

32 SOFT
DATA LOADERS
Productivity Solutions for QAD ERP

XLD Service Installation Guide



XLD Service Installation Guide

Thank you for your interest in 32 Soft's productivity solutions for QAD. You are on the path to improved data management and productivity.

This Installation Guide will show you how to load the XLD Service that allows our Data Loaders to communicate safely and effectively with your QAD ERP system. The installation is quick and easy. It should take you only about a half hour.


If you have previously trialed or adopted our Data Loader technology, you have already installed the listener and can skip to page 11.


Table of Contents


Requirements for XLD Service	3
Installation Instructions	3
Create a directory/folder	3
Transfer files to the server	4
Set Access Rights	5
Edit configuration files & scripts	6
Compile the programs	10
First Installation	10
If XLD already installed	11
Start the XLD Service	12
Set Up the Data Loader(s)	13
Test the Data Loader Template	13


DATA LOADERS

We have dozens of
DATA LOADERS
for every major QAD process

 Master Data & Inventory

 Distribution & Supply Chain

 Finance

 Production

To browse a complete list of Data Loaders, [click here](#).

XLD Service Installation Guide

Requirements for the XLD service

- The XLD service needs to be installed on the server where the QAD code resides.
- The XLD service is a Java process and it needs at least version 1.5 of Java. Either of the JRE (Java Runtime Environment) or JDK (Java Development Kit) versions should work fine. Linux systems usually already have latest Java installed.

Notes about Java:

- To find the Java version you have installed, run “java -version”.
- Run “whereis java” to find where the Java binary is installed.
 - On Linux, Java is usually linked under /usr/bin
 - /usr/bin/java is linked to /etc/alternatives/java
- If you run a listing “ls -l /etc/alternatives/java”, you can confirm where java is installed.
- If needed you can download and install latest Java from the official site www.java.com

Installation Instructions

Create a directory/folder for the XLD service on the QAD server

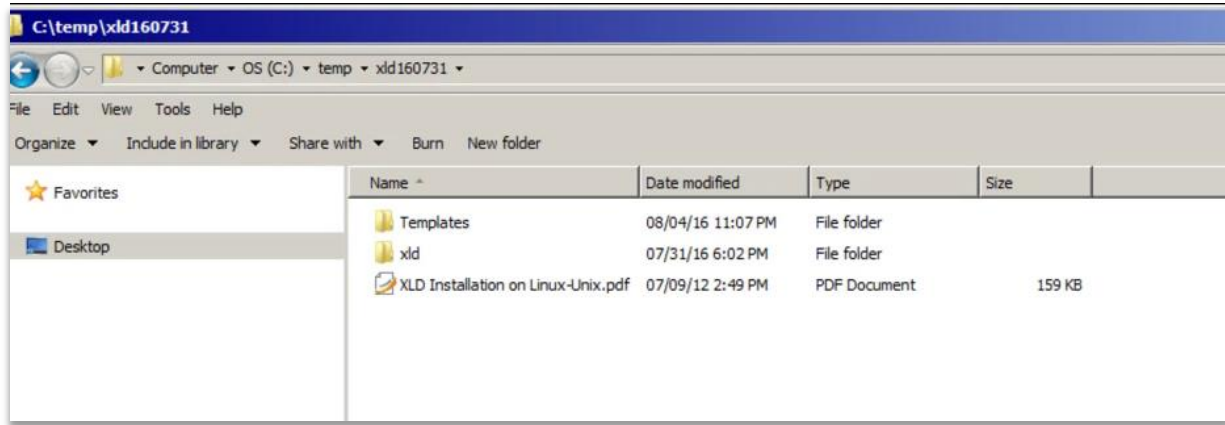
The 32soft XLD can be installed in any folder on the QAD code server. If you have Test and Production databases on same server, we recommend that you create two separate folders. This way you can apply and test updates in the Test environment without affecting Production.

1. Create a folder called *32soft* and under it create a folder called *xld*. If needed, you can have multiple XLD folders for different environments. For example, if you have Development, Test and Production environments on the same server, you could have an *xld.test* folder for both DEV and TEST connections and a separate *xld.prod* folder for Production.

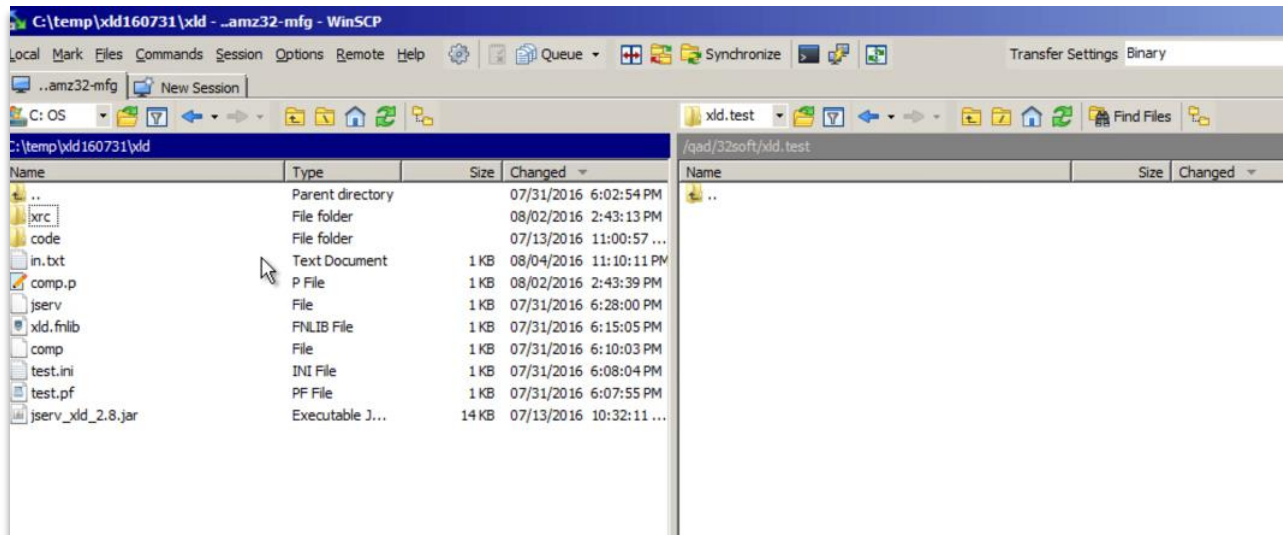
Note: The 32soft xld folder can be owned by any account, not necessarily root. It can be owned, for example, by the QAD mfg account or by a new account called xld. The owner of the folder should be used for starting the XLD service.

Transfer the XLD files to the server

1. Once the xld folder is created, transfer the xld files received from 32 Soft to your server. The 32soft xld files are sent via email and are packaged as a zip file. When you unzip the file, you will see the xld folder with the XLD service files and the Templates folder with the Data Loader(s).



2. Transfer the xld folder to the QAD server in binary mode. For example, using WinSCP, set the Transfer Mode to Binary and transfer the entire xld folder content to the new /apps/32soft/xld.test folder.



Set the access rights for the XLD files

1. Connect to the server as the owner of the 32 Soft xld folder.
2. Go to the xld folder and run the following commands:

```
chmod 644 *.*
```

```
chmod 644 xrc/*
```

```
chmod 755 code xrc
```

and

```
chmod 744 comp jserv
```

```
mfg@amz32:/apps/32soft$ mkdir xld.test
mfg@amz32:/apps/32soft$ cd xld.test
mfg@amz32:/apps/32soft/xld.test$ chmod 644 *.*
mfg@amz32:/apps/32soft/xld.test$ chmod 644 xrc/*
mfg@amz32:/apps/32soft/xld.test$ chmod 755 code xrc
mfg@amz32:/apps/32soft/xld.test$ chmod 744 comp jserv
mfg@amz32:/apps/32soft/xld.test$ ll
total 68
drwxr-xr-x  4 mfg qad  4096 Aug  4 23:56 ./
drwxr-x--- 219 mfg qad 12288 Aug  4 23:54 ../
drwxr-xr-x  2 mfg qad  4096 Aug  4 23:56 code/
-rwxr--r--  1 mfg qad   299 Jul 31 18:10 comp*
-rw-r--r--  1 mfg qad   275 Aug  2 14:43 comp.p
-rw-r--r--  1 mfg qad    63 Aug  4 23:10 in.txt
-rwxr--r--  1 mfg qad   128 Jul 31 18:28 jserv*
-rw-r--r--  1 mfg qad 14293 Jul 13 10:32 jserv_xld_2.8.jar
-rw-r--r--  1 mfg qad   358 Jul 31 18:08 test.ini
-rw-r--r--  1 mfg qad   130 Jul 31 18:07 test.pf
-rw-r--r--  1 mfg qad   661 Jul 31 18:15 xld.fnlib
drwxr-xr-x  2 mfg qad  4096 Aug  4 23:56 xrc/
mfg@amz32:/apps/32soft/xld.test$
```

Edit the configuration files and scripts

Next, you will need to configure several files and scripts.

xld.fnlib

This configuration file is used to set the Progress DLC environment variable, the QAD PROPATH, the JAVA_HOME for Java environment, and the JSERV_HOME for the 32 Soft XLD service.

1. To find the Progress DLC and QAD PROPATH variables, you have two options:
 - a. Check your QAD CHUI (character) connection script by running “cat /qad/bin/client.test” and noting the DLC and PROPATH variables.
Note: substitute /qad/bin/client.test with your QAD CHUI connection script
 - b. Connect to QAD, go to the Progress editor and from there run “message os-getenv (“DLC”)” to get the DLC variable, and “message propath” to get the PROPATH entries.
2. In the xld.fnlib, set the DLC as reported by the first command.

```
#!/bin/sh
# Script to start multi-user session of MFG/PRO
#
#TERM=xterm; export TERM
#stty intr '^c'
DLC=/qad/dlc100B; export DLC
PATH=$PATH:$DLC; export PATH
PROMSGS=$DLC/promsgs; export PROMSGS
PROTERMCAP=$DLC/protermcap; export PROTERMCAP
PS1='$$ '; export PS1
PROPATH=./code,/qad/qadeb21SP5/custom,/qad/qadeb21SP5/patch,/qad/qadeb21SP5,/qad/qadeb21SP5/bbi,.; export PROPATH
JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64/jre; export JAVA_HOME
JSERV_HOME=/apps/32soft/xld.test; export JSERV_HOME
```

3. Set the PROPATH to what was reported by second command.
 - a. Copy the entire PROPATH to the xld.fnlib, with the exception of the DLC entries which are always at the end.
 - b. Make sure to add “./code” at the beginning and include “.” at the end.
 - c. Remove any 3rd party add-ons or customizations which are not needed for the XLD Loaders to function properly from the PROPATH.
4. Set JAVA_HOME to point to your Java installation directory; and JSERV_HOME to the 32Soft xld folder (for example: /apps/32soft/xld.test).
5. Save and exit.

Parameter file

This file is used to connect to the databases for the corresponding environment. You could have one parameter file for Development and one for Test, for example.

1. Edit a Parameter File called test.pf and edit the databases needed. The XLD service needs to connect to at least 2 databases: the main QAD database and the ADM database.
2. If you use Qxtend, make sure to add the Qxtend database as well.

```
# include only qad db and adm db
db /qad/db/mfgeb21SP5 -ld qaddb -trig triggers
-db /qad/db/admeb21SP5 -ld qadadm -trig triggers
```

You can have multiple PF files, one for each environment, for example dev.pf and test.pf

ini file

The ini file is used to start the XLD listener for the corresponding environment. You could have one ini file for Development and one for Test, for example.

1. Edit an ini file called test.ini. This file needs to reference the corresponding test.pf file in the “dbases” line. You also need to set debug to false and change the port if needed.

```
# Please update the /etc/services file (Optional)
# with DB Name the service port #
# Use Port # when configuring Excel Loader sheet
#
debug=false
port=10013
params=. $(dirname $0)/xld.fnlib; cd $JSERV HOME;
client=$DLC/bin/_progres -c 30 -d mdy -yy 1920 -Bt 350 -D 100 -mmax 3000 -nb 200
-s 63 -noshvarfix -rand 2 -p cli.p -b
dbases= -pf test.pf
```

2. Every environment should run on separate ports if they are on same server, so you need one port for DEV and another port for TEST.
3. Set debug to true only when debugging and running interactively to see any error messages on the screen. We recommend a separate ini file for debugging purposes. For example, have a testdbg.ini with debug=true and a different port.
4. If you use different languages than English, make sure you add your code page parameters to the client line similar to what you have in your QAD client connection script. For example, for Chinese, you might have "-cprcodein GB2312 -cpstream GB2312 -cpinternal GB2312 -cpcoll basic", enter exactly what you use for connection to QAD.

```
# Please update the /etc/services file (Optionally)
# with DB Name the service port #
# Use Port # when configuring Excel Loader sheet
#
debug=false
port=10013
params=. $(dirname $0)/xld.fnlib; cd $JSERV_HOME;
cli=mt=$DLC/bin/_progres -c 30 -d mdy -yy 1920 -Bt 350 -D 100 -mmax 3000 -nb 200
-s 63 -noshvarfix -rand 2 -p cli.p -b
dbases= -pf test.pf
```

5. Once this file is edited, save and exit.

comp script

1. The only thing you need to change is the reference to the database connection file (PF). We will reference test.pf as an example.

```
#!/bin/sh
# Script to start multi-user session of MFG/PRO

. $(dirname $0)/xld.fnlib

cd $JSERV_HOME          # change to home directory

# exec $DLC/bin/_progres &DB etc
$DLC/bin/_progres -c 30 -d mdy -yy 1920 -Bt 350 -D 100 -mmax 3000 -nb 200 -s 63
-noshvarfix -pf $JSERV_HOME/test.pf -param comp.p
```

2. Again, if you use languages other than English, and you use code pages, be sure to add your code page parameters to the progress call, similar to what's in your QAD client connection script. So again, for Chinese you might have -cprcodein GB2312 -cpstream GB2312 -cpinternal GB2312 -cpcoll basic", enter exactly what you use for connection to QAD.

jserv script

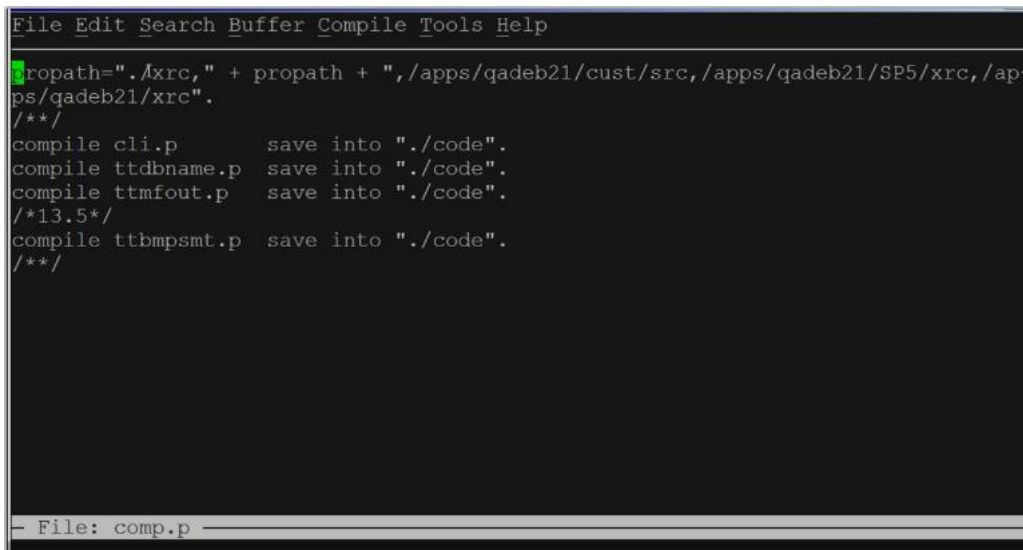
NOTE: You need to edit the jserv script only if you use code pages for different languages. Otherwise this does not require changes.

1. The default code page is ISO 8859-1 code page. If you do use code pages, they must be specified.
2. So, update jserv script and edit option -Dfile.encoding with your code-page to the Java call. For example, you could have
\$JAVA_HOME/bin/java -Dfile.encoding=GB2312 -jar jserv_xld_2.8.jar \$1
for Chinese.

```
#!/bin/sh
#
. $(dirname $0)/xld.fnlib
cd $JSERV_HOME
$JAVA_HOME/bin/java -Dfile.encoding=ISO-8859-1 -jar jserv_xld_2.8.jar $1
```

Compile the programs for first time XLD Service Installation

1. After all these files are configured, run "comp" script to compile the programs. This will open a Progress editor with comp.p program in it, connected to the databases specified in the PF.
2. The comp.p program you receive from 32 Soft already contains commands to compile all the necessary files for first time installation. However, you will need to update the comp.p program to set the propath line at the top with your QAD source code. If you have a QAD service pack installed, its source code folder should be referenced before the QAD standard folder.
3. For example you can have:
propath="./xrc," + propath
+ ",/apps/qadeb21/cust/src" (for your custom source, if needed)
+ ",/apps/qadeb21/SP5/xrc" (for the patch, if applicable)
+ ",/apps/qadeb21/xrc". (for standard QAD)



```
File Edit Search Buffer Compile Tools Help

propath="./xrc," + propath + ",/apps/qadeb21/cust/src,/apps/qadeb21/SP5/xrc,/ap~
ps/qadeb21/xrc".
/**/
compile cli.p      save into "./code".
compile ttdbname.p save into "./code".
compile ttmfout.p  save into "./code".
/*13.5*/
compile ttbmpt.p   save into "./code".
/**/

- File: comp.p -
```

4. Your custom and any other 3rd party add-on's source code should be included only if it is necessary for the 32 Soft XLD Loaders to function properly.
5. Press F1 to compile all the programs.

Compile the programs if you already have XLD Service Installed

If you are installing a new Data Loader in an environment already running XLD, you will need to compile only the program or programs specified by the 32 Soft consultants.

1. Comment out all the programs already compiled.

```
File Edit Search Buffer Compile Tools Help
propath="./xrc," + propath + ",/apps/qadeb21/cust/src,/apps/qadeb21/SP5/xrc,/ap~
ps/qadeb21/xrc".
/*
**/
compile cli.p      save into "./code".
compile ttdbname.p save into "./code".
compile ttmfout.p  save into "./code".
/*13.5*/
compile ttbmpsmt.p save into "./code".
**/
*/
```

2. Add lines at the end to compile the needed programs.

```
File Edit Search Buffer Compile Tools Help
propath="./xrc," + propath + ",/apps/qadeb21/cust/src,/apps/qadeb21/SP5/xrc,/ap~
ps/qadeb21/xrc".
/*
**/
compile cli.p      save into "./code".
compile ttdbname.p save into "./code".
compile ttmfout.p  save into "./code".
/*13.5*/
compile ttbmpsmt.p save into "./code".
**/
*/
compile programname.p save into "./code".
```

3. Press F1 to compile the program(s).

Start the XLD Service (Listener)

Once all files are configured and all programs compiled, you can start the XLD service (listener). You can run the XLD service either interactively or in background.

Interactive mode

1. Run `./jserv test.ini`
2. In order to stop the listener, press Ctrl-C.

Background mode

1. Run `./jserv test.ini &`
2. Press ENTER one more time after the listener start message is displayed.

Note: On some operating systems, you might need to run the jserv through the nohup command in order to leave the listener running after you close the terminal: `nohup ./jserv test.ini &`

This will redirect the output automatically to a file called nohup.out. Because of this, it's very important to make sure the debug variable is set to false when you run the jserv in background, otherwise the output file will become too large.

3. Once the listener is started, a new file will be generated in the xld folder, called test.ini.bcli.
4. You can use this script to check the connection to the database: `./test.ini.bcli<in.txt|cat`
5. You will see that a login has been tried and has failed. If no additional errors are returned, then the files are set-up properly.
6. We recommend that the jserv be run automatically at server boot time. There are many ways to do this. The simplest for a Unix based system would be to add a line at the end of the `/etc/rc.local` script to start the service.

For example, in order to start the listener as user mfg, you could use:

```
su - mfg -c /apps/32soft/xld.test/jserv test.ini >/apps/32soft/xld.test/test.out 2>&1 &
```

In order to start the listener as root, you could use:

```
/apps/32soft/xld.test/jserv test.ini >/apps/32soft/xld.test/test.out 2>&1 &
```

7. To verify the XLD service is running, you can run the following command, connected as the account used to start the service:
`netstat -nap | grep port_used_by_the_XLD_service` (for example: `netstat -nap | grep 10013`)

This command returns the Java process listening on the port.

8. To stop the XLD service, use the kill command, specifying the Java process id to be killed. Normally, the XLD service should not be stopped. When idle it has no database connections, and it uses very little memory and CPU time.

Set Up the Data Loaders

The Data Loaders are found in the Templates folder from the XLD archive. You will need to open and update every Data Loader with the specific database connection information.

1. Open the Loader and allow macros to run. All 32 Soft Data Loaders use macros, which need to be enabled. If you don't receive the notification about allowing macros to run and they are not enabled, you need to open Excel and go to Options (under File)... then Trust Center...then click the Settings and check the Macro settings. Make sure the second option "Disable all macros with notification" is checked. Once this is changed, exit Excel and reopen the spreadsheet.
2. Next click the Setup button.
3. In the Setup window, click on the Maintain button and enter the password 32SOFT (all upper case).
4. A new worksheet called "db" will open. Here you can specify the DB Name (such as TEST), the Server IP or Name, and the Port, which should match the one from the ini file used to start the listener. You can have one entry for each environment. So, you can have one for TEST, one for Development, one for Production, etc.
5. Once done, click on "Save and Close"

Test the Excel template

1. Click the Setup button and select the Test connection.
2. Enter a parent item and click Download to download the product structure.
3. Enter QAD user ID and password. We can now see the product structure for this parent item.
4. Change the "quantity per" ... and click "Upload" to load the data back to QAD. If you see it was loaded successfully, you know everything is setup correctly.

Now you know everything is set-up correctly, and you are on your way to a more productive QAD experience with Data Loaders! Thank you for your interest. If you have any questions or need additional help, please do not hesitate to [contact](#) us.