Magic xpa 2.3
UNIX
Release Notes

OUTPERFORM THE FUTURE™

Introducing Magic Software's Magic xpa 2.3 for UNIX Platforms

We are delighted to announce the launch of a new release of Magic Software's Magic xpa Application Platform.

Magic xpa allows organizations to quickly and cost effectively enjoy all the benefits of Rich Internet Applications (RIA) and Software-as-a-Service (SaaS) applications, whether on-premise, or on-demand.

Based upon a unique, unitary development paradigm, Magic xpa gives the power to quickly develop, enhance, and deploy business applications under multiple deployment models and at a fraction of the cost and time compared to conventional .NET or Java environments.

Magic xpa Web Page

Make sure to visit our Magic xpa Web page, where you can view and download various documents.

Magic xpa Licensing

In addition to the new unitary development and deployment paradigm for RIA and SaaS, Magic xpa also supports any previous Magic Software editions and forms of development and deployment. However, in order to maintain your former development and deployment capabilities, you need to obtain new Magic xpa licenses that reflect your current licenses.

To obtain Magic xpa licenses, please contact your local Magic Software representative.

Migrating from uniPaaS V1.9 to Magic xpa

Migrating a uniPaaS V1.9 application to Magic xpa is quite straightforward and no explicit migration procedure is required.

You can directly access and open uniPaaS V1.9 applications from the Magic xpa Studio and Server engines.

Migrating from eDeveloper V9.4 to Magic xpa

Migrating an eDeveloper V9.4 application to Magic xpa is fast and easy.

Magic xpa provides a collection of wizards to easily migrate your application, INI settings, and interface builder data.

In the Migration subfolder of the product, located in the Start menu of your desktop, you will find a shortcut to each of the available conversion wizards.

We recommend reading the Migration chapter in the *Magic xpa Help* before migrating your eDeveloper V9.4 applications.

Compatibility

For more information about the various platforms on which Magic xpa has been checked for operation by Magic Software Enterprises, refer to the **Compatibility Guide.pdf** file provided with this installation.

Installing Magic xpa

Pre-Installation

- During the installation, several user-environment files are overwritten. Therefore it is best to back up the following files before starting the installation process: **.cshrc, .profile, .bash_profile** (applies to Linux only)
- If you already have a previous Magic xpa server version installed, it is best to install the product using a different user name.

Installation Steps

1. Create a new user. (The installation should be performed using a non-root user.)
2. Log in as the new user.
3. Uncompress the installation file (**magicxpa_<Version>.<platform>.tar.gz**) using the local uncompress utility or a compatible utility, such as gunzip).
4. Run the command from **$HOME directory: tar xvf <installation file>**.
   The installation file name is **magicxpa_<Version>.<platform>.tar**.
5. Run the **./magicxpainstall** command and enter the requested information.
6. After the installation has been successfully completed, run the **$HOME/sbin/mgroot.sh** file as a root user. This script copies Magic xpa files that should be accessed by your Web server.
7. To set up an Apache Web server, append the **$HOME/web_utils/magic.conf** file to the Apache configuration file (**httpd.conf**) , place the requester in the modules directory and restart the Apache Web server (see Apache Requester Installation and Configuration).
8. Log out from the new user and log in again to enable the new environment settings.
9. If you need to uninstall the product, delete the user home directory created for the installation. For a complete removal, delete the files copied by the **$MAGIC_HOME/sbin/mgroot.sh** script and remove the changes that were applied to the Web server.

Post Installation

It is necessary to define for Magic xpa that a specific gateway must be loaded by pointing to a variable that contains a DB number. The DB number points to a specific executable that is the relevant gateway.

In UNIX operating systems an environment variable points to the executable, which should be used for a specific gateway, situated in $HOME/etc/mgenv

For example, in UNIX: **MAGIC_ DB_ 14_ DRIVER=$HOME/bin/mgoracle10**
where the number 14 refers to the DB number.

Note:

- If the installation fails, it is best to delete all files in **$HOME** and start a new installation from the beginning.
- A remarked entry (meaning that it's preceded by semicolon) named **MGLOCAL**, which points to the Korean support library **mglocal.kor**, exists in the $HOME/etc/**mgenv** file. This file is required for proper functioning of browser-based applications when using Korean/Hangul characters.

Installation Components

- Magic xpa Server (bin/mgxparuntime)
- Magic xpa Broker (broker/mgbroker)
- Magic xpa command line requester (broker/mgrqcmdl)
- Magic xpa gateway 10g and 11g (bin/mgoracle10, bin/mgoracle11) – versions are platform specific
- Magic xpa gateway for DB2 UDB Version 8.1 (bin/mgdb2) – for AIX and Linux platforms only
- Magic xpa gateway for ODBC (bin/mgodbc) – for Linux platform only
- Magic xpa memory gateway (bin/mgmemory)
- Magic xpa CGI requester (cgibin/mgrqcgi023)
- Magic xpa requesters for Apache Web Server (cgibin/mod_V2_mgrequest023.so , cgibin/mod_V2.2_mgrequest023.so)
- Magic xpa UDF/UDP examples (userproc directory)
- Magic xpa Web utility files used for Browser Client support (web_utils directory)
- Magic xpa license server (FlexLM 7 in license directory)
- Magic xpa Hangul support (language/mglocal.kor)
- Magic xpa SNMP sub-agent (snmp/mgsnmp.so)
- Magic xpa Messaging component (messaging/messaging.mff)
- Systinet server for Java (web_services directory)

This product includes software developed by the Apache Software Foundation (http://www.apache.org/).

Starting the Magic xpa Server and Broker

To run the Magic xpa Server:

1. Start the license server by running the **$HOME/license/mglmstart** script.
2. Use the **mgxparuntime.sh** script or invoke the **mgxparuntime** executable file. By default Magic xpa uses the INI file specified in the **MGENV** environment variable.
    - For the non-default INI file, use: **mgxparuntime –ini=<ini file> &**
    - For an additional INI file, use: **mgxparuntime –ini=<ini file> @<additional ini> &**

Several scripts exist in the **sbin** directory to simplify the Magic xpa server administration:

startb	Start the Magic xpa Broker
stopb	Stop the Magic xpa Broker (and Server engines)
stopm	Stop all Magic xpa Server engines connected to the Magic xpa Broker
checkm	Check which Magic xpa Server engines are connected to the Magic xpa Broker

Note: The **stopb** and **stopm** scripts require supplying the broker password, as shown in the following example:

stopb –password=<broker supervisor password>

Note:	The Magic xpa log file is created for each server you start. Its name is determined by the ExternalLogFileName entry in the MAGIC.INI file.

Magic xpa Requesters

The MGREQENV environment variable points to the MGREQ.INI file used by the Magic xpa Server, the Magic xpa Broker, and the Magic xpa command line requester. (The installation sets Magic xpa Requesters for UNIX MGREQENV = $HOME/etc/MGREQ.INI).

To send a request to a Magic xpa Server on UNIX from an Internet Browser, there are two types of requesters:

1. The Magic xpa CGI requester (mgrqcgi023):

   http://<server_name>/cgi-bin/mgrqcgi023?appname=example1&prgname=prog1
2. The Magic xpa requester for Apache (mod_V2.2_mgrequest023.so):

   http://<server_name>/mgrequest023?appname=example1&prgname=prog1

Additional Settings

The following settings in the MGREQ.INI file affect the requester execution.

- RetryMainTime
- KeepAlive

File Names

In Windows platforms, files can be referred to by either a URL or by file name, relatively or by full path/URL.

In non-Windows platforms, such as UNIX platforms, files can be referred to by a full URL only. Any reference to a file name with a slash (/) is considered to be a path name, either full or relative.

Examples:

/etc/home1/a.jpg (full path)

http://myserver/myalias/a.jpg (full URL)

myalias/a.jpg – is considered to be a relative path name, not a relative URL.

Colors

To use colors properly on UNIX platforms, you must define all the colors that are used as non-system colors. The easiest way to do this is to access the color file in the Magic xpa Studio and define the colors accordingly.

External Code Pages

When installing Magic xpa on UNIX platforms, the **ExternalCodePage** ini setting is set to 1252 (Windows 1252 is the Western European code page). This setting must be modified for any-non Western European languages, such as Hebrew or Thai, since it affects Unicode to ANSI conversions.

Apache Requester Installation and Configuration

Apache Module Requester Setup

Magic xpa 2.3 includes requesters for the Apache Web Server version 2.0 and version 2.2. The requester module **mod_V2.2_mgrequest023.so** should be placed in the **modules** directory of the Apache installation (default: /usr/local/httpd/modules) with execute permissions.

The installation includes the following requesters:

mod_v2_mgrequest023.so	to be used with Apache 2.0
mod_V2.2_mgrequest023.so	to be used with Apache 2.2

1. Add the following lines to the Apache configuration file, **httpd.conf**.

```
LoadModule mgrequest023_module
  modules/mod_V2.2_mgrequest023.so

<Location /mgrequest023>
   SetHandler mgrequest023-handler
</Location>

SetEnv MGREQ_INI_PATH  <directory>
```

2. **For AIX:** Add the **$MAGIC_HOME/lib** to the **$LIBPATH** environment variable.
   For Linux and Solaris: Add the **$MAGIC_HOME/lib** to the **$LD_LIBRARY_PATH** environment variable.
3. Restart the Apache Web server.

The Apache requester is configured using the MGREQ.INI file. The directory location of the MGREQ.INI file is specified by the **MGREQ_INI_PATH** setting in the Apache configuration file, **httpd.conf**.

Example

SetEnv MGREQ_INI_PATH /usr/local/httpd/conf

The Apache requester uses the /usr/local/httpd/conf/ MGREQ.INI file.

To use this requester, call Magic xpa using a URL, such as:
http://server/mgrequest023?appname=...

You should also modify the MAGIC.INI file to read:
InternetDispatcherPath=/mgrequest023

Using an Apache Web Server with a Non-Default Port

To use Apache with a non-default port (port number other than 80), change the setting shown below in the MAGIC.INI file:

InternetDispatcherPath= http://server:port/cgi-bin/mgrqcgi023

instead of

/cgi-bin/mgrqcgi023

Platform-specific Information

AIX

The Magic xpa 2.3 Server for AIX should be used with AIX 6.1 or with other more recent operating systems that are backward compatible.

The Magic xpa 2.3 server is designed to use the Power 6 processor instruction set and is optimized for Power 6 and Power 7 processors.

The Oracle gateway should be used with the Oracle 10g client and above.

The Magic xpa DB2 gateway for AIX should be used with the DB2 Version 8.1 client.

The WebSphere MQ 5.3 client/server is required for working with the MQ messaging capabilities.

JRE 1.6 is required for working with Java integration capabilities.

Apache 2.0.45 (or a more recent version) is required in order to use the Apache 2 requester.

Solaris

The Magic xpa 2.3 Server for Solaris should be used with Solaris 10 or with other more recent operating systems that are backward compatible.

The Oracle gateway should be used with the Oracle 10g and above.

The WebSphere MQ 5.3 client/server is required for working with the MQ messaging capabilities.

JRE 1.6 is required for working with Java integration capabilities.

Apache 2.0.45 (or a more recent version) is required in order to use the Apache 2 requester.

Linux

For Intel processors only, Linux requires Kernel 2.6.18-194 and up with glibc-2.5-49 and up.

The Oracle gateway should be used with the Oracle 10g client and above.

The Websphere MQ 5.3 client/server is required for working with the MQ messaging capabilities.

JRE 1.6 is required for working with Java integration capabilities.

Apache 2.0.43 (or a more recent version) is required in order to use the Apache 2 requester.

Gateway-specific Information

To enable the use of a particular gateway, remove the # sign from the corresponding entry in the **$MAGIC_HOME/etc/mgenv** file.

When using the Oracle gateway, make sure that **ORACLE_HOME** and **ORACLE_SID** are set in the **$MAGIC_HOME/etc/mgenv** file, and that the environment variable **LD_LIBRARY_PATH** (or **LIBPATH** for AIX) includes the **$ORACLE_HOME/lib** directory.

When using the DB2 gateway, make sure that **DB2INSTANCE** is set in the **$MAGIC_HOME/etc/mgenv** file.

ODBC Gateway on the Linux Platform

General Information

Gateway name: **mgodbc**

Required software: This gateway works with the **UnixODBC** ODBC manager.

It was tested with the following database gateways:

MySQL MyODBC driver (libmyodbc-<ver>.so) – to access MyODBC software and for more information on this particular driver refer to http://www.mysql.com

Installation and Setup Instructions

1. Uncomment the entry **MAGIC_DB_20_DRIVER** in the **mgenv** file. Uncomment means to remove the semicolon preceding the entry.
2. Install the **UnixODBC** ODBC manager, this product can be downloaded from: http://www.unixodbc.org. Follow the online instructions to generate the ODBC manager.

   Locate the following two files (shared libraries): **libodbc.so.1.0.0** and **libodbcinst.so.1.0.0**

   Copy the files to the directory **$MAGIC_HOME/lib**.
3. In the same directory create symbolic links for the two libraries:

   ln –s libodbc.so.1.0.0 libodbc.so.1

   ln –s libodbcinst.so.1.0.0 libodbcinst.so.1
4. Install the ODBC driver. Refer to the specific driver documentation for installation instructions.
5. Make sure that the libraries have Execute permission. Use the **chmod +x** command to set execute permission.
6. Create a hidden file named .**odbc.ini** in the user's home directory. For example: /usr/magicadm/.odbc.ini. This file is used to configure ODBC DSNs. Refer to the ODBC manager documentation for more explanations regarding the setup of this file.

To help you setup quickly, we have included the following **.odbc.ini** file as an example:

```
[mysql]
Driver   = /usr/magicadm/mysql/libmyodbc-2.50.23.so
Trace    = No
Tracefile= mysql.log
Database = samp_db
```

Each section defines a DSN (Data Source Name). In the above example, there is one defined DSN named mysql. The driver entry in each section should be set to the full path of the ODBC driver. For a list of valid entries and their meanings, refer to the ODBC driver documentation.

Alternatively a general /etc/odbc.ini file can be used.

Setting the Magic Configuration File (MAGIC.INI)

1. Set a Magic xpa database using the Database repository.
2. Copy the database definition in the **MAGIC_DATABASES** section from the MAGIC.INI file on Windows to the MAGIC.INI file on Linux. It is highly recommended to backup the MAGIC.INI file before editing.

Limitations and Recommendations

JMS

Connectivity to messaging servers via JMS is not supported using the provided Messaging component.

Before you can use JMS with the Sun Reference application, the environment variables listed below are needed to run J2EE applications on UNIX platforms:

Variable Name	Values
$JAVA_HOME	Directory where the Java 2 SDK, Standard Edition, is installed
$J2EE_HOME	Directory in which the J2EE SDK is installed
$CLASSPATH	Include the following: .;$J2EE_HOME/lib/j2ee.jar; $J2EE_HOME/lib/locale
$PATH	Include $J2EE_HOME/bin

Backups

We highly recommended backing up Magic xpa configuration files, such as MAGIC.INI, MGRB.INI, MGREQ.INI, and license.dat, before modifying them.

Compression

There is no compression when the server is a UNIX platform.

Java Integration

The Java CLASSPATH separator character on UNIX platforms is a colon (:) as opposed to the Windows platform separator character, which is a semicolon (;).

For example: **CLASSPATH = /java/MyClasses:/java/OtherClasses**

For more information, please refer to the Java documentation (Java 2 SDK Tools and Utilities at http://www.oracle.com/technetwork/java/javase/documentation/index.html).

AIX

The **JAVA_HOME** entry should be set in the MAGIC_JAVA section of the MAGIC.INI file.

For example: **If JAVA_HOME = /usr/java6**

Magic xpa appends **/jre/bin/classic/libjvm.a** in order to find the **libjvm.a** library.

If you encounter problems locating this file you can use the environment variable: **MG_JAVALIB**, which should be set to the absolute path of the library file.

For example: **MG_JAVALIB = /usr/java6/jre/bin/classic/libjvm.a**

The **AIX LIBPATH** variable should include **/usr/java6/jre/bin:/usr/java6/jre/bin/classic**

Solaris

If Java is installed on your server, you should edit the following scripts: **.cshrc** and .**profile**.

The **LD_LIBRARY_PATH** environment variable should include **$JAVA_HOME/jre/lib/sparc/client**

Linux

If Java is installed on your server, you should edit the following scripts: .**cshrc** and **.profile**.

The **LD_LIBRARY_PATH** environment variable should include **$JAVA_HOME/jre/lib/i386/client** and **$JAVA_HOME/jre/lib/i386**

WebSphere MQ

If you are using an MQ client software, you should set the following logical name in the MAGIC.INI file: **WMQ_ModuleName = C**

If you are using an MQ server software, meaning that the MQ Queue manager runs on the same machine as the Magic xpa Server, you should set the following logical name in the MAGIC.INI file: **WMQ_ModuleName = S**

External Procedures

User-defined procedures should be compiled according to this platform specific list:

Platform	Compiler Version & Vendor	c++ Compiler	c Compiler
Linux	gcc version 4.1.2	g++	gcc
Solaris	Sun Studio 11 Software	CC	cc
AIX	IBM XL C/C++ for AIX, V12.1	xlC_r	cc_r

FQDN (Fully Qualified Domain Name)

The broker and enterprise server should bind using a specific network adapter by specifying a FQDN (instead of IP address). The requester layer should translate the FQDN to IP and bind using IP on a specific adapter.

FQDN stands for fully qualified domain name – for example "linuxdev.Magic"

The MGREQ.INI file contains the following entry: BindFirstIPAddress=Y/[N]

> Y – During binding to a port, the server will resolve the host name and will bind to the resolved IP address.
>
> N – The server will bind to any IP address (*.port – for backwards compatibility)

To enable a Magic xpa engine and broker to work with a specific network adapter (if there are multiple adapters on a machine):

1. Edit the MGREQ.INI file and enable **BindFirstIPAddress** (= Y) and set **MessagingServer** to **FQDN/port**.
2. Edit the MGRB.INI file and set **MessagingServer** to **FQDN/port**.
3. In the MAGIC.INI file, set **TCP/IP = 2, 30, 1500-2000 /LocalHost=FQDN**.
4. In the MAGIC.INI file, set the **Default Broker** to **FQDN/port**.

The table below shows the binding for the server module:

Port Number	
Port-No	BindFirstIPAddress=N */Port-No BindFirstIPAddress=Y IP-address/Port-No
Ip Address/Port-No	IP-Address/Port-No

Systinet Installation

Prerequisites

1. Install Java 6.

   Only the 32-bit JRE is needed for Systinet.

2. Update **.profile**.

   JRE 6 needs to be referenced in the **JAVA_HOME** variable and as the default runtime for any Java application. Add the following to the end of the **.profile** for the Magic xpa install ID (for example: uni19g):

   ```
   # Set home path for Systinet
   export WASP_HOME=$MAGIC_HOME/ssj
   # Set home jave runtime as version 6
   export JAVA_HOME=/usr/java6/jre
   # Set default Java runtime to 6 in Path
   export PATH=/usr/java6/jre/bin:/usr/java6/bin:$PATH
   ```

3. Configure the **ssjinstallconfig** file.

   Under **$MAGIC_HOME/web_services**, open and edit the **ssjinstallconfig** file. The following needs to be updated:

   a. **Dname –** The value should be the server name plus the domain.
   b. **installation.destination** – The value should be the path where you want to install Systinet (for example: $MAGIC_HOME/ssj).

Installation

Run **installssj** under **$MAGIC_HOME/web_services** to install Systinet. Check for errors after the installation.

Post Installation

Copy support JARS to the Systinet install folder

Copy the following files from **$MAGIC_HOME/support** to **$WASP_HOME/lib** and change the mode to **644**:

- uniSSJ.jar
- uniRequester.jar
- saaj_utils.jar

```
cp -p $MAGIC_HOME/support/uniSSJ.jar $WASP_HOME/lib
cp -p $MAGIC_HOME/support/uniRequester.jar $WASP_HOME/lib
cp -p $MAGIC_HOME/support/saaj_utils.jar $WASP_HOME/lib
```

Update server.sh

The following changes are required in **$WASP_HOME/bin/server.sh**:

1. Add the following to the top after the call to **env.sh** (approximately line 5):

```
# Step c in installation document
LD_LIBRARY_PATH="$LD_LIBRARY_PATH":$MAGIC_HOME/lib
```

2. Change the JAVA call at the bottom of the script by adding the following to the **CLASSPATH**:

```
"$WASP_HOME"/lib/uniSSJ.jar:"$WASP_HOME"/lib/uniRequester
.jar:"$WASP_HOME"/lib/saaj_utils.jar
```

3. Add the following after the **CLASSPATH** declaration:

```
-Djava.library.path="$MAGIC_HOME"/lib
```

Update File Permissions to Start as a Different User

1. Run the following commands to update the file/folder permissions so that other users besides the owner can start and stop Systinet.

```
chmod 777 $MAGIC_HOME/ssj/log
chmod 777 $MAGIC_HOME/ssj/log/*
chmod 775 $MAGIC_HOME/ssj/store/hsqldb
chmod 777 $MAGIC_HOME/ssj/store/hsqldb/*
chmod 775 $MAGIC_HOME/ssj
chmod 777 $MAGIC_HOME/ssj/app
chmod 777 $MAGIC_HOME/ssj/app/*
chmod 777 $MAGIC_HOME/ssj/work
```

2. Start the server from a different account.

3. Rerun the commands in step 1.

***Note:** If you have issues deploying a JAR file, it is likely permission based. You will need to check the issue in the exception displayed by Systinet and run the **chmod** command to fix it.

Deploying a Rich Client Application

To be able to deploy a Rich Client application on UNIX platforms:

1. The following files and folders are created once you use the Rich Client Deployment Builder:

   `appname\appname.application`

   `appname\appname.publish.html`

   `appname\mgxpaRIA_x_y_z_www\` (x,y,z represent the Magic xpa version and www is a unique number representing the specific version)

   `appname\Images\`

   a. Place them in the **MagicRIAApplications/*appname*** alias on the Web server.

   b. Users can access the application from the following URL: http://*appserver*/MagicRIAApplications/*appname*/*appname*.publish.html

2. Add the following into the **httpd.conf** Apache configuration file in this order:

```
AddType application/x-ms-application .application
AddType application/x-ms-application .manifest
AddType application/octet-stream deploy
AddType application/x-msdownload .dll
AddHandler default-handler .jpg .gif .js .txt .bat .msi
```

3. Manually change the **HTTPCompressionLevel** in the application's **publish.html** file to **None**, since there is no compression when the server is a non-Windows platform. For example:

```
<body onload="initialize()">
    <xml id="rcExecProps">
        <properties>
           <property key="protocol" val="http"/>
           <property key="server" val="aix51:2261"/>
           <property key="requester" val="/mgrequest023"/>
           <property key="appname" val="frame"/>
           <property key="prgname" val="START"/>
           <property key="arguments" val=""/>
           <property key="envvars" val=""/>
           <property key="UseWindowsXPThemes" val="Y"/>
           <property key="HTTPCompressionLevel" val="None"/>
           <property key="DisplayStatisticInformation" val="N"/>
           <property key="InternalLogLevel" val=""/>
           <property key="InternalLogFile" val=""/>
           <property key="InternalLogSync" val="Session"/>
           <property key="LogClientSequenceForActivityMonitor" val="N"/>
        </properties>
     </xml>
   <table align="center">
```