# Intralibrary Installation Guide: Version 3.3

Revision: 8
Created: 4th August 2005
Last Revised: 8th June 2010
Contact: support@intrallect.com
Company: Intrallect Ltd
Product: intraLibrary, Digital Repository

# Intralibrary Installation Guide: Version 3.3

## Table Of Contents


1. Prerequisites ..... 3
   1.1. Java Environment ..... 3
   1.2. Database ..... 3
   1.3. Application Server ..... 3
   1.4. Installation Package ..... 4
   1.5. Internet Connection ..... 6
   1.6. Pre-Installation ..... 6
2. Installation ..... 6
   2.1. IntraLibrary Web Application ..... 6
   2.2. Configuring Properties ..... 7
   2.3. Deploying the Licence ..... 9
   2.4. Configuring the intraLibrary Context ..... 9
   2.5. Configuring a Datasource ..... 11
   2.6. Copying Dependencies ..... 12
3. Database Setup ..... 14
   3.1. Creating the Database ..... 14
   3.2. Populating the Database ..... 15
   3.3. Database Configuration ..... 15
4. Confirming Installation ..... 16
5. Setting the Application URL .....
6. Troubleshooting ..... 18

# 1. Prerequisites

## 1.1. Java Environment

This version of intraLibrary relies on the Java 1.5 Runtime Environment. For this version of intraLibrary to function, it must be running under Java 1.5.

The full sdk must be installed and available on your server. The jre alone is not enough. The Java 1.5 sdk is available for many platforms here: http://java.sun.com/j2se/1.5/

IntraLibrary has been tested on the Windows, Linux and Solaris environments available from Sun and also on Apple's Java 1.5 implementation for Apple OS X.

Note that intraLibrary has **not** been tested on the Blackdown port of the JVM, the IBM JVM or any other JVM not mentioned above. If you have any experience deploying intraLibrary on any other platform or environment we'd like to hear from you.

## 1.2. Database

IntraLibrary is designed to run using using the MySQL database server. You will require MySQL version 5.0 for this version of intraLibrary.

MySQL 5.0 is available for many platforms here:
http://dev.mysql.com/downloads/mysql/5.0.html

## 1.3. Application Server

IntraLibrary will run on any J2EE compliant Java Webserver adhering to the Servlet 2.4 specification and the JSP 2.0 specification, the rest of this document assumes that Tomcat 5.5.x is the application server being used. If you intend deploying intraLibrary on anything other than Tomcat, then follow the instructions from your application server vendor on how to deploy a web application. For help in deploying to a Servlet Container other than Tomcat 5.5.x contact a member of our support team.

If you want to make use of thumbnail functionality within intraLibrary, the server must be

started with the *java.awt.headless=true* property. The easiest way to achieve this is to add the property to the catalina startup script (.sh or .bat) in the Tomcat bin directory:

*JAVA_OPTS="-Djava.awt.headless=true"*

**NOTE:** We (and others!) have been experiencing strange behaviour when running Tomcat as a service on the Windows platform. In particular when allocating the minimum and maximum memory heap size. We cannot currently recommend running Tomcat as service using the Tomcat service installation tool. However some of our customers have had more success by creating a service manually. If you'd like help with this, contact a member of our support team.

**NOTE:** If you are installing Tomcat on Windows ensure that there are no spaces in the installation path as this can cause problems with the java class loader.

### 1.4. Installation Package

If you're reading this, then you probably already have the intraLibrary 3.3 installation package. If you don't have the full installation package then please contact a member of the intrallect support team to organise this.

This package contains:

- intraLibrary web application archive - *intralibrary.war*
- config directory containing:
  - global intraLibrary settings - *(intralibrary.properties)*
  - authentication directory containing:
    - The default authentication process - *(authentication_chain_descriptor)*
    - A sample set of LDAP authentication properties - *(ldap_authenticator.properties)*
    - A sample set of SQL authentication properties - *(sql_authenticator.properties)*
  - emailTemplates directory containing a number of language specific directories (e.g. 'en', 'es') that contain the following:
    - The text of the publish notification email sent to administrators - *(adminPublishNotificationEmail.txt)*
    - The text of the notification email sent to administrators or librarians when annotations are added to an object - *(annotationNotificationEmail.txt)*

        - The text of the notification email sent to the object owner when a user comments on one of their objects - *(objectOwnerAnnotationNotificationEmail.txt)*
        - The text of the notification email sent to users when an object is moved - *(stageChangeEmail.txt)*
        - The text of the notification email sent to named external users when an object is moved into a certain stage - *(stageChangeExternalUserEmail.txt)*
        - The text of the notification email sent to the object owner when an object is moved into a certain stage - *(stageChangeObjectOwnerEmail.txt)*
        - The text of the publish notification email sent to users - *(userPublishNotificationEmail.txt)*
        - The text of the notification email sent to administrators or librarians when a user adds a tag to an object - *(tagNotificationEmail.txt)*
        - The text of the notification email sent to the object owner when a users add a tag to one of their objects - *(objectOwnerTagNotificationEmail.txt)*
    - eventhandling directory containing:
        - A default event handler definition (for more information on event handling see the intraLibrary event handling documentation) - *event_handler_descriptors.xml*
- authenticationLib directory containing:
    - IntraLibrary Athens authentication module - *(attica.jar)*
    - IntraLibrary authentication module - *(intralibrary-authentication.jar)*
    - JDOM - *(jdom.jar)*
    - Log4J - *(log4j-1.2.8.jar)*
- dataSourceLib directory containing:
    - MySQL database driver - *(mysql-connector-java-5.0.4-bin.jar)*
- realmLib directory containing:
    - IntraLibrary custom Tomcat Realm - *(intralibrary-realm.jar)*
- endorsed directory containing:
    - Xalan library - *(xalan-x.x.x.jar)*
    - Xerces library - *(xercesImpl-x.x.x.jar)*
    - XML API's for Xerces library - *(xml-apis.jar)*
    - Serializer - *(serializer.jar)*
- dbscripts directory containing:
    - Main database setup script - *(intralibrary.sql)*
    - Sample MySQL conf file - *(my.ini)*
    - application_profiles folder containing:
        - UK LOM Core Application Profile script- *(uklomcore0p2.sql)*
        - CANCore Application Profile script - *(cancore.sql)*
    - metadata_models folder containing:
        - Digital Curriculum Metadata Model script - *(bbcdc_lom_model.sql)*

    - MIX Image Metadata Model (Z39.87) script - *(mix_model.sql)*
- developmentResources directory containing:
    - DevelopmentResourcesDocs
    - jsps
    - lib
    - src
- docs directory
- licences directory for third party components
- sample webapp context file - *(sample_context.xml)*

### 1.5. Internet Connection

The intraLibrary web application needs access to the internet to perform some XML validation that occurs when using certain areas of functionality within intraLibrary.

The intraLibrary web application also needs to be able to send email from the application server. A valid, working and reachable smtp server must be defined. We will provide details on how to configure this later in the document.

### 1.6. Pre-Installation

This installation guide assumes that Tomcat 5.5.x, MySQL 5.0.x and Java 1.5 have been correctly installed. We would recommend that you check both are working correctly before moving onto the installation of intraLibrary.

# 2. Installation

### 2.1. IntraLibrary Web Application

The basis of the intraLibrary web application is the *intralibrary.war file*. You can put this archive anywhere on the server, but we'd recommended it to be on the same physical volume as your Tomcat installation.

Once the web application is in place we can start to configure the properties to match your system.

**Note:** The intralibrary.war file is not designed to auto-deploy and you will experience problems if you place the file in the webapps directory of your Tomcat installation.

## 2.2. Configuring Properties

All configurable aspects of intraLibrary (global properties, authentication, email content) are stored within a single *config* directory. This directory can be found at the root level of the installation package and should be copied to the filesystem somewhere (this can be anywhere on the file system). We **strongly** recommend that the directory is created outside of the webapp directory to ease the upgrade process in the future. The config directory should be accessible to the user that is used to start Tomcat. Once this directory has been copied we can go on to edit the global properties.

There are a number of properties that need to be defined in the **config/intralibrary.properties** file. The file supplied with the installation package contains some sample values.

The mandatory entries that need to be edited are:

- **returnAddress**

  Return address for any emails that are sent out. This should be set so the recipient can identify which instance of the application the mail refers to. For example *intralibrary1@yourcompany.com*

- **repositoryName**

  This is the name used to identify this instance of intraLibrary

- **adminEmail**

  Address for system admin ( report bugs, broken links etc ).

- **idPrefix**

  This needs to be a globally unique id used to identify this instance of intraLibrary. To avoid potential conflicts we suggest using an reverse internet naming scheme. For example *com.your-company.intralibrary1*

The optional entries that may be added to the properties file are:

- **proxyHost**

  This is the host name or ip address of the proxy that the application needs to be routed through.

- **proxyPort**

  This is the port that the proxy is listening on.

- **entryPage**

  This property overrides the default entry page for all users. If this is not set, then all users will be taken to the browse library page. If you wish users to be taken to their upload area after login, then set this property to *upload*. This has no effect for users who do not have access to an upload area. The only valid values for this property are *browse* and *upload*.

- **objectCacheSize**

  This maintains the upper limit for objects stored in the application cache. The default is 250. Change this to higher values for improved performance on machines with a lot of physical memory or reduce it for machines with hardly any free memory.

- **populateCacheOnStartup**

  The default is true. If you have a large cache size, then startup time for the application will be slowed down while the cache is populated. During debugging or testing, it may be useful to set this property to false. Note if the cache isn't populated at start up then the initial browsing and searching of object will run slowly.

- **reindexOnStartup**

  The default is true. If you have a lot of objects in the repository, then startup time for the application will be slowed down while all the metadata for each object is indexed. During debugging or testing, it may be useful to set this property to false.

- **logLevel**

  Determines the level, and quantity, of logging messages that will be output. Valid values are DEBUG, INFO, WARN, ERROR, FATAL

- **allowedFileSpacePaths**

This restricts the file systems that intraLibrary has access to. These file systems are only relevant if intraLibrary is configured to allow importing from the file system.

- **bigThumbnailDimension**

  This specifies the width, in pixels, of the big thumbnail shown on the image search results view. The thumbnail is scaled to the appropriate width whilst preserving the aspect ration. The default value for this property is 200. The maximum value for this property is 300

- **smallThumbnailDimension**

  This specifies the width and height, in pixels, of the small thumbnail shown on the image search results view. The default value for this property is 75. The minimum value for this property is 30. The maximum value for this property is 200.

- **thumbnailQuality**

  This specifies the quality of the thumbnail being displayed on the image search results view. The default value for this property is 7. The property is in the range 1-9, with 1 being the lowest quality and 9 being the highest. Note that the higher the quality the slower the rendering of the thumbnail.

- **stylesheet**

  This allows you to add an external stylesheet. See the integration guide for further details.

## 2.3. Deploying the Licence

The licence file will be provided seperately to the intraLibrary application, if you don't yet have a licence please contact a member of our support team.

To deploy the licence and ensure that it will be picked up by your intraLibrary installation the file *intralibrary-licence.txt* should be placed at the root level of the *config* directory which you created in the previous section.

## 2.4. Configuring the intraLibrary context

For the rest of this section we will refer to the root directory of your Tomcat installation as ***TOMCAT_HOME***. As mentioned earlier in the documentation if your TOMCAT_HOME has spaces in the path you will experience problems, so try to avoid this.

The intraLibrary web application needs to be *registered* to allow the application server to run it. There are multiple ways of doing this but we recommend creating a context descriptor in $CATALINA_HOME/conf/[enginename]/[hostname]/.

More information about configuring contexts can be found here:
http://tomcat.apache.org/tomcat-5.5-doc/config/context.html

Included in this install package is a *sample_context.xml* file which you can use as a template for your installation. A brief explanation of some of the parameters follows below:

- **docBase**

  This attribute indicates where on your server the *intralibrary* web archive is located. Remember, we recommend that this should be on the same physical volume as the application server itself.

  example: docBase="/disk1/tools/repositories/intralibrary.war"

- **Environment name="configDir"**

  This attribute indicates where on your server the *config* folder is located, you should only change the *value* part of this entry as any other changes will prevent the application from running successfully. Remember, we recommend that this folder lives outwith the web application.

  example: <Environment name="configDir" type="java.lang.String" value="/Volumes/projects/intralibrary/config" override="true" />

- **<Realm className="com.intrallect.realm.IntraLibraryRealm" configDir="/Volumes/projects/intralibrary/config"/>**

  This element indicates the fully qualified class name of the intraLibrary realm that the servlet container should use when authenticating against the intraLibrary context

  The className attribute should not be changed, but the value of configDir should be exactly the same as the value of the configDir defined in the previous step.

## 2.5. Configuring a Datasource

NOTE: The following instructions are written specifically for creating a *datasource* using Tomcat 5.5.x, the MySQL 5.0.* database engine and the database driver we supply. If you need more help with this then please feel free to contact a member of our support team.

To enable the communication between the application and the database, we need to create a *datasource*. We do this by making changes to the *context* we created in the section above, either prior to adding the context to the server.xml file or once the context is in place.

Inside the *context* element discussed above, you'll find a *Resource* element. You will need to specify the following attributes on the Resource element to setup the *datasource*. More information about configuring a datasource using Tomcat 5.5 can be found here:http://tomcat.apache.org/tomcat-5.5-doc/jndi-datasource-examples-howto.html

- **url**

  This is the connection string that the application would use to connect to the database. The example below will work fine as long as the database server is on the same machine as the application server and can by connected to as *localhost*. If the database server is not on the same machine as the application server, you will have to change the *localhost* part of the url, to the name, or IP address of the server the database resides on.

  MySQL, by default, runs on port 3306. If your database engine runs on a different port then the port part of the url will obviously have to be changed.

  The last part of the url before the '?' is the database name. Note that you must set jdbcCompliantTruncation=false.

  If you wish to have full support for unicode characters and the utf-8 character set then set useUnicode=true and set characterEncoding=UTF8

  E.g.

  *jdbc:mysql://localhost:3306/intralibrary?autoReconnect=true&jdbcCompliantTruncation=fa*

- **maxIdle**

This is the maximum number of idle database connections that you wish to have in the database connection pool. For unlimited connections, set to 0.

example: *100*

- **maxActive**

  This is the maximum number of database connections in the pool. Set to 0 for no limit.

  example: *500*

- **driverClassName**

  This is the fully qualified name of the JDBC Driver class. If you are using the database driver we have supplied then the example value below will be fine.

  example: *com.mysql.jdbc.Driver*

- **maxWait**

  This is the maximum time to wait for a database connection to become available in ms, in this example 10 seconds. To wait indefinitely set this value to -1. An error will occur if the application has to wait for a time exceeding this limit.

  example: *10000*

- **username**

  This is the username that the application will use to connect to the database. This user must have read and write access to the database whose name is defined in the database connection url defined above.

  example: *intrallect*

- **password**

  This is the password for the user defined above.

  example: *intrallect*

- **factory**

  Unless you are implementing a different database pool to the one supplied, leave this entry as is.

## 2.6. Copying Dependencies

NOTE: The following instructions are written specifically for using Tomcat 5.5.x.

The final part of configuring the *datasource* is to copy the following file from the install package *dataSourceLib* directory to *TOMCAT_HOME/common/lib*:

If you are prompted to overwrite any existing files, go ahead. You only live once!:

- mysql-connector-java-5.0.4-bin.jar
- commons-dbcp-1.2.1.jar
- commons-pool-1.3.jar
- commons-collections-3.2.jar

Although Tomcat already contains XML libraries, we need to make sure that the ones that are loaded are the ones that we need. To enable us to do this we make use of the ability to add jar files to an *endorsed* directory.

To achieve this you should copy the following files from the install package *endorsed* directory to *TOMCAT_HOME/common/endorsed*. Again if you are prompted to overwrite existing files then go ahead:

- xercesImpl.jar
- xalan.jar
- xml-apis.jar
- serializer.jar

Certain parts of intraLibrary have been split out into independent modules, to allow intraLibrary to access these jar files and their dependencies need to be available within the *common* directory.

To achieve this you should copy the following files from the install package *authenticationLib* directory to *TOMCAT_HOME/common/lib*. Again if you are prompted to overwrite existing files then go ahead:

- attica.jar
- commons-codec-1.3.jar
- commons-discovery-0.4.jar
- commons-id-0.1-dev.jar
- intralibrary-authentication.jar
- jdom.jar

- log4j-1.2.8.jar
- xercesImpl.jar
- xmlsec-1.4.0.jar

Some of the entry points to intraLibrary are protected by authentication as provided by the servlet container to allow this authentication to function correctly a custom realm must be provided.

To achieve this you should copy the following files from the install package *realmLib* directory to *TOMCAT_HOME/server/lib*. Again if you are prompted to overwrite existing files then go ahead:

- intralibrary-realm.jar

# 3. Database Setup

### 3.1. Creating the Database

We need to create a database for the application to use. We do this by executing the following sql statement from the MySQL command line:

***create database intralibrary;***

you do not need to use the name *intralibrary* but the following instructions will assume you have.

Next we need to create a user for the application to connect to the database as and give that user the correct permissions. Here we are using the username/password of intrallect/intrallect. This can be set to anything you like but must match the values that were used when setting up the datasource.

***grant all on intralibrary.* to intrallect identified by 'intrallect';***

***grant all on intralibrary.* to intrallect@'%' identified by 'intrallect';*** On some database installations the wildcard ('%') does not work when connecting to the database from a machine which resolves as localhost. You can leave out this step if you are confident you can.

***grant all on intralibrary.* to intrallect@'localhost' identified by 'intrallect';***

### 3.2. Populating the Database

We now need to create the database tables and populate these tables with the initial values that the application needs.

In the dbscripts folder, the main sql file that needs to be run to set up the database is: 'intralibrary.sql'.

There are a number of ways that this can be done. The easiest and most reliable is to connect to the database and from the database command line issue the following command:

***source*** *absolute/path/to/file;* e.g. source /home/intralibrary/dbscripts/intralibrary.sql;

obviously replacing 'absolute/path/to/file' with the actual file path!

If you wish to add support for UKLomCore and CANCore application profiles, you should also run the *uklomcore0p2.sql* and *cancore.sql* files in this directory.

### 3.3. Database Configuration

Although it is outside the scope of this documentation to go into a detailed discussion of database configuration, when using MySQL, the following advice may be useful.

MySQL configuration lives in a file named either *my.cnf* or *my.ini*. Its name and location depends on the installed environment. The following parameters (and numerous others) can be set in this file and effect performance of intraLibrary and should be investigated. We've included some sample values which we've found to work well but your environment may differ. For more information see here:http://www.mysql.com/doc/en/MySQL_Database_Administration.html

- set-variable = max_connections=100000
- set-variable = max_allowed_packet=20M
- set-variable = wait_timeout=360

**Note:**It is essential that the value for max_connections does not exceed the max_connections variable setup in the my.cnf(my.ini) file for the MySQL server. If you are unfamiliar with this configuration file look here:http://www.mysql.com/doc/en/MySQL_Database_Administration.html

IntraLibrary will not operate if the MySQL server has been configured with the STRICT_TRANS_TABLE mode switched on. To switch this mode off, remove any reference to STRICT_TRANS_TABLE for the configuration file and restart MySQL.

# 4. Confirming Installation

We should now be ready to restart the application server and confirm that the intraLibrary application is running.

The first thing we need to do is to make sure we can see or read the output from the application server. Depending on how you set up the *context*, this output could go to one of a few places. By default, Tomcat writes all its logging to files inside *TOMCAT_HOME/logs*. If you have used the supplied *sample_context.xml* file then the output file you should be interested in is *catalina.out*. [Note: on Windows systems this output file does not exist: output is written to the Tomcat console window.]

Before you start the server make sure you have included the licence file in the correct place.

If you start the server now, and your logging level is set to the correct details, you should see some output to the log file that resembles the text below.

```
****************************************

populating learning object cache with 0 objects...
finished populating learning object cache

****************************************
application version: n.n (nnnnn)
database server version:5.0.27-community-nt
```

```
database url:jdbc:mysql://localhost:3306/intralibrary?autoReconnect=true&jdbcCompliantTruncation
database driver:MySQL-AB JDBC Driver
max database connections(0=unlimited):0
max upload file size:49Mb
servlet container:Apache Tomcat/5.5.20
java version:1.5.0_10
---------------------------------------
configDir location:/Volumes/projects/intralibrary/config
http proxy:no proxy used
mail host:mail.company.com
admin email:admin@company.com
error email:errors@company.com
return email:intralibrary1@company.com
id prefix:com.company.intralibrary
repository name:intralibrary
object cache size:250
entry page:browse
debug:false
reindexOnStartup:true
---------------------------------------
total memory assigned:26Mb
maximum memory assigned:112Mb
---------------------------------------
licence assigned to:company
max contributors:10
current contributors:1
expires:
---------------------------------------
IntraLibrary started dd-MMM-yyyy hh:nn:ss
*****************************************
```

Assuming that you have no errors in this output then congratulations, you have successfully installed intraLibrary 3.3.

If you don't see the output below then try browsing, through a web browser, to the newly defined context. If you see the intraLibrary login page then everything has been installed successfully

Now that you have a successful implementation of intraLibrary you can login using the username/password of intrallect/intrallect

# 6. Setting the Application URL

Once the installation has been completed you should log in and set the application URL, this value is configured within the system section of the admin area (under 'System Options' tab) in the web interface.

The application URL determines the base url of the application e.g. http://localhost:8080/intralibrary and is used within the application when construction URLs are exposed outwith the web application e.g. URLs within SRU search results, RSS feeds, harvested result sets etc.

If this value is not set then intraLibrary's external facing interfaces will not be fully functional