(QSYS/QCPFMSG) +
                            MSGDTA('ERROR !!!!!! ' *CAT &MSGID *CAT ' +

                            ' *CAT &MSG1 *CAT ' ' *CAT &MSG2) +
                            MSGTYPE(*ESCAPE)
                            GOTO CLOSE
END:
                ENDPGM
```

W_DEMO Example

This example illustrates the main features of LAUNCHER Office for creating a Word document :

- retrieving information from the AS/400 and sending it to the PC (serial number, user, time)
- filling a table
- inserting a document, inserting a picture.

This is the resulting Word document

LAUNCHER Office

AS/400 Serial Number : : **4486D0A**
Current User : **QPGMR**
System Time : **17:51:04**

Your AS/400 created this document, using Word's functionality with LAUNCHER Office features.

The program you just called has created the document from the following template :

« C:\Program Files\LAUNCHER<X>\Samples\LNCMODEL.DOC »,

and the data was read or calculated on the AS/400.

You can see the CL source in the QCLSRC file of the LAUNCHER Office library.

The final document, as you see it now, was saved by the AS/400 program under this name :

« C:\Program Files\LAUNCHER<X>\Samples\LncResult-175157.doc »

Insert data from the AS/400

Calculated values on the AS/400 have been inserted into this document.

The serial number, the current user and system time you can see above all come from the AS/400.

To insert these values, the CL program uses Word **bookmarks.** They are specific positions in the text identified by their name.

To see the bookmarks you need to configure Word :

Menu "Tools" – "Options" – "Display", then check the Bookmark option.

To know the bookmark names :

Menu "Insert" – "Bookmarks"

Insert external paragraphs

This paragraph does not exist in the main document. It is a distinct Word file named "LncModel_Para01.doc", located in Launchers "Samples" directory.

The AS/400 program asked Word to insert this paragraph. This feature is very useful for including specific clauses for which the AS/400 has the references in its database.

This paragraph could also be a free text. LAUNCHER Office can ask the user to type or change text in Word, save it into a distinct file and then insert it in another document to print.

Working with Word tables

LAUNCHER Office can create and fill in Word tables.

The following table exists in the template but without any values, except the column headings.

We display in this table the library list of the AS/400 but it can of course list invoices, articles or any commercial data...

Library	Description	Owner
LAUNCHER	System Library	QSECOFR
QGPL		QSYS
QSYS		QSYS
QHLPSYS		QSYS
QUSRSYS		QSYS

Insert pictures

The picture in the upper left corner was inserted by the CL program.

As for the text, a bookmark is placed into the template and the program just gives the name of the picture file to insert ; « LNCModel_img01.gif ».

This is the CL source

```
             PGM        PARM(&SVRADDR)
             DCL        VAR(&SVRADDR) TYPE(*CHAR) LEN(30)
/* VARIABLES USED TO OPEN COMMUNICATION AND SEND COMMANDS ...*/
             DCL        VAR(&HANDLE) TYPE(*CHAR) LEN(50)
             DCL        VAR(&CCSID) TYPE(*CHAR) LEN(10)
             DCL        VAR(&CMD) TYPE(*CHAR) LEN(10)
             DCL        VAR(&OPT) TYPE(*CHAR) LEN(1)
             DCL        VAR(&PARM1) TYPE(*CHAR) LEN(512)
             DCL        VAR(&PARM2) TYPE(*CHAR) LEN(1024)
             DCL        VAR(&RESULT) TYPE(*CHAR) LEN(512)

             DCL        VAR(&MSGID) TYPE(*CHAR) LEN(7)
             DCL        VAR(&MSG1) TYPE(*CHAR) LEN(80)
             DCL        VAR(&MSG2) TYPE(*CHAR) LEN(80)
             DCL        VAR(&USER) TYPE(*CHAR) LEN(10)
             DCL        VAR(&SRLN) TYPE(*CHAR) LEN(8)
             DCL        VAR(&TIME) TYPE(*CHAR) LEN(7)
             DCL        VAR(&ULIBL) TYPE(*CHAR) LEN(275)
             DCL        VAR(&SLIBL) TYPE(*CHAR) LEN(165)
             DCL        VAR(&LTEXT) TYPE(*CHAR) LEN(50)
             DCL        VAR(&LOWNR) TYPE(*CHAR) LEN(10)
             DCL        VAR(&LLIB) TYPE(*CHAR) LEN(10)

             DCL        VAR(&IDX) TYPE(*DEC) LEN(4)

             DCL        VAR(&DOTTED) TYPE(*CHAR) LEN(30)
             DCL        VAR(&ROW) TYPE(*DEC) LEN(2)
             DCL        VAR(&COL) TYPE(*DEC) LEN(2)
             DCL        VAR(&C_ROW) TYPE(*CHAR) LEN(2)
             DCL        VAR(&C_COL) TYPE(*CHAR) LEN(2)

             RTVJOBA    USER(&USER) USRLIBL(&ULIBL) SYSLIBL(&SLIBL)
             RTVSYSVAL  SYSVAL(QSRLNBR) RTNVAR(&SRLN)
             RTVSYSVAL  SYSVAL(QTIME) RTNVAR(&TIME)
```

```
/* OPEN ONLY ONE CONVERSATION WITH THE SERVER */
             CHGVAR     VAR(&HANDLE) VALUE('*ONLY')
/* TRANSLATE CCSID ACCORDING TO THE JOB CCSID */
             CHGVAR     VAR(&CCSID) VALUE('*JOB')

/* CONNECT TO THE SERVER ...*/
             CALL       PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR0))

/* START WORD ON THE WINDOWS STATION */

             CHGVAR     VAR(&CMD) VALUE('WORDOPEN ')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

             CHGVAR     VAR(&CMD) VALUE('WORDHIDE')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
 /*          CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +     */
 /*                     &PARM1 &PARM2 &RESULT)                  */
 /*          MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))  */

/* OPEN A DOCUMENT FILE ON THE WINDOWS STATION */

             CHGVAR     VAR(&CMD) VALUE('WNEWFILE ')
             CHGVAR     VAR(&PARM1) VALUE('%LNCDIR%\SAMPLES\LNCMODEL
.DOC')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/*  WRITE TEXT AT A BOOKMARK */

             CHGVAR     VAR(&CMD) VALUE('WBOOKMARK ')
             CHGVAR     VAR(&PARM1) VALUE('USER')
             CHGVAR     VAR(&PARM2) VALUE(&USER)
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             CHGVAR     VAR(&PARM1) VALUE('SNAS400')
             CHGVAR     VAR(&PARM2) VALUE(&SRLN)
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
```

```
                    &PARM1 &PARM2 &RESULT)

             CHGVAR     VAR(&PARM1) VALUE('TIME')
             CHGVAR     VAR(&PARM2) VALUE(%SST(&TIME 1 2) *CAT ':' +
                        *CAT %SST(&TIME 3 2) *CAT ':' *CAT +
                        %SST(&TIME 5 2))
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)

             CHGVAR     VAR(&CMD) VALUE('WBOOKMARK ')
             CHGVAR     VAR(&PARM1) VALUE('TABLE')
FRST_ROW:    CHGVAR     VAR(&IDX) VALUE(1)
NEXT_ROW:
             CHGVAR     VAR(&LLIB) VALUE(%SST(&ULIBL &IDX 10))
             CHGVAR     VAR(&IDX) VALUE(&IDX + 11)
             IF         COND(&LLIB *EQ '          ') THEN(GOTO +
                        CMDLBL(END_LIST))
             RTVOBJD    OBJ(&LLIB) OBJTYPE(*LIB) TEXT(&LTEXT) +
                        OWNER(&LOWNR)
             MONMSG     MSGID(CPF0000) EXEC(GOTO CMDLBL(NEXT_ROW))

             CHGVAR     VAR(&PARM2) VALUE(&LLIB)
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             CHGVAR     VAR(&PARM1) VALUE('*RIGHT')
             CHGVAR     VAR(&PARM2) VALUE(&LTEXT)
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             CHGVAR     VAR(&PARM2) VALUE(&LOWNR)
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)

             GOTO       CMDLBL(NEXT_ROW)

END_LIST:
             CHGVAR     VAR(&ULIBL) VALUE(&SLIBL)
             CHGVAR     VAR(&SLIBL) VALUE(*BLANK)
             IF         COND(&ULIBL *NE *BLANK) THEN(GOTO +
                        CMDLBL(FRST_ROW))

             CHGVAR     VAR(&CMD) VALUE(WSELECT)
             CHGVAR     VAR(&PARM1) VALUE(*ROW)
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
```

```
            &PARM1 &PARM2 &RESULT)

CHGVAR      VAR(&CMD) VALUE(WSETPROP)
CHGVAR      VAR(&PARM1) +
            VALUE('Selection.Borders.Item(wdBorderBotto+
            m).LineStyle')
CHGVAR      VAR(&PARM2) VALUE('wdLineStyleSingle')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
            &PARM1 &PARM2 &RESULT)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

CHGVAR      VAR(&CMD) VALUE('WBOOKMARK ')
CHGVAR      VAR(&PARM1) VALUE('PARA01')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
            &PARM1 &PARM2 &RESULT)
CHGVAR      VAR(&CMD) VALUE(WINSERTF)
CHGVAR      VAR(&PARM1) +
            VALUE('%LNCDIR%\SAMPLES\LNCpara01.DOC ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
            &PARM1 &PARM2 &RESULT)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

CHGVAR      VAR(&CMD) VALUE('WBOOKMARK ')
CHGVAR      VAR(&PARM1) VALUE('IMG01')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
            &PARM1 &PARM2 &RESULT)
CHGVAR      VAR(&CMD) VALUE(WINSERTIMG)
CHGVAR      VAR(&PARM1) +
            VALUE('%LNCDIR%\SAMPLES\LNCIMG01.GIF ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
            &PARM1 &PARM2 &RESULT)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))


CHGVAR      VAR(&CMD) VALUE('WORDSHOW')
CHGVAR      VAR(&PARM1) VALUE(' ')
CHGVAR      VAR(&PARM2) VALUE(' ')
CALL        PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
            &PARM1 &PARM2 &RESULT)
MONMSG      MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

CHGVAR      VAR(&CMD) VALUE('WORDWAIT')
```

```
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* END OF COMMUNICATION WITH THE SERVER STATION ... */
CLOSE:
             CHGVAR     VAR(&CMD) VALUE('END ')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
/*           CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +     /*
/*                      &PARM1 &PARM2 &RESULT)                   */
/*           MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))  */

             CALL       PGM(LNCCLOSE) PARM(&HANDLE)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
             GOTO END
ERROR:
             CHGVAR     VAR(&CMD) VALUE('END ')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR0))
ERROR0:
             CALL       PGM(LNCCLOSE) PARM(&HANDLE)

             RCVMSG     MSG(&MSG1) SECLVL(&MSG2) MSGID(&MSGID)
             SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) +
                        MSGDTA('Error !!!!! ' *CAT &MSGID *CAT ' +
                        ' *CAT &MSG1 *CAT ' ' *CAT &MSG2) +
                        MSGTYPE(*ESCAPE)
END:
             SNDPGMMSG  MSGID(CPF9898) MSGF(QCPFMSG) +
                        MSGDTA('Document ''result.doc'' saved by +
                        ' *CAT &USER *CAT 'in samples directory') +
                        MSGTYPE(*COMP)
             ENDPGM
```

X_DEMO Example

This example introduces some of Launcher's functions with Excel.

Sheet management

We first insert a new sheet with the XLADDSHEET command, then we change its name with the XLSHNAME command. With LAUNCHER Office you can also copy or delete sheets from the workbook.

Data insertion

Among all the commands to fill and insert data in a sheet, the easiest and most practical is XLSETLINE witch fills the entire line in one operation.

The first parameter contains the values, separated by the %SEP% keyword.

The second parameter controls each cell's formatting.

We first write a header line in bold characters then 2 value lines, the second one formatted with 2 decimal places with the NUMFMT(2).

Graph insertion

Once the table has been filled in, we create a graph using the XLDRAWGR command in a new sheet. We choose an AREA-type graph, but any of Excel's graph types may be used.

Then we change some of the graph's properties, its name and the name of the series, with the XLSETPROP command.

Finally, after all these operations have been performed, we can make Excel visible with the EXCELSHOW command.

This is the CL source of the example

```
             PGM

             DCL         VAR(&SVRADDR) TYPE(*CHAR) LEN(30)


/* DECLARATION OF VARIABLES NEEDED FOR A CONNECTION TO THE SERVER   *
/
             DCL         VAR(&HANDLE) TYPE(*CHAR) LEN(50)

             DCL         VAR(&CCSID) TYPE(*CHAR) LEN(10)

             DCL         VAR(&CMD) TYPE(*CHAR) LEN(10)

             DCL         VAR(&OPT) TYPE(*CHAR) LEN(1)

             DCL         VAR(&PARM1) TYPE(*CHAR) LEN(512)

             DCL         VAR(&PARM2) TYPE(*CHAR) LEN(1024)
```

```
             DCL        VAR(&RESULT) TYPE(*CHAR) LEN(512)

             DCL        VAR(&MSGID) TYPE(*CHAR) LEN(7)

             DCL        VAR(&MSG1) TYPE(*CHAR) LEN(80)

             DCL        VAR(&MSG2) TYPE(*CHAR) LEN(80)

             CHGVAR     VAR(&SVRADDR) VALUE('*DEV')

/* OPEN ONLY ONE CONVERSATION WITH THE SERVER */
             CHGVAR     VAR(&HANDLE) VALUE('*ONLY')
/* TRANSLATE THE CCSID ACCORDING TO THE JOB CCSOD */
             CHGVAR     VAR(&CCSID) VALUE('*JOB')
/* CONNECTION WITH THE SERVER */
             CALL       PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* OPENING EXCEL */
             CHGVAR     VAR(&CMD) VALUE('EXCELOPEN ')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* ADD A NEW SHEET */
             CHGVAR     VAR(&CMD) VALUE('XLADDSHEET ')
             CHGVAR     VAR(&PARM1) VALUE(' ')

             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +

                        &PARM1 &PARM2 &RESULT)

             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* RENAME THE SHEET */

             CHGVAR     VAR(&CMD) VALUE('XLSHNAME ')

             CHGVAR     VAR(&PARM1) VALUE('VENTES 1ER TRIMESTRE')

             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +

                        &PARM1 &PARM2 &RESULT)

             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* INSERT FIRST LINE WITH HEADERS */

             CHGVAR     VAR(&CMD) VALUE('XLSETLINE  ')

             CHGVAR     VAR(&PARM1) VALUE('JANVIER%SEP%FEVRIER%SEP%MAR
S')
             CHGVAR     VAR(&PARM2) VALUE('PROP(FONT.BOLD)=
                        PROP(FONT.BOLD)=
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +

                        &PARM1 &PARM2 &RESULT)

             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

```
/* FIRST DATA LINE */

             CHGVAR     VAR(&CMD) VALUE('XLSETLINE  ')

            CHGVAR     VAR(&PARM1) VALUE('000123%SEP%22250%SEP%301')

            CHGVAR     VAR(&PARM2) VALUE('FORMAT()%SEP%NUMFMT(2)')

            CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +

                       &PARM1 &PARM2 &RESULT)

            MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* SECOND DATA LINE */

            CHGVAR     VAR(&PARM1) VALUE('223%SEP%23250%SEP%285')
            CHGVAR     VAR(&PARM2) VALUE('FORMAT()%SEP%NUMFMT(2)')
            CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                       &PARM1 &PARM2 &RESULT)

            MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* INSERT THE GRAPHIC */

            CHGVAR     VAR(&CMD) VALUE('XLDRAWGR ')

            CHGVAR     VAR(&PARM1) VALUE('A1:C3')

            CHGVAR     VAR(&PARM2) VALUE('''AREA'' ''ROWS'' ''1'' ''0
'' ''+
                       MARKET'' ')

            CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +

                       &PARM1 &PARM2 &RESULT)

            MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* CHANGING THE GRAPHS TITLE */

            CHGVAR     VAR(&CMD) VALUE('XLSETPROP')

            CHGVAR     VAR(&PARM1) VALUE('ACTIVECHART.CHARTTITLE.+

                       CHARACTERS.TEXT')

            CHGVAR     VAR(&PARM2) VALUE('VENTES TRIMESTRE 1')

             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +

                        &PARM1 &PARM2 &RESULT)

             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))

/* CHANGE FIRST SERIE NAME */

             CHGVAR     VAR(&CMD) VALUE('XLSETPROP')

             CHGVAR     VAR(&PARM1) VALUE('ACTIVECHART.+
```

```
                        SERIESCOLLECTION(1).NAME')

             CHGVAR     VAR(&PARM2) VALUE('PRODUIT A')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* SECOND SERIE */
             CHGVAR     VAR(&CMD) VALUE('XLSETPROP')
             CHGVAR     VAR(&PARM1) VALUE('ACTIVECHART.SERIESCOLLECTI
ON(2)+
                        .NAME')
             CHGVAR     VAR(&PARM2) VALUE('PRODUIT B')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* MAKE EXCEL VISIBLE */
             CHGVAR     VAR(&CMD) VALUE('EXCELSHOW ')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* END */
             CHGVAR     VAR(&CMD) VALUE('END ')
             CHGVAR     VAR(&PARM1) VALUE(' ')
             CHGVAR     VAR(&PARM2) VALUE(' ')
             CALL       PGM(LNCCMD) PARM(&HANDLE &CMD &OPT +
                        &PARM1 &PARM2 &RESULT)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* END OF COMMUNICATION...*/
             CALL       PGM(LNCCLOSE) PARM(&HANDLE)
             MONMSG     MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
```

```
             GOTO       END
ERROR:
             RCVMSG     MSG(&MSG1) SECLVL(&MSG2) MSGID(&MSGID)
             SNDPGMMSG  MSGID(CPF9898) MSGF(QSYS/QCPFMSG) +
                        MSGDTA('ERROR !!!!! ' *CAT &MSGID *CAT ' +
                        ' *CAT &MSG1 *CAT ' ' *CAT &MSG2) +
                        MSGTYPE(*ESCAPE)
END:
             ENDPGM
```

RPG programs samples

Word example

The principle of this example is to fill in an order table.

We placed a bookmark at the first cell of the table that will receive data (on the second line, the first one is reserved for the table's header).

So, the program will create a document based on a template, write data for the document's header, and fill in the table (it uses the **WSETLINE** command).

The table contains only one line, because Word will automatically grow the table, once you are in the right-most cell (and the last one) and you want to go to the next cell.

This means that we will not have any blank lines.

The text field that follows the table will shift down as we fill in the table.

The empty document is as follows (the bookmark indicators are on) :

Word program source (in RPG language)

```
*****************************************************************
* Files used :       SP_CUST --> Customers file                 *
*                   SP_ORD_ID --> Logical file on ORDERS file   *
*                   SP_DET_OR --> Logical file on DETAIL file   *
*****************************************************************
     FSP_CUST   if   e           k disk
     FSP_ORD_ID if   e           k disk
     FSP_DET_OR if   e           k disk

*****************************************************************
* Declaration of the variables used by LAUNCHER/400. Take a look  *
* at the user's guide to know how to use them.                    *
*****************************************************************
     DHANDLE           s             50    inz('*ONLY')
     DSVRADDR          s             30    inz('*DEV')
     DCCSID            s             10    inz('*JOB')
     DLNCCMD           s             10    inz(*blanks)
     DLNCPARM1         s            512    inz(*blanks)
     DLNCPARM2         s           1024    inz(*blanks)
     DLNCOPT           s              1    inz(*blanks)
     DLNCRESULT        s            512    inz(*blanks)

*****************************************************************
```

```
* Variables used by the program                                        *
************************************************************************
     DPREMIER          s               9    inz(*blanks)
     DDATECMD          s                d   datfmt(*EUR)
     DDATECMDS         s               10    inz(*blanks)
     DPRIXUP           s               8p 0 inz(0)
     DPRIXTP           s               8p 0 inz(0)
     DPRIXU            s               8    inz(*blanks)
     DPRIXT            s               8    inz(*blanks)
     DQTE              s               10    inz(*blanks)
     DRECORD           s               9

************************************************************************
* Program entry point, defining parameters                             *
************************************************************************
     C     *ENTRY        PLIST
     c     premier       PARM                    record
*
* Show the screen and the data corresponding to the parameters
*
     C     PREMIER       chain     SP_CUST                       90
*
* Connecting to LAUNCHER/400. The server is addressed by the
* variable SVRADDR. *DEV means that the server is the who emulates
* the screen (when using a 5250 emulation screen).
*
     c                   call      'LNCOPEN'
     c                   parm                    handle
     c                   parm                    svraddr
     c                   parm                    ccsid

*---------------------------------------------------------------------
* Edits a sales document that will be sent by e-mail
*---------------------------------------------------------------------
     c                   eval      lnccmd='WORDOPEN'
     c                   exsr      CALLLNC
     c                   eval      lnccmd='WMINIMIZE'
     c                   exsr      CALLLNC
* Creates a new document based on a template
     c                   eval      lnccmd='WNEWFILE'
     c                   eval      lncparm1=''+
     c                             '\Letter.dot'
     c                   exsr      CALLLNC
     c                   eval      lnccmd='WMINIMIZE'
```

```
     c                   exsr      CALLLNC
* Using bookmarks, data are set in the document
     c                   eval      lnccmd='WBOOKMARK'
     c                   eval      lncparm1='FNAME'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='LNAME'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='ADDRESS'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='ZIP'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='CITY'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='STATE'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='PHONE'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='FAX'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='EMAIL'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
* Small routine to show the orders of the customer.
* We go on the order's file
     c     cust_id       setll     sp_ord_id
     c     cust_id       reade     sp_ord_id                      91
     c   91              goto      Fin
* On the document, go to the table
     c                   eval      lnccmd='WBOOKMARK'
     c                   eval      lncparm1='TABLE'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c     loop_order    tag
     c                   move      ORDER_DATE    datecmd
     c                   movel     datecmd       datecmds
* Then, go to the order's detail
```

```
     c     order_id      setll     sp_det_or
     c     order_id      reade     sp_det_or                      92
     c   92              goto      Suite
     c     loop_detail   tag
     c                   eval      lnccmd='WSETLINE'
     c                   eval      lncparm1= + '%SEP%'
     c                             + PART_DESC
     c                   exsr      CALLLNC
     c                   eval      PRIXUP=
     c                   movel     prixup        prixu
     c                   eval      lnccmd='WBOOKMARK'
     c                   eval      lncparm1='''*RIGHT'' ''NUMFMT(0)''
 '
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   movel(p)  QUANTITY      QTE
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      PRIXTP=
     c                   movel     prixtp        prixt
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
* We read next detail from the order
     c     order_id      reade     sp_det_or                      92
     c  N92              eval      lncparm1='*RIGHT'
     c  N92              eval      lncparm2=
     c  N92              exsr      CALLLNC
     c  N92              goto      loop_detail
     c     Suite         tag
* And here, the next order
     c     cust_id       reade     sp_ord_id                      93
     c  N93              eval      lncparm1='*RIGHT'
     c  N93              eval      lncparm2=
     c  N93              exsr      CALLLNC
     c  N93              goto      loop_order
* All data are set, wa can save and send it
     c     Fin           tag
     c                   eval      lnccmd='WSAVEAS'
     c                   eval      lncparm1='%LNCDIR%\samples\' +
     c                             firstname + '.doc'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lnccmd='WSENDTO'
     c                   eval      lncparm1=
* The next line is not executed, preventing from abusive mail
```

```
*    c                   exsr      CALLLNC
     c                   eval      lnccmd='WMAXIMIZE'
     c                   exsr      CALLLNC
      * Exiting from the Launcher server
     c                   eval      lnccmd='END'
     c                   exsr      CALLLNC
     c                   call      'LNCCLOSE'
     c                   parm                    HANDLE
     c                   seton                                        lr

*----------------------------------------------------------------
* SUB CALLLNC : This sub-routine send the command to the LAUNCHER/400
*    server. The different parameters are already set by main
*    program. For more information, take a look at the user's manual
*----------------------------------------------------------------
     c     CALLLNC       BEGSR
     c                   call      'LNCCMD'
     c                   parm                    HANDLE
     c                   parm                    LNCCMD
     c                   parm                    LNCOPT
     c                   parm                    LNCPARM1
     c                   parm                    LNCPARM2
     c                   parm                    LNCRESULT
     c                   ENDSR
```

Excel example

Here, we are going to use the cell names to fill in the data for the document's header.

The table will be filled using the **XLSETLINE** command.

Then, a macro will try to draw a border for the table.

Excel program source (in RPG language)

```
*****************************************************************
* Files used :       SP_CUST --> Customers file                 *
*                  SP_ORD_ID --> Logical file on ORDERS file    *
*                  SP_DET_OR --> Logical file on DETAIL file    *
*****************************************************************
     FSP_CUST   if   e           k disk
     FSP_ORD_ID if   e           k disk
     FSP_DET_OR if   e           k disk
*****************************************************************
* Declaration of the variables used by LAUNCHER/400. Take a look   *
```

```
* at the user's guide to know how to use them.                      *
*********************************************************************
     DHANDLE           s              50    inz('*ONLY')
     DSVRADDR          s              30    inz('*DEV')
     DCCSID            s              10    inz('*JOB')
     DLNCCMD           s              10    inz(*blanks)
     DLNCPARM1         s             512    inz(*blanks)
     DLNCPARM2         s            1024    inz(*blanks)
     DLNCOPT           s               1    inz(*blanks)
     DLNCRESULT        s             512    inz(*blanks)
      ***************************************************************
*****
* Variables used by the program                                      *
*********************************************************************
     DPREMIER          s               9    inz(*blanks)
     DDATECMD          s                d   datfmt(*EUR)
     DDATECMDS         s              10    inz(*blanks)
     DPRIXUP           s               8p 0 inz(0)
     DPRIXTP           s               8p 0 inz(0)
     DPRIXU            s               8    inz(*blanks)
     DPRIXT            s               8    inz(*blanks)
     DQTE              s              10    inz(*blanks)
     DRECORD           s               9
*********************************************************************
* Program entry point, defining parameters                           *
*********************************************************************
     C     *ENTRY        PLIST
     c     premier       PARM                    record
*
* Show the screen and the data corresponding to the parameters
*
     C     PREMIER       chain     SP_CUST                         90
*
* Connecting to LAUNCHER/400. The server is addressed by the variable
*     SVRADDR. *DEV means that the server is the who emulates the
*     screen (when using a 5250 emulation screen).
*
     c                   call      'LNCOPEN'
     c                   parm                    handle
     c                   parm                    svraddr
     c                   parm                    ccsid
*--------------------------------------------------------------------
* Edits a sales document that will be sent by e-mail
*--------------------------------------------------------------------
* Creates a new document based on a template
```

```
     c                   eval      lnccmd='XLOPENFILE'
     c                   eval      lncparm1=''+
     c                             '\letter.xlt'
     c                   exsr      CALLLNC
     c                   eval      lnccmd='XLMINIMIZE'
     c                   exsr      CALLLNC
* Using cell names, data are set in the sheet
     c                   eval      lnccmd='XLSETTEXT'
     c                   eval      lncparm1='NAME'
     c                   eval      lncparm2= + lastname
     c                   exsr      CALLLNC
     c                   eval      lncparm1='ADDRESS'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='ZIP'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='CITY'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='STATE'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='PHONE'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='FAX'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lncparm1='EMAIL'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
* Small routine to show the orders of the customer.
* We go on the order's file
     c     cust_id       setll     sp_ord_id
     c     cust_id       reade     sp_ord_id
     91
     c   91              goto      Fin
* On the sheet, go to the cell which begin the table
     c                   eval      lnccmd='XLGOTOCELL'
     c                   eval      lncparm1='TABLE'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c     loop_order    tag
     c                   move      ORDER_DATE    datecmd
```

```
     c                   movel     datecmd       datecmds
* Then, go to the order's detail
     c     order_id      setll     sp_det_or
     c     order_id      reade     sp_det_or
     92
     c   92              goto      Suite
     c     loop_detail   tag
     c                   eval      lnccmd='XLSETLINE'
     c                   eval      PRIXUP=
     c                   movel     prixup        prixu
     c                   movel(p)  QUANTITY      QTE
     c                   eval      PRIXTP=
     c                   movel     prixtp        prixt
     c                   eval      lncparm1= + '%SEP%' +
     c                             PART_DESC + '%SEP%' + prixu +
     c                             '%SEP%' + QTE + '%SEP%' + PRIXT
     c                   exsr      CALLLNC
* We read next detail from the order
     c     order_id      reade     sp_det_or                          9
2
     c  N92              goto      loop_detail
     c     Suite         tag
* And here, the next order
     c     cust_id       reade     sp_ord_id                          9
3
     c  N93              goto      loop_order
* All data are set, we can save and send it
     c     Fin           tag
* Execute a macro to make a page setup
     c                   eval      lnccmd='XLEXEMACRO'
     c                   eval      lncparm1='PageSetup'
     c                   exsr      CALLLNC
     c                   eval      lnccmd='XLSAVEAS'
     c                   eval      lncparm1='%LNCDIR%\samples\'
     c                             + firstname + '.xls'
     c                   eval      lncparm2=
     c                   exsr      CALLLNC
     c                   eval      lnccmd='XLSENDTO'
     c                   eval      lncparm1=
* The next line is not executed, preventing from abusive mail
* c                   exsr      CALLLNC
     c                   eval      lnccmd='XLMAXIMIZE'
     c                   exsr      CALLLNC
     c                   seton                                      lr
* Exiting from the Launcher server
```

```
c                   eval      lnccmd='END'
c                   exsr      CALLLNC
c                   call      'LNCCLOSE'
c                   parm                    HANDLE
```

FAQ and miscellaneous

How to know the AS/400 processor group ?

To know the processor group when EASYCOM or LAUNCHER is not installed :

Open a Terminal session with QSECOFR profile , enter the command :

WRKLICINF

The following screen is displayed :

```
WORK WITH LICENSE INFORMATION S1234567
20/01/19 16:22:21
SYSTEM SERIAL NUMBER . . . . . . . . . . : 1234567
PROCESSOR GROUP . . . . . . . . . . . . : P05
  . . .
```

Some CL commands

Call a program	CALL
Change a variable	CHGVAR
Copyright CALL	COPYRIGHT
Create bounded CL program	CRTBNDCL
Create CL module	CRTCLMOD
Create CL program	CRTCLPGM
Convert CL 38 source	CVTCLSRC
Convert TCP/IP CL Source	CVTTCPCL
Convert date format	CVTDAT
Data	DATA
Declare CL variable	DCL
DO group beginning	DO
Declare a file	DCLF
Delete a module	DLTMOD
Delete a program	DLTPGM
Display a module	DSPMOD
Display a program	DSPPGM
Display programs references	DSPPGMREF
Else	ELSE
DO group end	ENDDO
Data end	ENDINP
Program end	ENDPGM
End of reception	ENDRCV
Goto	GOTO
If	IF
Intercept a message	MONMSG
Program	PGM
Qualifier definition	QUAL
Receive file screen or BD	RCVF
Receive a message	RCVMSG
Return	RETURN
Remove a message	RMVMSG
Retrieve a CL source	RTVCLSRC
Retrieve a Data Area	RTVDTAARA
Retrieve a Journal entry	RTVJRNE
Retrieve member description	RTVMBRD
Retrieve message	RTVMSG
Send screen file	SNDF

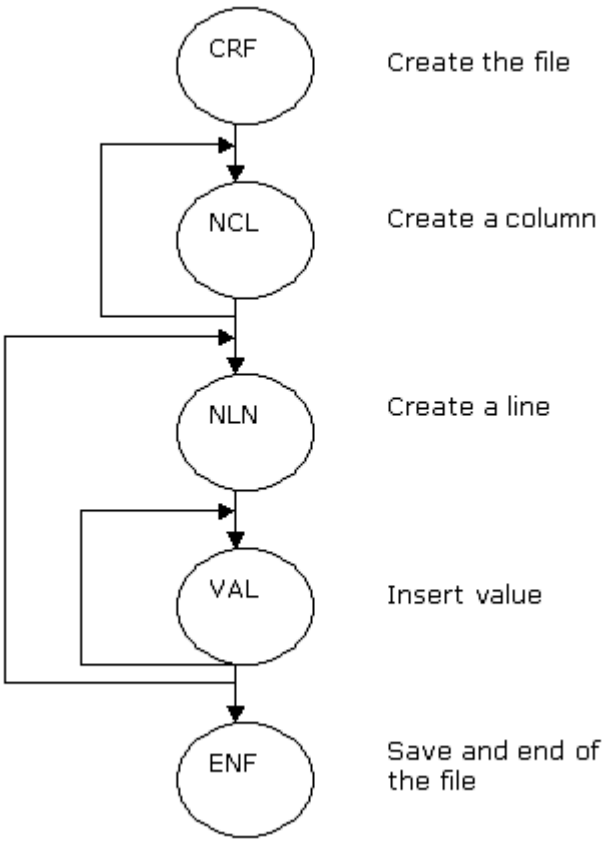
Send program message	SNDPGMMSG
Send / Receive screen file	SNDRCVF
Send a reply	SNDRPY
Send user message	SNDUSRMSG
Start SEU	STRSEU
Transfer control to program	TFRCTL
Wait	WAIT

How to create a merge file ?

The best way is to use the DBFXFER command which transfers the file from the AS/400 to the PC in one operation with different options that can be set using the DBXPROP command.

If you need to control or build the file you can use the following commands to create the file.

- CRF creates the file,
- NLN new line,
- NCL new column,
- VAL inserts the value,
- ENF ends of the creation, frees all resources.



The addition of values, in a given column, is done by the order VAL. The entry of the value will always be done in the current line. To create a new line condemns the seizure in the preceding line.

The safeguard of the data and the fence of the file are done by a call with order ENF.

It is also possible to carry out this operation into only once by carrying out a transfer of data by order DBFXFER. Thanks has this order, the data are exported to format CSV, directly exploitable by Excel.

Changing the PARM1 and PARM2 parameters size

See : Command PROPERTY

Send a spool file to Word

With LAUNCHER Ofiice, it is possible to send the contents of a spool file.

1° First create and compile a physical file with this description :

Only one field is necessary : 236 characters minimum.

2° Compile it into PCFILE for instance. You can use the following command to copy the spool file into the physical file :

CPYSPLF FILE(SPLFNAME) TOFILE(BIBLIO/PCFILE) MBROPT(*REPLACE)

3° The file can now be transferred to the PC and opened with Word.

Exemple

```
 PGM
/* DECLARATIONS */
DCL VAR(&HANDLE) TYPE(*CHAR) LEN(50)
DCL VAR(&CMD) TYPE(*CHAR) LEN(10)
DCL VAR(&OPT) TYPE(*CHAR) LEN(1)
DCL VAR(&PARM1) TYPE(*CHAR) LEN(512)
DCL VAR(&PARM2) TYPE(*CHAR) LEN(1024)
DCL VAR(&RESULT) TYPE(*CHAR) LEN(512)
/* OPEN LAUNCHER */
LNCOPEN
/* SPOULE FILE TRANSFERT */
CHGVAR VAR(&CMD) VALUE('DBFXFER')
CHGVAR VAR(&PARM1) VALUE('C:\TEMP\SPOULE.TXT')
CHGVAR VAR(&PARM2) VALUE('MAGALIE/PCFILER')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* OPENE WORD */
CHGVAR VAR(&CMD) VALUE('WORDOPEN')
CHGVAR VAR(&PARM1) VALUE(' ')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* WORD VISIBLE */
CHGVAR VAR(&CMD) VALUE('WORDSHOW')
CHGVAR VAR(&PARM1) VALUE(' ')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
```

```
&PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* OPEN FILE */
CHGVAR VAR(&CMD) VALUE('WOPENFILE')
CHGVAR VAR(&PARM1) VALUE('C:\TEMP\SPOULE.TXT')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
/* CLOSE WORD */
CHGVAR VAR(&CMD) VALUE('WORDCLOSE')
CHGVAR VAR(&PARM1) VALUE(' ')
CHGVAR VAR(&PARM2) VALUE(' ')
CALL PGM(LNCCMD) PARM(&HANDLE &CMD &OPT &PARM1 +
&PARM2 &RESULT)
MONMSG MSGID(LNC0000) EXEC(GOTO CMDLBL(ERROR))
LNCCLOSE
ENDPGM
```

Inserting text in page header or footer

The easiest way is to put a bookmark directly in the header or footer. Then, inserting text with WBOOKMARK command becomes possible.

SeekView displays property together with WdSeekCurrentPageHeader constant allows inserting in the header while WdSeekCurrentPageFooter constant allows inserting in the footer, then directly insert the text using WTYPETEXT or WCOPY.

Example :

```
PGM
LNCOPEN
LNCCMD CMD(WORDOPEN)
LNCCMD CMD(WOPENFILE) +
PARM1('%LNCDIR%\SAMPLES\MODELE_STYLE.DOC')
LNCCMD CMD(WSETPROP)PARM1(activewindow.activepane.view.seekview+
) PARM2(wdseekcurrentpageheader)
LNCCMD CMD(WTYPETEXT) PARM1('PAGE HEADER')
LNCCMD CMD(WSETPROP)PARM1(ACTIVEWINDOW.ACTIVEPANE.VIEW.SEEKVIEW+
) PARM2(WDSEEKCURRENTPAGEFOOTER)
LNCCMD CMD(WTYPETEXT) PARM1('PAGE FOOTER')
LNCCMD CMD(WORDSHOW)
LNCCLOSE
ENDPGM
```

Other Example :

```
PGM
LNCOPEN
LNCCMD CMD(WORDOPEN)
LNCCMD CMD(WOPENFILE) +
PARM1('%LNCDIR%\SAMPLES\MODELE_STYLE.DOC')
LNCCMD CMD(WBOOKMARK) PARM1(entetePAGE) PARM2('PAGE HEADER')
LNCCMD CMD(WBOOKMARK) PARM1(piedpage) PARM2('PAGE FOOTER')
LNCCMD CMD(WORDSHOW)
LNCCLOSE
ENDPGM
```

MAC compatibility with LAUNCHER Office

Today LAUNCHER Office can only run on Windows.

LAUNCHER Office a program in batch

It is possible if LAUNCHER Office is installed on the target PC.

Warning : Stand by functions must be desactivated.

See : LNCOPEN

Recovering IP address of PC

The following CL program call LNCOPEN to receive 5250 emulated PC IP address for the current job.

In input :

&HANDLE = '*GETDOT'
&SVRADDR = '*DEV'

In output :
&SVRADDR = PC IP address

```
PGM
/* VARIABLES USED TO OPEN COMMUNICATIN AND SEND COMMANDS ...*/
 DCL  VAR(&SVRADDR) TYPE(*CHAR) LEN(30)
 DCL  VAR(&HANDLE) TYPE(*CHAR) LEN(50)
 DCL  VAR(&CCSID) TYPE(*CHAR) LEN(10)

/* CALL LNCOPEN WITH &HANDLE=*GETDOT, &SVRADDR=*DEV */
 CHGVAR VAR(&SVRADDR) VALUE(*DEV)
 CHGVAR VAR(&HANDLE) VALUE(*GETDOT)
 CHGVAR VAR(&CCSID) VALUE(*JOB)
 CALL  PGM(LNCOPEN) PARM(&HANDLE &SVRADDR &CCSID)

/* ON RETURN, &SVRADDR=IP ADDRESS */
END:
 SNDPGMMSG MSGID(CPF9898) MSGF(QCPFMSG) +
     MSGDTA(&SVRADDR) MSGTYPE(*COMP)
ENDPGM
```

Service Mode error while using Word, Excel, Access

PROBLEM:

Word, Excel, Access and any others OLE applications can be used in 'service' mode, The messages 'Word can not be launched', or 'Excel can not be launched' are displayed.

Warning : The following solution applied only if LAUNCHER Office is in **NT service mode.**

If LAUNCHER Office service is configured to run with **a precise user profile**, this profile must have the rights to run Word, Excel and Access.

There are two ways to grant those rights :

1. Granting administration rights to the LAUNCHER Office machine.

This is the easiest way to do. It consists in adding the profile to the machine 'Administrators' group where LAUNCHER Office Service run.

Adding Excel, Word et Access applications execution COM rights.

Proceed with the following steps for each application (Word, Excel, and Access) :

Launch application (Word, Excel or Access), for the concerned user and open the macros manager (with ALT-F11).

Close the application as well as all Office applications.

Launch 'dcomcnfg' from 'Start/Execute' menu.

Select the application to be modified in the main tab.

With Word, select ' Microsoft Word Document '.

With Excel, select ' Microsoft Excel Application '.

With Access, select ' Microsoft Access Application '.

Click on Properties, and select 'Safety' tab.

Click on 'Using customised execution rights'.

Then click on the 'Modified' button according to this choice.

It clearly shows here that the launching rights are not granted to a 'normal' user.

Click on 'Add' and select a concerned user or group.

The whole operation may be done again for another application (Word, Excel Access, or else).

Printing a closed PDF document with word

1. Transformez le document Word en document PDF

2. Imprimez le document .PDF sans l'ouvrir sous Word

```
LNCCMD CMD(SHELL) PARM1('c:\temp\test.pdf')
                  PARM2('visible=false;wait=false;action=print')
```

See : Command SHELL

Cancelling an EXCEL sheet

Use XLDELSHEET command to cancel an Excel sheet. In this case, the system issues a confirmation window.

OK can be set as default answer.

A trick to avoid some Excel windows is to set "DisplayAlerts" property to "False".

Example:

Using LNCCMD with **XLSETPROP** command :

```
CHGVAR VAR(&CMD) VALUE(XLSETPROP)
CHGVAR VAR(&PARM1) VALUE('displayalerts')
CHGVAR VAR(PARM2) VALUE(false)
```

Sending faxes with LAUNCHER Office

If you have got a Fax system on your Windows network or station, then, you should be able to use it from your AS/400 with LAUNCHER Office.

The fax system can be a "Printer Fax driver" or a "Domino/Exchange fax connector".

1) Printer Fax driver

With such a system, to send a document by fax, you just have to print it, using a special printer driver configured on your PC.

Then the printer driver ask you for the phone number; Or the phone number is embedded in the document, hidden, with a special prefix (Manufacturer dependant).

For LAUNCHER Office, you must be able to embed the phone number in the document.

See your fax software documentation for this feature.

In the document template, the phone number must be inserted via a merge field, or via a bookmark.

Use LNCPRTDOC to merge the template, and the database record containing the phone number and all other variable fields.

Use LNCCMD command to print:

LNCCMD CMD(WPRINT) PARM1('Printer="Name of fax/printer driver"')

2) Domino or Exchange fax connector

With a fax connector on Domino or Exchange, to send a fax, you have to create a new message, attach documents.

The phone number is given as the destination recipient, with a special prefix

Example: FAX:555 555 555).

With LAUNCHER Office, you can send faxes like you send emails.

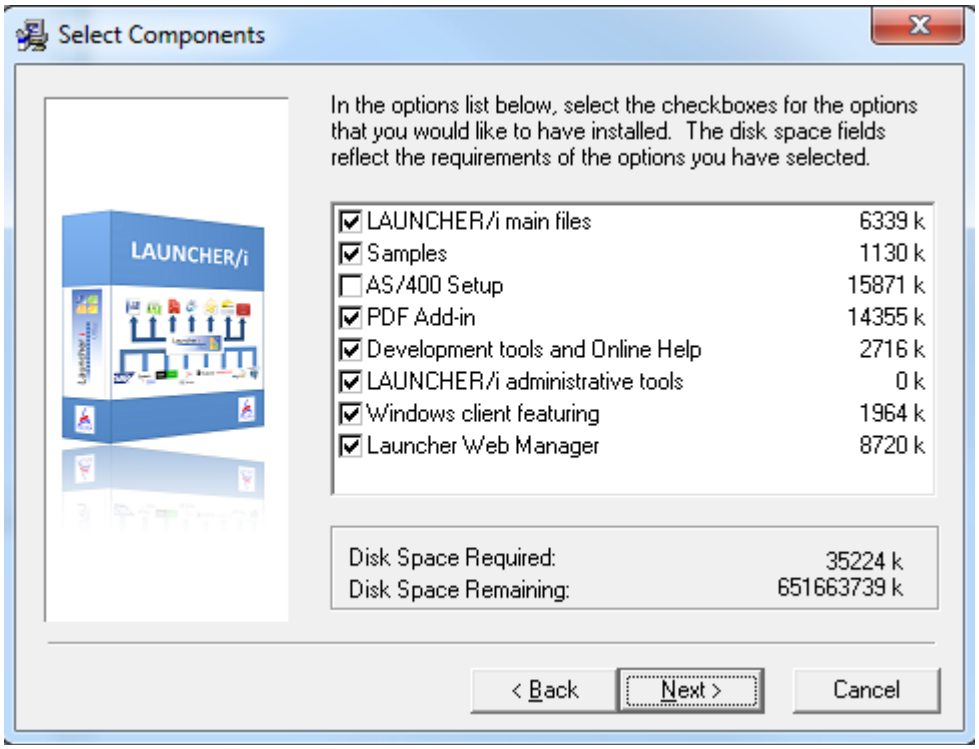
Use for example the command: LNCSNDMAIL.

Launcher Web Manager

LAUNCHER Web Manager est une application web permettant de piloter **LAUNCHER Office.** Grace à cette application vous pouvez offrir à vos utilisateurs un accès simplifier aux fonctionnalitées de **LAUNCHER Office.**

Installation

The installation of Launcher Web Manager is done by checking the corresponding option when installing LAUNCHER Office



Once the installation is completed and after rebooting the Windows PC, the application is accessible at the following URL: http://localhost:8080/launcherI/

Requirements

It is required to install Launcher Web Manager on the same Windows PC as the LAUNCHER Office application. Futhermore for the Launcher Web Mananger to work correctly, Launcher Office must be running.

The Launcer Web Manager uses the port 8080, this post must be free on the Windows PC. To allow remote accès to the application, you might need to configure your firewall to open the 8080 port.

First time login

By default only one user is available :

Username : admin

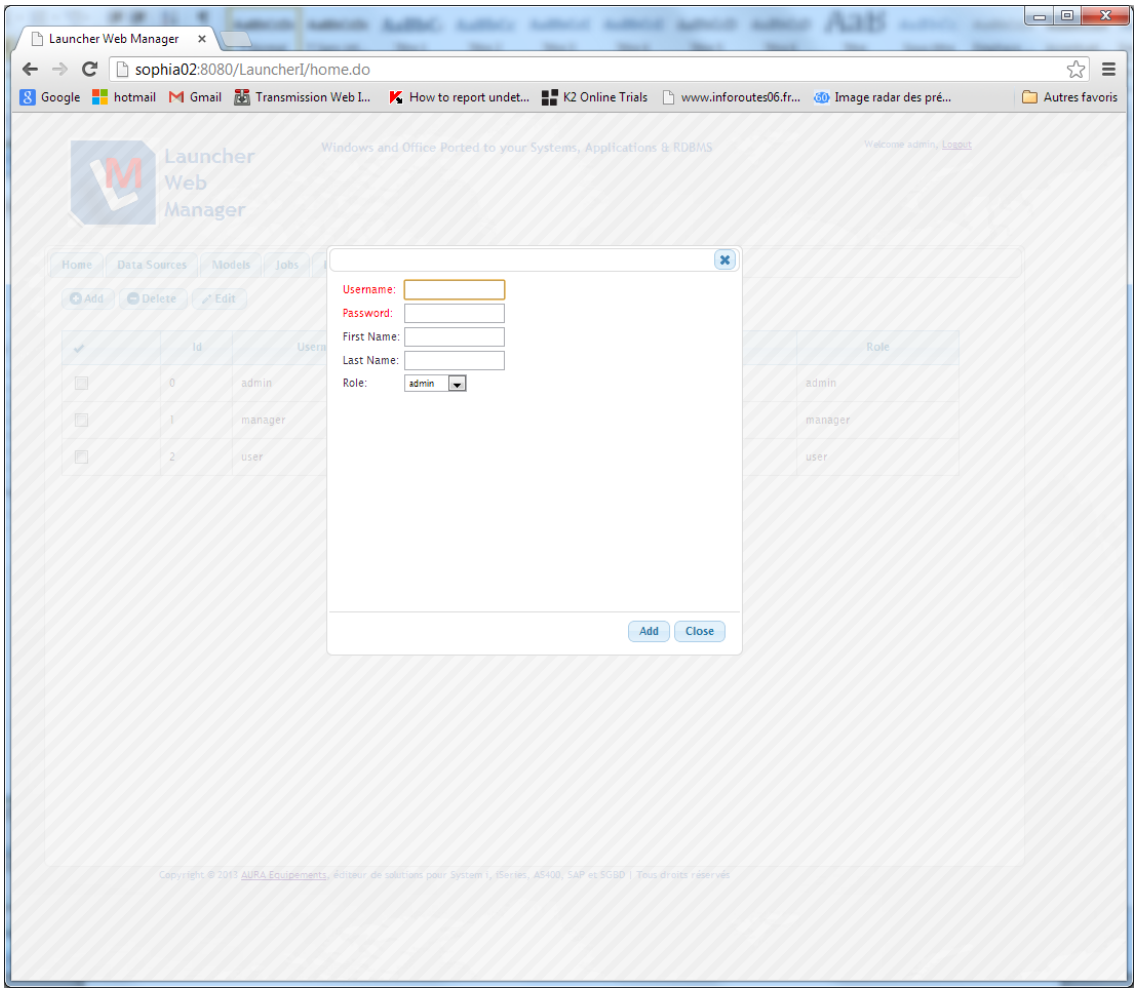
Password : admin

It is recommended that you change the admin password as soon as you log in fthe first time.

Users : managing users

It is possible for the admin user to add/delete/modify users. Each user has to have a username, password and a role. The username must be unique.

The first name, last name fields are optionnals.



Roles

There are three different roles in Launcher Web Manager, the roles are described bellow:

'User' : this is the lowest authorization level. The users with this role can only generate new documents and download them.

'Manager' : the 'manager' manager role includes the features of the 'user' role and allow users to define data sources and models which are used to generate documents.

Admin : the admin role includes all the features of the 'manager' role and adds the ability to manage users. Futhermore, uses with admin role are able to see and do actions on all the models, datasources and files in the application.

Data sources : data sources management

Data sources represent the mean to get data from several data base types. Those data will be then sent to LAUNCHER Office to generate documents. Depending on your LAUNCHER Office version, you will be able to create data sources from SAP, Oracle, MySQL, PostgreSQL, MS SQL Server, Sysbase, System i DB2 or a simple CSV file.

SAP data source

In order to create a SAP data source, the following fields are mandatory :

Name : the name which will be used for the data source in Launcher Web Manager

IP Address : ip address of the SAP server, it has to be an IPV4 address

Username : name of the user on the SAP system

SAP Instance : number of the SAP instance

SAP Client : number of the SAP client

SAP Gateway Server : ip address of the SAP gateway server, it can be the same as the SAP server ip address, it also has to be an IPV4 address

SAP Gateway Service : number of the SAP gateway service

It is possible to create two different type of SAP data sources, the BAPI one of the Table one.

Bapi :

BAPI data sources provide access to the data from an SAP BAPI execution. When you ar edefining such a data source, you need to provide the name of the BAPI you want to execute as well as the name of the result table. If the BAPI has parameters, they will be asked to the

user at document generation time

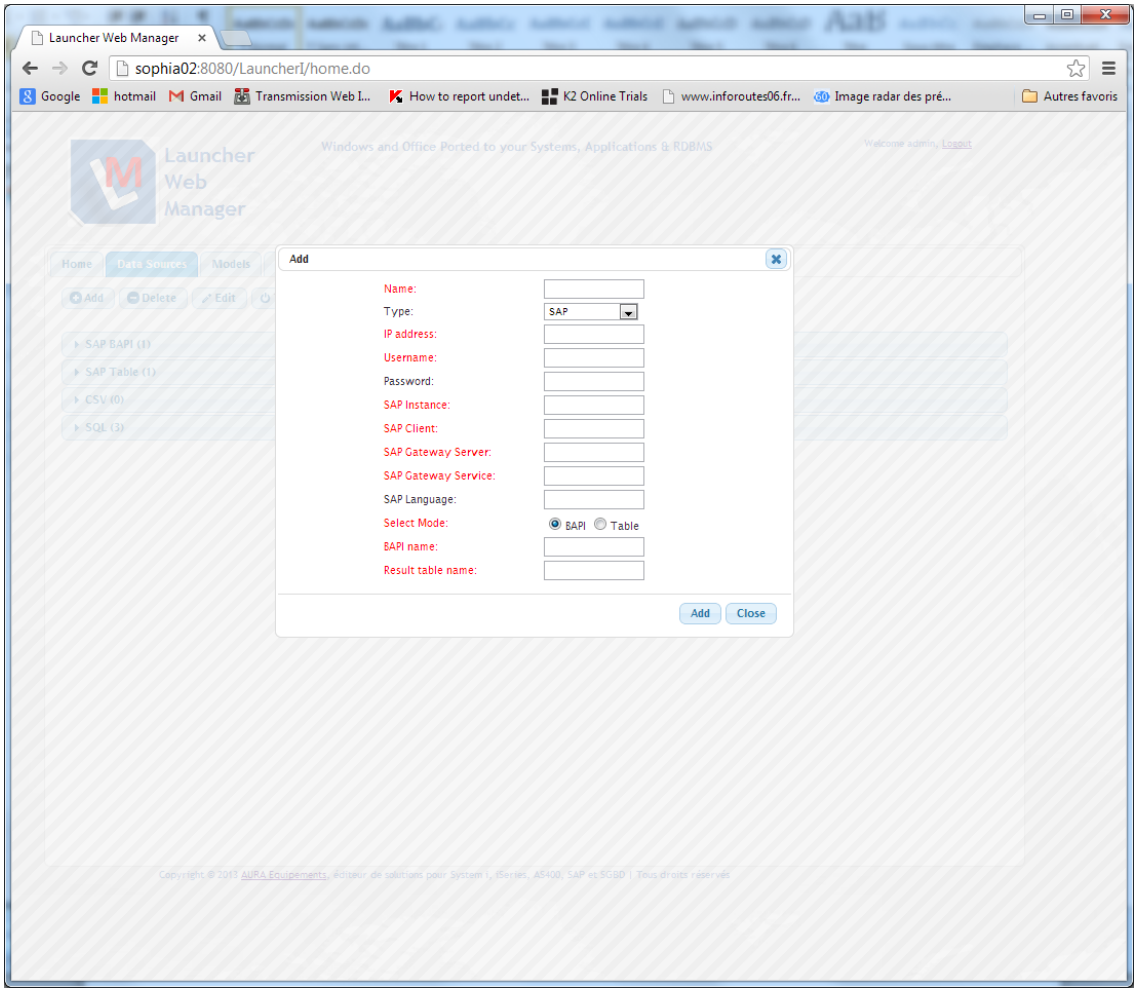
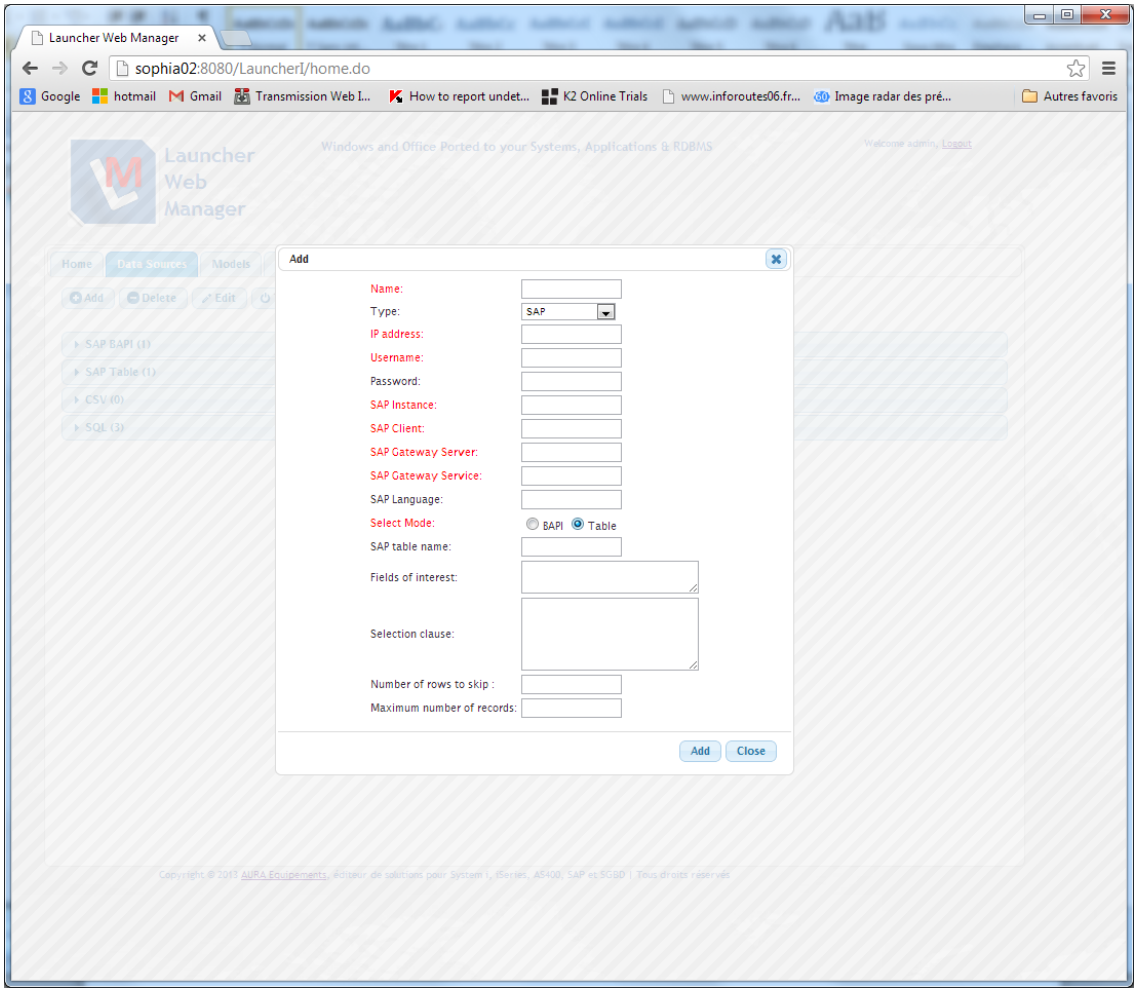


Table :

In the case of a SAP table data source, the data will be pulled directly from a SAP table. You have to specify the name of the SAP table, the fields of interest you wish to grab, the selection criteria, the number of rows you might want to skip, and the maximum number of rows you

want to get.



SQL like data source

SQL like data source means data source using the various following data bases, Oracle, MySQL, PostgreSQL, MS SQL Server, Sysbase and System i DB2. For this kind of data source, you have to fill the following fields :

Name : the name which will be used for the data source in Launcher Web Manager

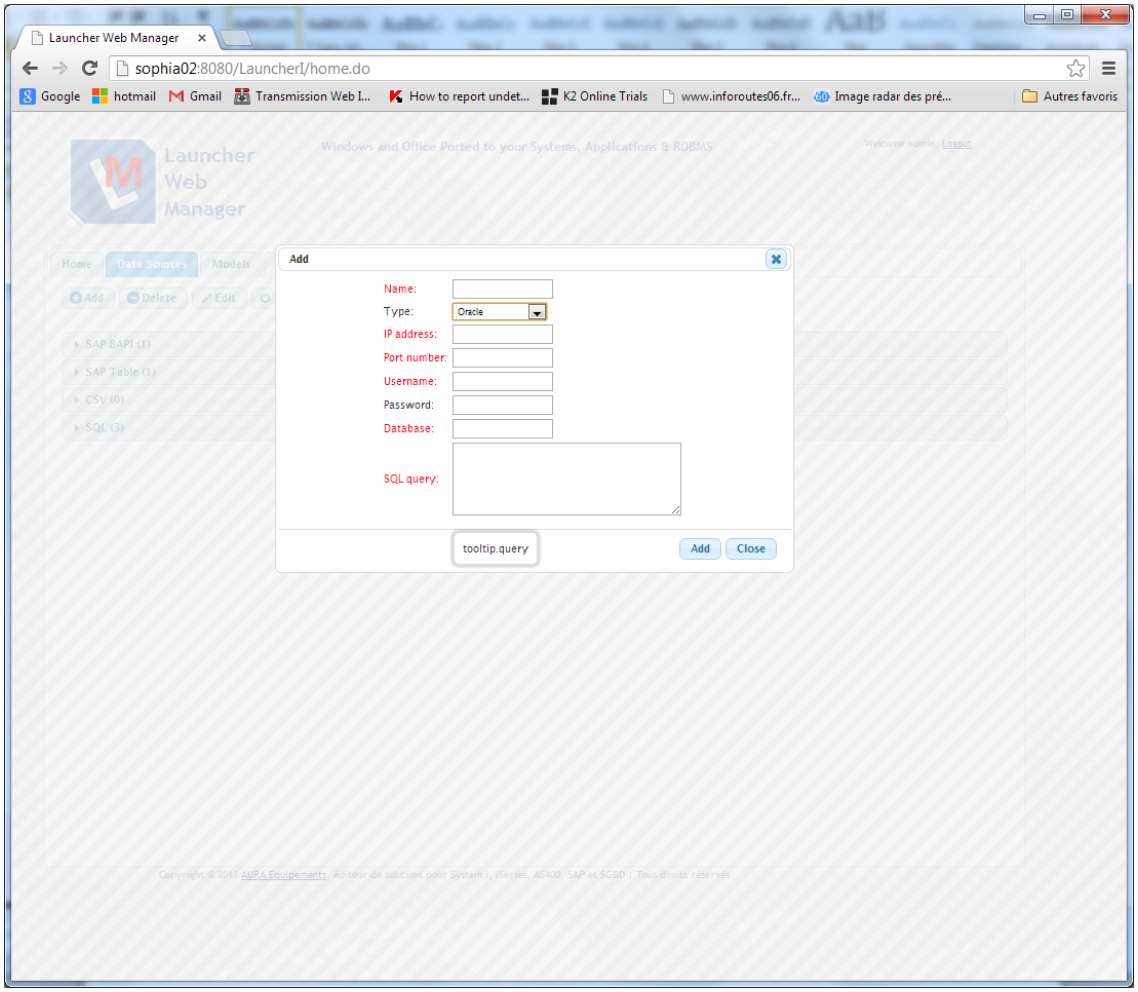
IP Address : ip address of the data base server, it has to be an IPV4 address

Port : port number of the data base

Username : name of the user allowed to access the database

Data base : name of the data base or the schema to access

SQL Query : the SQL query used to grab the data from the data base



CSV Data source

Finally, you can use a plain CSV file has a data source. In this cas, the following fields are mandatory :

Name : the name which will be used for the data source in Launcher Web Manager

Path : the full path of the csv file. The file has to be accessible from the Launcher Web Manager application

Models : managing documents models

Document models have to be either MS Word or MS Excel files, furthermore, they have to be associated with an existing data source.

The following fields are mandatory :

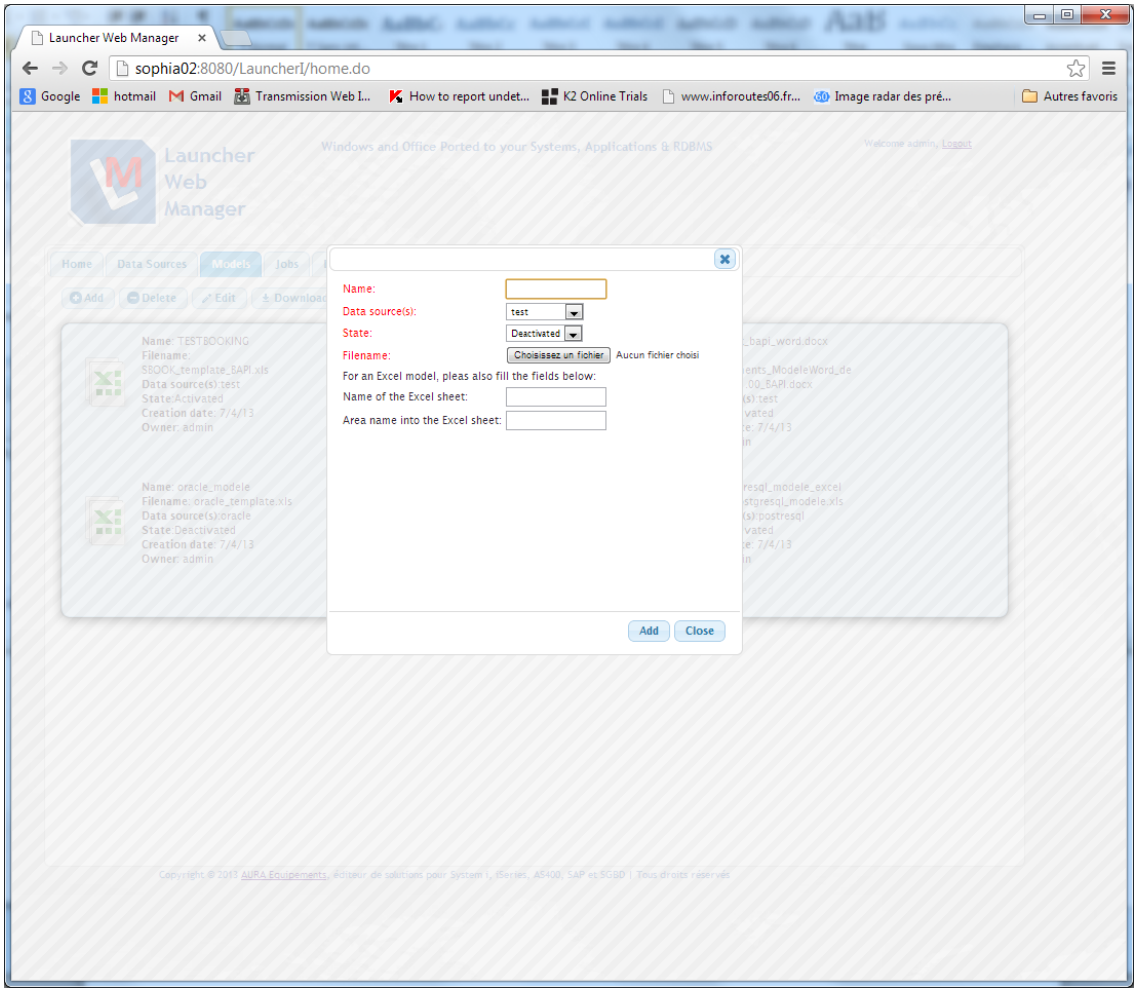
Name : the name of the model in Launcher Web Manager

Data source : the name of the data source to be associated with the model

File : the Ms Word or Excel file which will be used as model

State : You can specify activated or deactivated. If a model is deactivated, other users do not have access to it. This allows you to test your models before allowing all users to use it.

If the model is an MS Excel document, you also have to specify the sheet name and the name range to use.



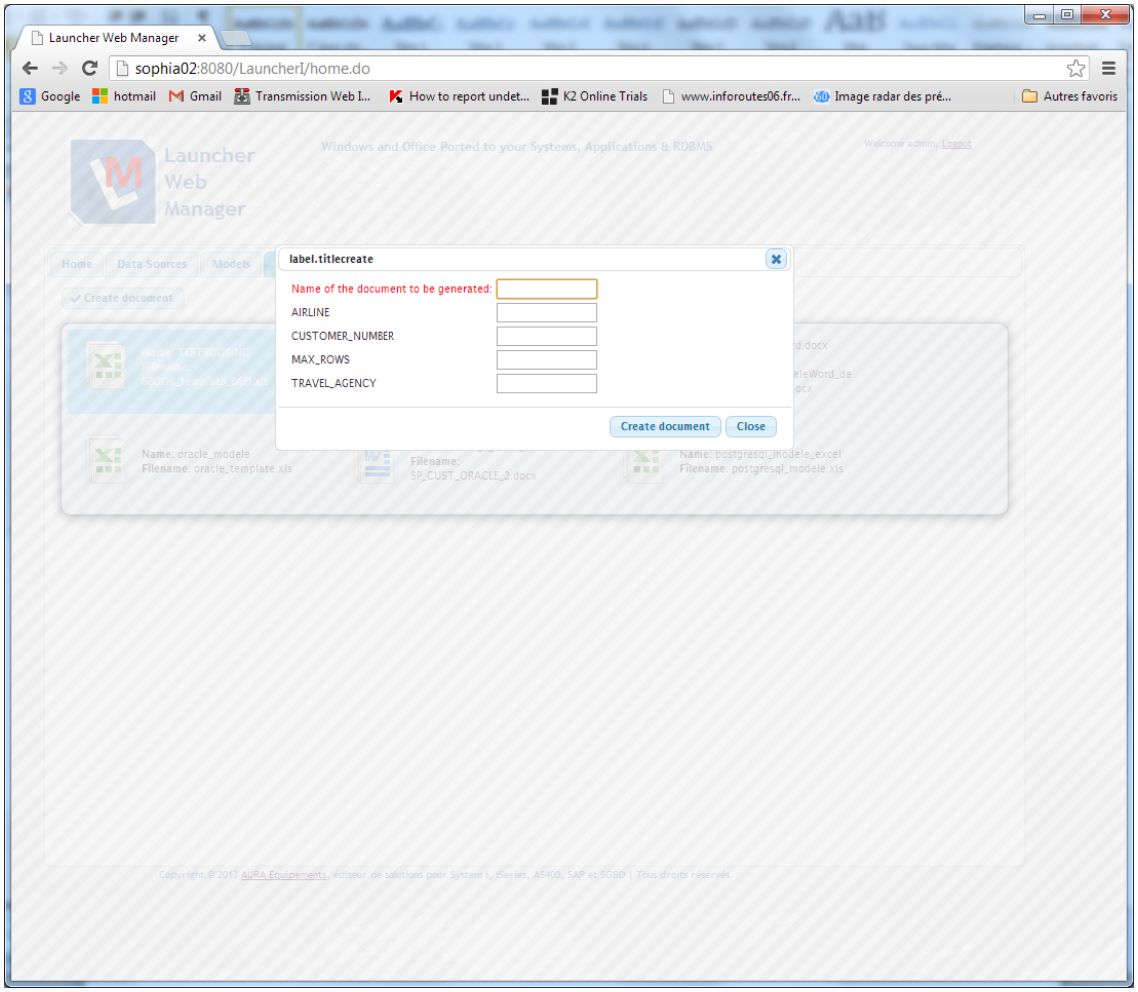
Jobs : document generation

The jobs tab let you generate documents from the models. This tab is accessible to all users. Users with a ‘user’ role can only see activated models. Users with a ‘manager’ role see activated models and disactivated models which belong to them.

To generate a new document, you have to select the model and then click on the generate button.

You have to specify a name for the final document. The name of the documents have to be unique. If the name is already used, the previous file will be replaced.

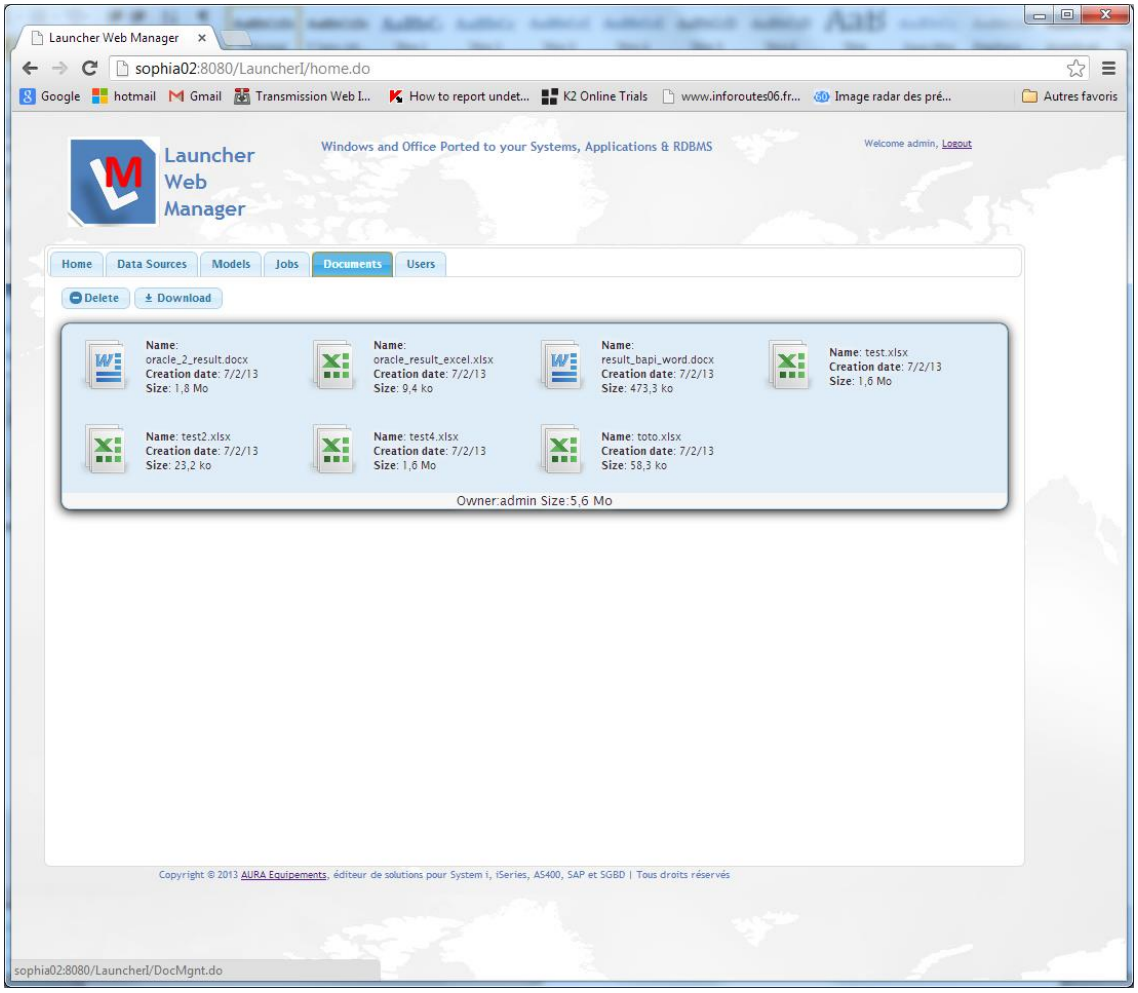
If the model is based on a SAP BAPI, the user will be then able to enter the BAPI parameters.



Documents : managing generated documents

The ‘Documents’ tab let you manage the generated documents, you can either download or delete them. Users having a ‘user’ or ‘manager’ role can only see their own documents. The ones having the ‘admin’ role can visualize, delete, or download, all the users’ documents.

You can download the files by either selecting them and then hitting the download button, or by double-clicking on the file’s icon.



Errors codes

Connection error 10061 (Hex 274D) : Connection Refused

Check the IP address or the name of the AS/400.

Check that the Job **LAUNCHERD** has been properly launched on the AS/400.

The command to start the job is : **STRLNCD**

The command to stop the job is : **ENDLNCD**

LAUNCHER Office uses port number **6078** by default.

By default, **LAUNCHERD** uses **port** number **5681**.

Connection error 11001 (Hex2AF9) : Host not found

The AS/400 is designated by its name on the TCP/IP network.

Check the name. The TCP/IP network could not find it either on the local system or on a DNS server.

Check the AS/400 name, the DNS servers, or use an IP address of the form xxx.xxx.xxx.xxx

Connection error 10060 (Hex 274C) : Connection Time out

The TCP/IP address contacted does not exist on the network.

Check the AS/400's TCP/IP address or its name.

If an AS/400 machine name is used, check that it is properly referenced on the DNS servers.

How to reach us ?

For support request, please create a ticket on the following website: http://support.easycom-aura.com

You need to register the first time, you will then receive your password.

Support requests sent to tech@easycom-aura.com are no longer be taken into account.

For general or commercial information: info@easycom-aura.com.

Copyright