Infor LN Public Interfaces Reference Guide (On-premises)

Release Enterprise Server 10.8

Publication Information

Release: Infor LN Enterprise Server 10.8

Publication date: November 7, 2025

Contents

About this guide

Intended audience

This guide is intended for IT professionals working in implementation projects or IT optimization phases for Infor LN. Basic knowledge about the Infor LN software structure and Infor LN's 4GL programming language is a pre-requisite.

Related documents

You can find these documents on docs.infor.com:

- *Infor LN Studio Application Development Guide*
- *Infor LN Studio Integration Development Guide*
- *Infor LN Extensions Development Guide*

You can find the *Infor ES Programmer's Guide* in KB2924522. The content of this guide is also available in the help pages of Infor LN Studio.

Contacting Infor

If you have questions about Infor products, go to Infor Concierge at https://concierge.infor.com and create a support incident.

For the latest documentation, go to Documentation Central at docs.infor.com. We recommend that you check this website periodically for updated documentation.

Chapter 1 Introduction

Infor LN is a standard ERP application with rich functionality. With its built-in flexibility with parameters, workflows, dynamic processes, it can be adjusted to serve the business processes in the industries Infor LN is designed for. To close the small gaps between the standard functionality and the specific business needs, Infor LN offers a variety of extensibility possibilities. The main goal of extensibility is to develop the last-mile functionality for your organization without changing the core standard software components and using only the public interfaces of the standard application. In this way, you can develop the extensions fully separated from the standard components and upgrading the standard software will therefore not result in additional efforts and costs for upgrading the customizations. Extensions are not influenced by the upgrade process.

Public Interfaces

Public Interfaces are methods with LN application functionality that can be called from extensions.

Public Interfaces explained

The development of extensions can be made easier if methods from the LN Application can be used. This can be done by using the so-called LN Public Interfaces. LN Public Interfaces are functions in the LN Application that are available for anyone who develops extensions on LN. The available LN Public Interfaces are visible in LN Studio and in the Extension Modeler. LN Public Interfaces are generic functions with a certain level of complexity that will likely be used by multiple customers. Simple read actions, for example, needs to be developed by customers or implementation partners themselves. Moreover, LN Public Interfaces will perform something what cannot easily be achieved with one or more standard sessions or processes in LN.

How to request a new Public Interface

Apply the following process if you need a new LN Public Interface:

1 Evaluate if the new LN Public Interface is generic and contains a certain level of complexity. Make sure the required feature cannot be achieved with personalizing a standard session or process.

2 Create an incident and clearly describe the required LN Public Interface. Use the Excel Sheet “Request Template for LN Public Interfaces & Process Extensions)” which is attached to KB2003722 in the Infor Customer Portal.

3 Infor Support will create a defect for this incident.

4 Infor Development will review the requested LN Public Interface and will either develop the new LN Public Interface or reject the request. If the LN Public Interface is developed it will be released via the regular delivery process.

5 After the new solution has been installed the customer can use the new LN Public Interface in the extension modeler and/or in LN Studio.

6 Infor Support will complete the incident.

Public Interface example

Find below an example of one of the LN Public Interfaces. This Public Interface converts an amount to another currency.

```
long Common.ConvertAmount(
            domain    tncmp         iFinancialCompany,
            domain    tcamnt        iSourceAmount,
            domain    tcccur        iSourceCurrency,
            domain    tcrtyp        iExchangeRateType,
            domain    tcdate        iRateDateUTC,
            domain    tcccur        iTargetCurrency,
    ref     domain    tcamnt        oTargetAmount,
    ref     domain    tcmcs.s999m   oExceptionMessage mb,
    ref               long          oExceptionID)
```

It is very important to use LN Public Interfaces properly and to catch the errors. For this reason, each Public Interface has the output arguments 'oExceptionMessage' and 'oExceptionID'.

If the Public Interface returns a value unequal to zero, then argument oExceptionMessage is filled for sure with the latest exception message and argument oExceptionID is a reference to an XML-object that contains some more information about the exception. This extra information can be retrieved by using the Public Interface 'Exception' in otcextextapi. It is also important to free up memory by calling Public Interface Exception.Delete(...).

If the Public Interface returns zero, arguments oExceptionMessage and oExceptionID could be filled due to information messages. So, in fact only the return value indicates if a public interface is successful or not.

The next example shows how to implement error handling when using LN Public Interfaces, in this case Common.ConvertAmount to convert an amount to another currency.

```
#pragma used dll "otcextextapi"
#pragma used dll "otcextemmapi"

                 long          exception.id, i
        domain   tcmcs.s999m   exception.message

if Common.ConvertAmount(..., exception.message, exception.id) <> 0 then
        |* Exception(s) found.
        for i = 1 to Exception.NumberOfMessages(exception.id)
                dal.set.error.message("@"& Exception.GetMessage(i))
        endfor
        Exception.Delete(exception.id)
        return(DALHOOKERROR)
else
        |* Call was successful. Exception messages could exist!
        Exception.Delete(exception.id)
endif
```

To be able to use the Public Interface in your extensions, you need to add a “#pragma used dll” statement for the DLL that contains the Public Interface. The DLL names can be found in the documentation of the available Public Interfaces. Note that the DLL name needs to be preceded by an “o” in the #pragma-statement.

Available Public Interfaces

The next chapters of this document describe the available Public Interfaces for Infor LN (Enterprise Server 10.8) and what their usage is.

If the Public Interfaces described in this document are not shown in the Extension Modeler or Infor LN Studio in your environment, it may be necessary to apply a Knowledge Base article (KB) that can be found in the Infor Customer Portal. The applicable KB number is mentioned in the Public Interface description.

Chapter 2 Public Interfaces for Exchange

Public Interfaces for ExchangeScheme

The following functions are available:

ExchangeScheme.NonRegularImport

DLL:	daextxchapi This function is available from KB2321682.
Syntax:	long ExchangeScheme.NonRegularImport(domain daxch.cxch iExchangeScheme, domain daxch.cbat iBatchFrom, domain daxch.cbat iBatchTo, domain daxch.pint iBatchSequenceNumberFrom, domain daxch.pint iBatchSequenceNumberTo, domain daxch.redo iProcessingType, domain daxch.yesno iReprocessRecordsRejectedDueToErrors, domain daxch.yesno iReprocessRecordsRejectedDueToConditions, domain daxch.yesno iOverruleBatchCompany, domain daxch.comp iBatchCompany, domain daxch.yesno iReallyQuitOnStopCondition, ref long oRunNumber, ref long oTryNumber, ref domain daxch.yesno oProcessStoppedDueToStopCondition, ref string oExceptionMessage(), ref long oExceptionId)
Usage:	Expl: This function does a non regular import of the given ExchangeScheme. Based on session "Import Data (on a Non Regular Basis)" (daxch0223m000) Pre: NA Post: Input: iExchangeScheme - Exchange Scheme code: Mandatory iBatchFrom - Batch From: iBatchTo - Batch To: Mandatory

```
                iBatchSequenceNumberFrom- Sequence Number From of the batch.
                iBatchSequenceNumberTo  - Sequence Number To of the batch.
                                          Mandatory
                iProcessingType         - The processing type of the import
                                          procedure: Mandatory
                        Allowed values
                                daxch.redo.new       1 (New run)
                                daxch.redo.restart   2 (Restart Previous run)
                                daxch.redo.reprocess 3 (Reprocess rejected records)
                                daxch.redo.continue  4 (Continue interrupted run)
                iReprocessRecordsRejectedDueToErrors
                                        daxch.yesno.yes = 1
                                        daxch.yesno.no  = 2
                                        - records that are rejected due to
                                          errors in the previous import run are
                                          processed again: Mandatory
                                        - This option has only influence when
                                          iProcessingType = daxch.redo.reprocess
                iReprocessRecordsRejectedDueToConditions
                                        daxch.yesno.yes = 1
                                        daxch.yesno.no  = 2
                                        - records that are rejected due to
                                          conditions in the previous run are
                                          processed again: Mandatory
                                        - This option has only influence when
                                          iProcessingType = daxch.redo.reprocess
                iOverruleBatchCompany
                                        daxch.yesno.yes = 1
                                        daxch.yesno.no  = 2
                                        - batch companies that are defined in
                                          the Batches (daxch0104m000) session
                                          are overruled: Mandatory
                iBatchCompany           - The LN company that is used instead
                                          of the batch company.
                                        - This option has only influence when
                                          iOverruleBatchCompany = daxch.yesno.yes
                iReallyQuitOnStopCondition
                                        daxch.yesno.yes = 1
                                        daxch.yesno.no  = 2
                                        - yes: an import program cannot continue
                                          if it is stopped.
                                        - no: the exchange process will not stop
                                          if a stop condition returns true
        Output:
                oRunNumber              - The run number of the exchange scheme
                                          that is imported or redone
                oTryNumber              - The Try number of the executed run.
                oProcessStoppedDueToStopCondition
                                        - Yes the Import was stopped due to a
                oExceptionMessage       - The last message if any message is
                                          found. If more than one message is
                                          given, these are present in the
                                          oExceptionID.
                oExceptionID            - An ID that refers to the exception
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
                                          stop condition
        Return: 0                       - The Exchange Scheme is imported
                                          or process stopped due to stop condition
                <> 0                    - Execution stopped. Exchange scheme,
                                          batch or sequence does not exist
```


ExchangeScheme.RegularImport

DLL:	daextxchapi This function is available from KB2321682.
Syntax:	long ExchangeScheme.RegularImport(domain daxch.cxch iExchangeScheme, domain daxch.redo iProcessingType, domain daxch.yesno iReprocessRecordsRejectedDueToErrors, domain daxch.yesno iReprocessRecordsRejectedDueToConditions, domain daxch.yesno iOverruleBatchCompany, domain daxch.comp iBatchCompany, domain daxch.yesno iReallyQuitOnStopCondition, domain daxch.yesno iIgnoreBatchesToImport, domain daxch.yesno iUncompressASCIIFiles, ref long oRunNumber, ref long oTryNumber, ref domain daxch.yesno oProcessStoppedDueToStopCondition, ref string oExceptionMessage(), ref long oExceptionId)
Usage:	Expl: This function does a regular import of the given ExchangeScheme. Based on session "Import Data (on a Regular Basis)" (daxch0224m000) Pre: NA Post: Input: iExchangeScheme - Exchange Scheme code: Mandatory iProcessingType - The processing type of the import procedure: Mandatory Allowed values daxch.redo.new 1 (New run) daxch.redo.restart 2 (Restart Previous run) daxch.redo.reprocess 3 (Reprocess rejected records) daxch.redo.continue 4 (Continue interrupted run) iReprocessRecordsRejectedDueToErrors daxch.yesno.yes = 1 daxch.yesno.no = 2 - records that are rejected due to errors in the previous import run are processed again: Mandatory - This option has only influence when iProcessingType = daxch.redo.reprocess iReprocessRecordsRejectedDueToConditions daxch.yesno.yes = 1 daxch.yesno.no = 2 - records that are rejected due to conditions in the previous run are processed again: Mandatory - This option has only influence when iProcessingType = daxch.redo.reprocess iOverruleBatchCompany daxch.yesno.yes = 1 daxch.yesno.no = 2 - batch companies that are defined in

```
                                        the Batches (daxch0104m000) session
                                        are overruled: Mandatory
                iBatchCompany           - The LN company that is used instead
                                        of the batch company.
                                        - This option has only influence when
                                        iOverruleBatchCompany = daxch.yesno.yes
                iReallyQuitOnStopCondition
                                        daxch.yesno.yes = 1
                                        daxch.yesno.no  = 2
                                        - yes: an import program cannot continue
                                        if it is stopped.
                                        - no: the exchange process will not stop
                                        if a stop condition returns true
                iIgnoreBatchesToImport
                                        daxch.yesno.yes = 1
                                        daxch.yesno.no  = 2
                                        - yes: the batches in the exchange scheme
                                        are ignored during the import procedure.
                                        - no: the batches in the exchange scheme
                                        are not ignored
                iUncompressASCIIFiles
                                        daxch.yesno.yes = 1
                                        daxch.yesno.no  = 2
                                        - yes: compressed ASCII files are restored
                                        to their original format before the
                                        import procedure
                                        - no: ASCII file may not be compressed
        Output:
                oRunNumber              - The run number of the exchange scheme
                                        that is imported or redone
                oTryNumber              - The Try number of the executed run.
                oProcessStoppedDueToStopCondition
                                        - Yes the Import was stopped due to a
                                        stop condition
                oExceptionMessage       - The last message if any message is
                                        found. If more than one message is
                                        given, these are present in the
                                        oExceptionID.
                oExceptionID            - An ID that refers to the exception
                                        information. Use the functions in
                                        Exception to get all relevant
                                        information.
        Return: 0                       - The Exchange Scheme is imported
                                        or process stopped due to stop condition
                <> 0                    - Execution stopped. Exchange scheme,
                                        batch or sequence does not exist
```

Public Interfaces for ExchangeLogBatchLineLevel

The following functions are available:

ExchangeLogBatchLineLevel.StartOverview

ExchangeLogBatchLineLevel.StartOverview

DLL: daextxchapi

This function is available from KB2321682.

Syntax:

```
long ExchangeLogBatchLineLevel.StartOverview(
                        long            iStartMode,
        const           string          iStartFilter(),
                        long            iSessionIndex,
        const           string          iQueryExtend(),
                domain  daxch.txch      iTypeOfExchange,
        const           string          iExchangeScheme(),
                        long            iRunNumber,
                        long            iTryNumber,
                domain  daxch.cbat      iBatch,
        ref             string          oExceptionMessage(),
        ref             long            oExceptionID )
```

Usage:

```
        Expl:   This function starts session Log Table (Batch Line Level)
                (daxch0509m000).
        Input:  iStartMode
                        Specifies the start mode for the session.
                        Possible values are:
                        MODAL -         The parent session is blocked until the
                                        child session exits, the session will be
                                        started as a zoom session.
                        MODELESS -      Parent and child are parallel
                                        sessions that can be manipulated
                                        simultaneously.
                iStartFilter
                        Not Used
                iSessionIndex
                        Not Used
                iQueryExtend
                        Not Used
                iTypeOfExchange
                        Mandatory
                iExchangeScheme
                        Mandatory
                iRunNumber      The Run Number
                                If value = 0 then the last runnumber is used
                iTryNumber      The Try Number
                                If value = 0 then the last try number is used
                iBatch          The Batch
        Output: oExceptionMessage       - The last message if the return
                                          value is not equal to 0.
                                          If more than one  message is
                                          given, these are present in the
                                          oExceptionID
                oExceptionID            - An ID that refers to all error
                                          information. Use the functions in
                                          Exception to get all relevant
                                          information.
        Return: 0                       - Session started
                <> 0                    - Otherwise.
```


Chapter 3 Public Interfaces for Job Management

Public Interfaces for Job

The following functions are available:

Job.Activate

Job.Activate

DLL:	ttextaadapi This function is available from KB3634024.

Syntax:	`long Job.Activate(`
	`  const  domain  ttaad.cjob  iJob,`
	`  ref  string  oExceptionMessage() mb,`
	`  ref  long  oExceptionID )`

Usage:	`Expl:  This Public Interface activates a Job. The following rules apply for the Job:`
	`- Current user must have authorization for the Job.`
	`- Job must have been defined with 'Use External Schedule'.`
	`- Job must contain at least one active session.`
	`- Job status must be 'Free'; this implies that it can be activated only once, until the execution is ready.`
	`The Job will be queued and picked up by the Job Scheduler as soon as possible. Note that if the Job is not defined as 'Periodical', it can be activated only once, because it will be deleted after the execution.`
	`Pre:  db.retry.point() must have been set.`
	`Post:  abort.transaction() or commit.transaction() must be done.`
	`Input:  iJob  - The Job which must be activated. Mandatory.`
	`Output:  oExceptionMessage  - A message if the return value is not equal to 0. This message contains the root cause of the of the method failure.`
	`oExceptionID  - An ID that refers to all error information. Use the functions in Exception to get the error messages.`
	`Return values:`
	`0  - Function is executed successfully`
	<> 0 - Error(s) occurred`

Public Interfaces for JobSession

The following functions are available:

JobSession.ChangeInputValue

DLL: ttextaadapi

This function is available from KB3522710.

Syntax:

```
long JobSession.ChangeInputValue(
        const   domain  ttaad.cjob      iJob,
        const   domain  ttaad.sequ      iSessionNumber,
        const           string          iFieldName(),
        const           string          iNewValue(),
        const           boolean         iValidateField,
        ref             string          oExceptionMessage() mb,
        ref             long            oExceptionID )
```

Usage:

```
        Expl:   This Public Interface changes one input value for a session in a job. To
                change multiple values, the function must be called multiple times.
                All types of variables are supported, including dates; however, dates
                must be absolute dates. If dates are stored as a relative date, the
                change will be ignored, or an error will be set (in case iValidateField
                is 'true').
                The new value must be cast to a string, so for non-string values, use the
                str$() function.
                Array fields can be handled as well. For string fields, a suffix
                (1,<element>) for iFieldName is needed, for example "form.ccur(1,2)" for
                the second element in the array. For non-string fields the suffix must be
                (<element>), e.g. "form.qty(2)". Ensure there are no blanks and leading
                zeros in the suffix.
        Pre:    db.retry.point() must have been set.
        Post:   abort.transaction() or commit.transaction() must be done.
        Input:  iJob                    - The Job for which the input value must be
                                          changed. Mandatory.
                iSessionNumber          - The Session Number (not the Sequence Number!)
                                          in the job for which the input value must
                                          be changed. Mandatory.
                iFieldName              - The Field Name in the session. Mandatory.
                iNewValue               - New value for the form field in the job
                                          session. Optional (empty value will clear the
                                          form field).
                iValidateField          - Indicator whether a non-existing field or a
                                          relative date must result in an error.
                                          Otherwise they will be ignored. Mandatory
                                          (true/false).
        Output: oExceptionMessage       - A message if the return value is not equal
                                          to 0. This message contains the root cause of
                                          the of the method failure.
                oExceptionID            - An ID that refers to all error information. Use
                                          the functions in Exception to get the error
```

```
                                        messages.
    Return values:
            0                       - Function is executed successfully
            <> 0                    - Error(s) occurred
```

Chapter 4 Public Interfaces for User Management

Public Interfaces for User

The following functions are available:

User.ConvertToRuntime

User.ConvertToRuntime

DLL:	ttextaadapi This function is available from KB3601093.
Syntax:	long User.ConvertToRuntime(const domain ttaad.user iUser, const boolean iUserData, const boolean iRemoteUserData, const boolean iDevicePreferences, const boolean iDevelopmentParameters, const boolean iDeveloperAuthorizations, ref string oExceptionMessage() mb, ref long oExceptionID)
Usage:	Expl: This Public Interface converts User Data to runtime. The functionality is the same as the Convert action in session User Data (ttaad2500m000). Transaction management is handled internally in this Public Interface. Pre: No transaction active. User must be able to switch to company 0. Post: None. Input: iUser - The User which must converted to runtime. Mandatory and must exist in the database. iUserData - If this option is selected, the user data, terminal authorizations and text group authorizations are converted to run time. Mandatory (true/false). iRemoteUserData - If this option is selected, the remote user data are converted to run time. Mandatory (true/false). Is ignored in Cloud Edition. iDevicePreferences - If this option is selected, the device preferences are converted to run time. Mandatory (true/false).

```
                iDevelopmentParameters   - If this option is selected, the development
                                           parameters are converted to run time.
                                           Mandatory (true/false).
                iDeveloperAuthorizations - If this option is selected, the developer
                                           authorizations are converted to run time.
                                           Mandatory (true/false).
        Output: oExceptionMessage        - A message if the return value is not equal
                                           to 0. This message contains the root cause of
                                           the of the method failure.
                oExceptionID             - An ID that refers to all error information. Use
                                           the functions in Exception to get the error
                                           messages.
        Return values:
                0                        - Function is executed successfully
                <> 0                     - Error(s) occurred
```

Chapter 5 Public Interfaces for Document Output Management

Public Interfaces for Documents

The following functions are available:

Documents.StartOverview

Documents.StartOverview

DLL:	ttextrpiapi This function is available from KB3636431.
Syntax:	long Documents.StartOverview(const long iStartMode, const string iStartFilter(), const long iSessionIndex, const string iQueryExtend(), const domain ttutc iBatchStartDate, const domain ttlong10 iBatchID, const domain ttlong10 iDocument, const domain ttrpi.bsta iDocumentStatus, const domain ttrpi.doct iDocumentType, const domain ttdesc40 iDocumentKeyWord1 mb, const domain ttdesc40 iDocumentKeyWord2 mb, const domain ttdesc40 iDocumentKeyWord3 mb, const domain ttdesc40 iDocumentKeyWord4 mb, ref domain ttutc oBatchStartDate, ref domain ttlong10 oBatchID, ref domain ttlong10 oDocument, ref domain ttdesc500 oExceptionMessage mb, ref long oExceptionID)
Usage:	Expl: This Public Interface starts the session "Documents (ttrpi3510m000)" with the given input parameters. Pre: - Post: - Input: iStartMode Specifies the start mode for the session.

```
                        Possible values are:
                        MODAL -         The parent session is blocked until the
                                        child session exits, the session will be
                                        started as a zoom session.
                        MODELESS -      Parent and child are parallel
                                        sessions that can be manipulated
                                        simultaneously.
                iStartFilter
                        Not used.
                iSessionIndex
                        Specifies the session index that is to be used.
                        If specific session index is to be used, below Indices
                        are only applicable:
                        iStartMode equals MODELESS:
                                        Index 1 - Documents by Batch (default)
                                        Index 2 - Documents by Status
                                        Index 4 - Documents by Document Type, Keyword 1
                                        Index 5 - Documents by Document Type, Keyword 2
                                        Index 6 - Documents by Document Type, Keyword 3
                                        Index 7 - Documents by Document Type, Keyword 4
                                        Index 8 - Documents by Document Type
                        iStartMode equals MODAL:
                                        Index 1 - Documents by Batch
                iQueryExtend
                        Optional
                        A specific query to be used when zooming to this session.
                iBatchStartDate         - The DOM batch start date.
                iBatchID                - The DOM batch ID.
                iDocument               - The DOM document number
                iDocumentStatus         - The DOM document status
                iDocumentType           - The DOM document type
                iDocumentKeyWord1       - The DOM Keyword 1
                iDocumentKeyWord2       - The DOM Keyword 2
                iDocumentKeyWord3       - The DOM Keyword 3
                iDocumentKeyWord4       - The DOM Keyword 4
        Output:

                For iStartMode MODAL:
                oBatchStartDate         - The DOM batch start date.
                oBatchID                - The DOM batch ID
                oDocument               - The DOM document number

                For exceptions:
                oExceptionMessage       - A message if the return value is not equal
                                          to 0. This message contains the root cause of
                                          the method failure.
                oExceptionID            - An ID that refers to all error information. Use
                                          the functions in Exception to get the error
                                          messages.
        Return values:
                0                       - Function is executed successfully
                <> 0                    - Error(s) occurred
```

Chapter 6 Public Interfaces for Application Development

Public Interfaces for AdditionalFile

The following functions are available:

AdditionalFile.Upload

AdditionalFile.Upload

DLL:	ttextadvapi This function is available from KB3641415.
Syntax:	long AdditionalFile.Upload(domain ttscm.sofc iAdditionalFile mb, domain ttdesc60 iDescription mb, boolean iEditable, domain ttst255m iFile mb, domain ttst255m iRevisionText mb, ref string oExceptionMessage() mb, ref long oExceptionID)
Usage:	Expl: This Public Interface uploads an additional file. The following rules apply for the upload action. - The upload action can only handle additional files in package tx - The current user should have development authorization for package tx - The current user should have a current Package VRC for development in package tx - The additional file argument (iAdditionalFile) consists of package, module and additional file name, where package and module should already exist - The additional file will be created or updated in the package VRC of package tx of the current package combination of the user - If SCM is on, the additional file will be checked out and checked in. Revision text will be used as checkin text. Pre: db.retry.point() must have been set. Post: abort.transaction() or commit.transaction() must be done. Input: iAdditionalFile - The additional file to be uploaded. Including package, module, additional file and extension. Mandatory

```
                iDescription    - The description of the additional file
                iEditable       - Indicates if this additional file is editable
                                  (true/false)
                iFile           - Complete path of the file to be uploaded. Mandatory
                iRevisionText   - Revision text to be used as checkin text
        Output: oExceptionMessage       - A message if the return value is not equal
                                          to 0. This message contains the root cause of
                                          the method failure.
                oExceptionID            - An ID that refers to all error information. Use
                                          the functions in Exception to get the error
                                          messages.
        Return values:
                0               - Function is executed successfully
                <> 0            - Error(s) occurred
```