User Manual

v2.3

1 Introduction

LogFaces is a standalone solution for collecting, storing, dispatching and filtering log data. LogFaces is a "box" which resides next to your system and lifts the logging burden off you. Built one level above the conventional logging API's such as Apache Logging Services it easily integrates with existing systems, including those built on J2EE, native C++, or Microsoft© .NET Framework. If your system is already using those API's, you can use logFaces for storing log, real-time log monitoring, re-evaluating operational discrepancies, receiving scheduled email reports with log files and more...

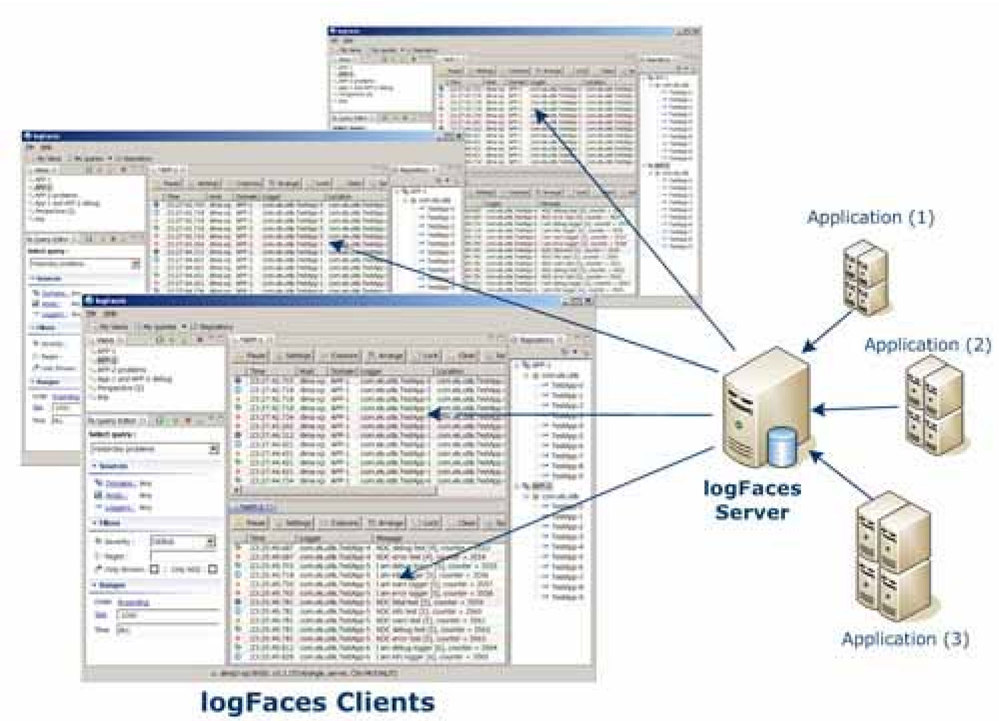


Figure 1-1 LogFaces Architecture

There are three players - your system, logFaces Server and logFaces Client. It works like this: your system/s send log data to the logFaces Server which routes this data to interested clients or **does something with this data on its way**. For example; saving some of it to database or discarding it, splitting and filtering it for various client listeners. The only requirement for any system to be operable with logFaces is to produce log in a format understood by the logFaces Server which collects this log. The current version of logFaces supports Apache log4xxx API's as schema of log statements. But technically logFaces will not be limited only to those formats. Today log4xxx API's are the de-facto standards in software development and most systems already support them. In such cases the integration with logFaces is straightforward and is only a matter of simple configuration.

LogFaces can be used in a compact mode (or **Server Mode** as we call it) where applications send log data directly to the Client. This mode is mostly dedicated for those who want to use logFaces as a real time viewer for single user. It fits better smaller projects due to its simplicity, however it doesn't have persistence capabilities, reporting and data mining features – it's a plain real time viewer of log data.

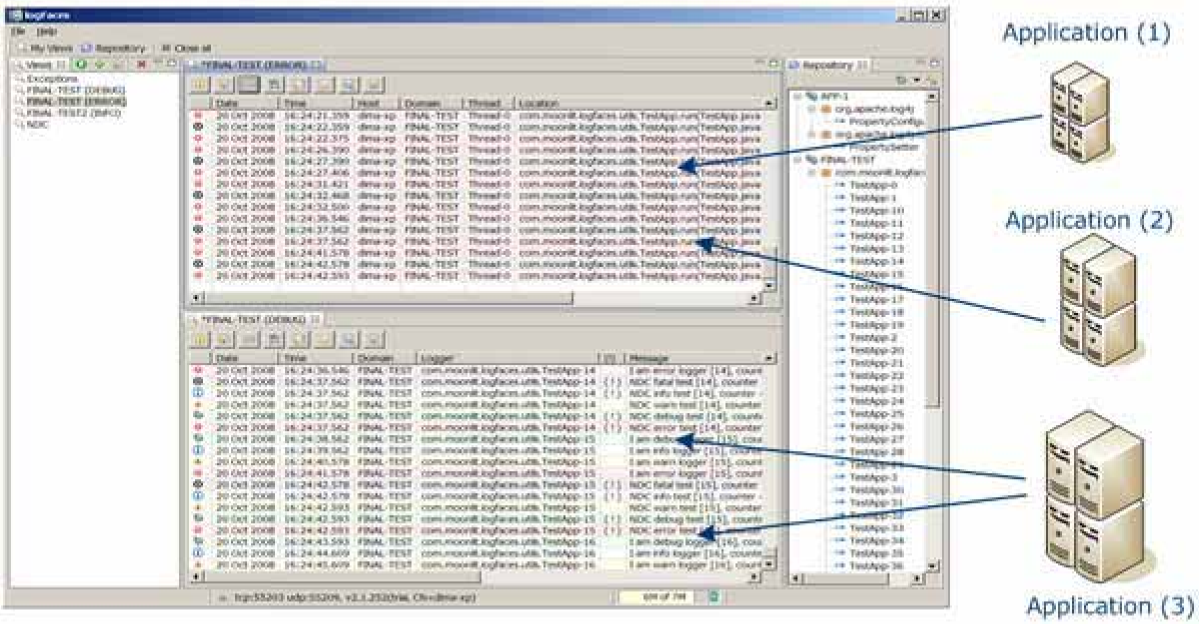


Figure 1-2 logFaces Client in server mode

2 Getting started with logFaces Server

This section will guide you through the quick process of installation and configuration. Java™ Runtime Environment (JRE) 1.5 or later must be installed before proceeding with installation either on Windows or on Linux.

2.1 Installation on Windows

Windows installation comes as **lfs-server.exe** windows installer, run it and simply follow the instructions. LogFaces Server is installed by default as Win32 Service thus you will have to have administrator privileges on this computer during installation. Every time your computer is restarted, logFaces Server will run. The service name is **LFS**; you should see it in the list of installed services on your machine. Installation will create two shortcuts in your Start menu – a link to administration console and link to uninstaller.

LogFaces Server will automatically start after the installation with default settings and trial license for **30 days evaluation**.

If you would like to manually stop/start logFaces Server you should do "**net stop lfs**" or "**net start lfs**" commands from the command prompt, or control the service from Windows control panel.

If you don't like to have logFaces Server running as a service, you can run it as regular console application manually by executing a batch file in \bin directory called **lfs.bat**.

After installation you would normally want to configure the server for your environment. Go to the link created by installer in a Start menu, by default it points to **http://localhost:8050/admin.** See section called "**Administration**" for more details.

2.2 Installation on Linux

Linux installation comes as an executable jar file - **lfs-server.jar**; if your system doesn't have an association to run executable jars, then open terminal and run it manually: **java –jar lfs-server.jar.** Otherwise simply execute the downloaded **lfs-server.jar**. In both cases make sure to specify executable permissions for **lfs-server.jar** before launching it. After installer starts, simply follow the instructions.

Bin directory under the installation will contain shell scrip named **lfs** which is used to control the server process. Open terminal and navigate to bin directory.

- In order to start the server as separate process execute command **lfs start**
- In order to start the server in terminal execute command **lfs console**
- In order to stop the server process execute command **lfs stop**
- In order to check the server process status execute command **lfs status**

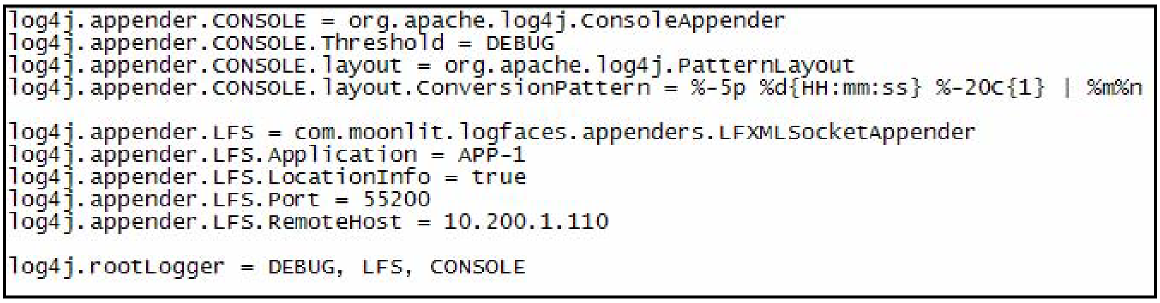
LogFaces Server will automatically start with default settings and trial license for **30 days evaluation**. After installation you would normally want to configure the server for your environment. Open your browser and navigate to **http://localhost:8050/admin.** See section called "**Administration**" for more details.

2.3 Integrating with applications

To work with logFaces, your application needs to be configured by adding several elements. Provided that your system is based on log4xxx API's, you should be having log4xxx configuration file/s, which usually come as property files or XML configuration files. Basically, we add an appender to your configuration file which knows how to communicate with logFaces Server.

2.3.1 Configuring Java™ applications

To get your Java application talk to logFaces, we add an appender to your log4j.properties file; see the appender called LFS in the example below:



```
log4j.appender.CONSOLE = org.apache.log4j.ConsoleAppender
log4j.appender.CONSOLE.Threshold = DEBUG
log4j.appender.CONSOLE.layout = org.apache.log4j.PatternLayout
log4j.appender.CONSOLE.layout.ConversionPattern = %-5p %d{HH:mm:ss} %-20C{1} | %m%n

log4j.appender.LFS = com.moonlit.logfaces.appenders.LFXMLSocketAppender
log4j.appender.LFS.Application = APP-1
log4j.appender.LFS.LocationInfo = true
log4j.appender.LFS.Port = 55200
log4j.appender.LFS.RemoteHost = 10.200.1.110

log4j.rootLogger = DEBUG, LFS, CONSOLE
```

Let's have a closer look into the properties:

log4j.appender.LFS = com.logFaces.appenders.LogFacesXmlSocketAppener

- This specifies the class of the appender - LogFacesXmlSocketAppender which is located in \lib\lfssappenders.jar file and also available separately for download from the site. You should copy this jar to your application's class path.

log4j.appender.LFS.Application = XXX

- This is so called "logging domain" name. It will identify your application under this name. logFaces will group all loggers related to the same domain so that you will be able to distinguish information coming from several applications which might be identical, or in the other words, different instances of the same application. Log4j configuration file is working for a single JVM instance, but having "Application" attribute will allow us to group several JVMs (on one or several computer) under the same name.

log4j.appender.LFS.LocationInfo = true/false

- Here you specify if logging information should contain location data, such as class name, method name and line numbers.

log4j.appender.LFS.RemoteHost = ip_address (host name)

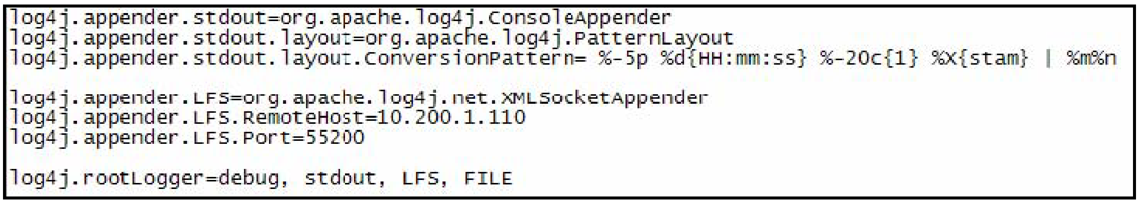
- This is the actual IP address or a host name of the computer where logFaces Server is running.

log4j.appender.LFS.Port = 55200

- logFaces Server listens on several ports for application. By default port 55200 is dedicated for TCP communication, but of course this can be modified to anything you like – use Administration Console to change it.

2.3.2 Configuring C++ applications

If your system is based on C++ and using Apache Log4cxx API for logging, you should configure it by adding XMLSocketAppender included in log4cxx API itself. Here is a snippet of configuration example:

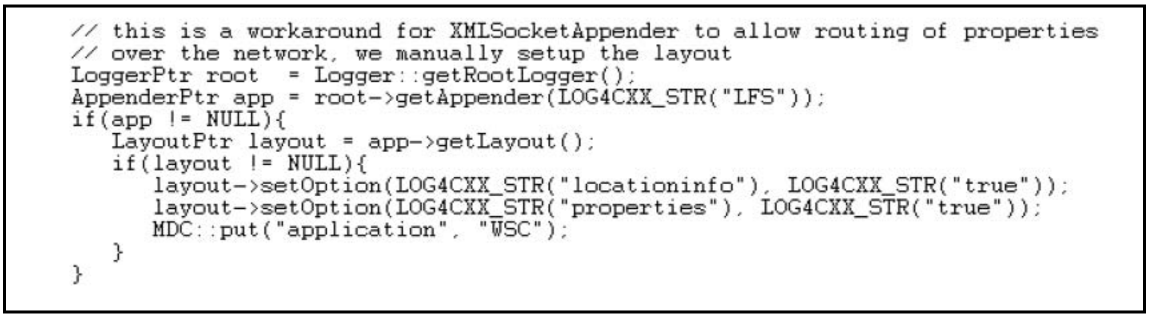


```
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern= %-5p %d{HH:mm:ss} %-20c{1} %X{stam} | %m%n

log4j.appender.LFS=org.apache.log4j.net.XMLSocketAppender
log4j.appender.LFS.RemoteHost=10.200.1.110
log4j.appender.LFS.Port=55200

log4j.rootLogger=debug, stdout, LFS, FILE
```

The meaning of the attributes is identical to the previous Java™ example. However, note that XMLSocketAppender doesn't (yet?) provide "Application" and "LocationInfo" attributes. This is not a problem for logFaces – those loggers which don't correspond to any logging domain will be automatically grouped in logFaces under name "**Default Domain**". Unfortunately, until those attributes are supported by the underlying APIs, we will have to add some code when initializing the logger in the application. The code snippet is shown below, what we do is simply getting into a root logger, digging out the LFS appender from there and manually set the missing attributes of the layout like this:



```
// this is a workaround for XMLSocketAppender to allow routing of properties
// over the network, we manually setup the layout
LoggerPtr root  = Logger::getRootLogger();
AppenderPtr app = root->getAppender(LOG4CXX_STR("LFS"));
if(app != NULL){
   LayoutPtr layout = app->getLayout();
   if(layout != NULL){
      layout->setOption(LOG4CXX_STR("locationinfo"), LOG4CXX_STR("true"));
      layout->setOption(LOG4CXX_STR("properties"), LOG4CXX_STR("true"));
      MDC::put("application", "WSC");
   }
}
```

In any case, those missing attributes are not a show stoppers, your application can still work with logFaces out of the box with those limitations.

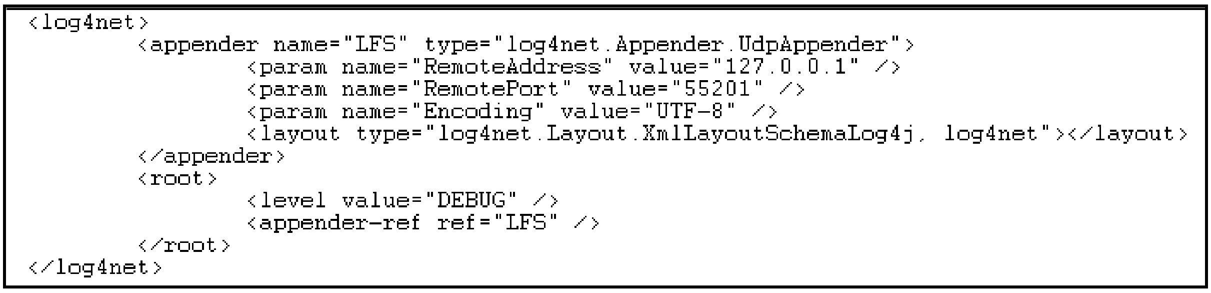
IMPORTANT:

> The MDC (message diagnostic context) works only in the context of the current thread. In case you have several threads in your application you should add **MDC::put("application", "xxx")** call in the beginning of every thread. Otherwise, the log statements coming from those threads will be orphaned and server will automatically put them under "Default Domain" which might be a bit confusing.

Future versions of logFaces will include proper appender to avoid those workarounds.

2.3.3 Configuring .Net applications

If your system is .Net based and using Apache log4net API for logging, you should configure it by adding UdpAppender. Here is a snippet of configuration example:



```
<log4net>
        <appender name="LFS" type="log4net.Appender.UdpAppender">
                <param name="RemoteAddress" value="127.0.0.1" />
                <param name="RemotePort" value="55201" />
                <param name="Encoding" value="UTF-8" />
                <layout type="log4net.Layout.XmlLayoutSchemaLog4j, log4net"></layout>
        </appender>
        <root>
                <level value="DEBUG" />
                <appender-ref ref="LFS" />
        </root>
</log4net>
```

As mentioned earlier, logFaces can listen for TCP and/or UDP. In this example, we use UDP appender and you should make sure that RemotePort attribute in this example corresponds to the one configure in logFaces, which of course can be modified using logFaces Administration Console.

2.4 Administration

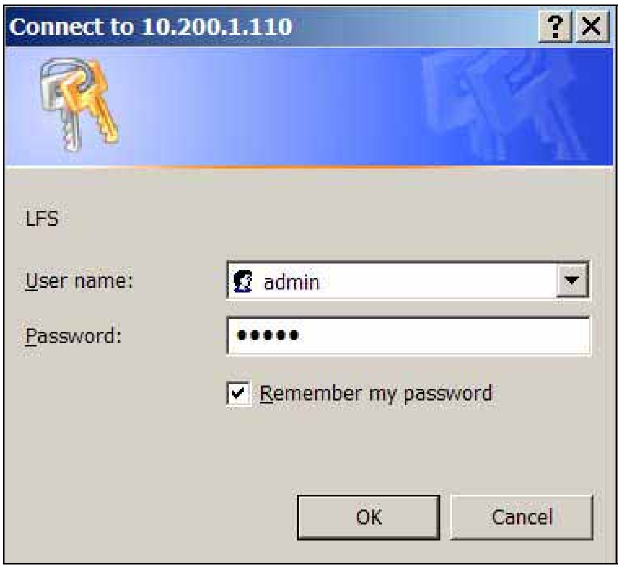
Most of logFaces Server configuration is done remotely using conventional browser.

To access Administration Console, open your browser and navigate to this URL:

http://localhost:8050/admin

The shortcut is also created for you during the installation in Windows Start menu. Of course, you can access the console from any other host as long as you network configuration allows the access.

Access to logFaces Administration Console is secured by user name and password, which are both defaulted to "**admin**" during installation:



User name: admin

Password: •••••

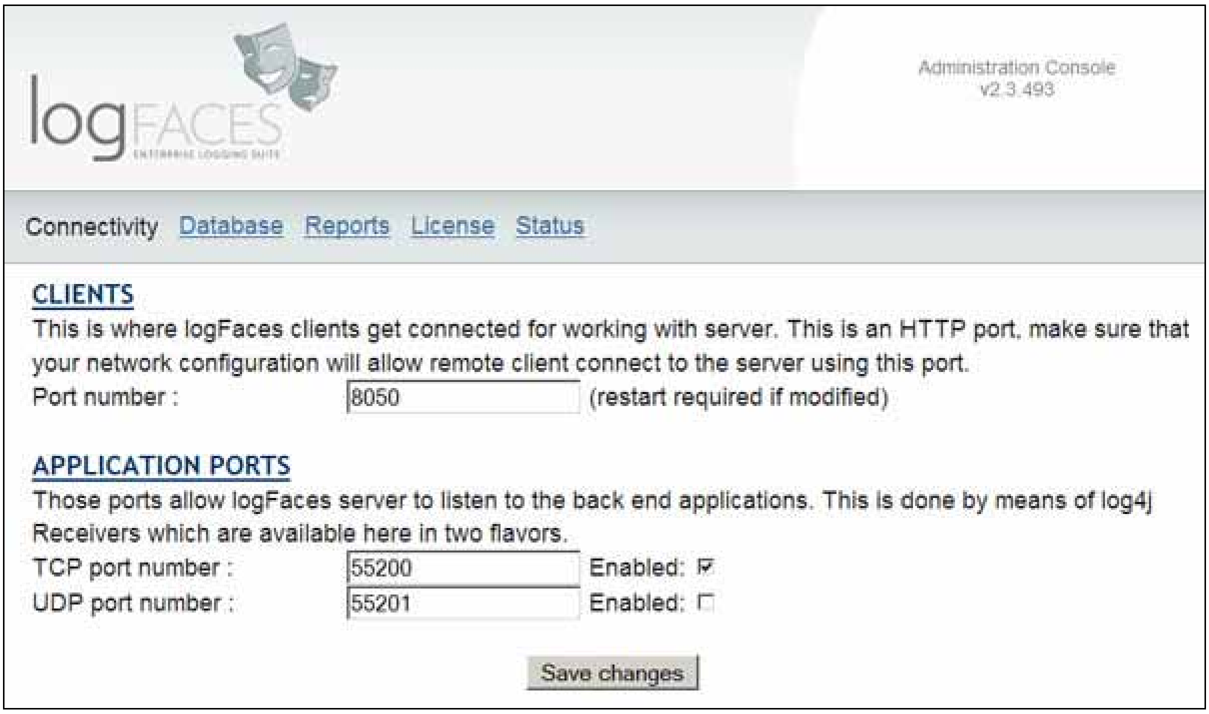
☑ Remember my password

If you want to modify user name and/or password for the future use, see **Advanced configuration "how to"** for more details.

Administration Console is grouped into several tabs where you can easily modify needed functionality – connectivity, database, reporting, licensing and status. Let's go through them in details.

2.4.1 Connectivity

Here you specify external and internal ports where Server will be listening.



CLIENTS

This is where logFaces clients get connected for working with server. This is an HTTP port, make sure that your network configuration will allow remote client connect to the server using this port.

Port number : 8050 (restart required if modified)

APPLICATION PORTS

Those ports allow logFaces server to listen to the back end applications. This is done by means of log4j Receivers which are available here in two flavors.

TCP port number : 55200 Enabled: ☑

UDP port number : 55201 Enabled: ☐

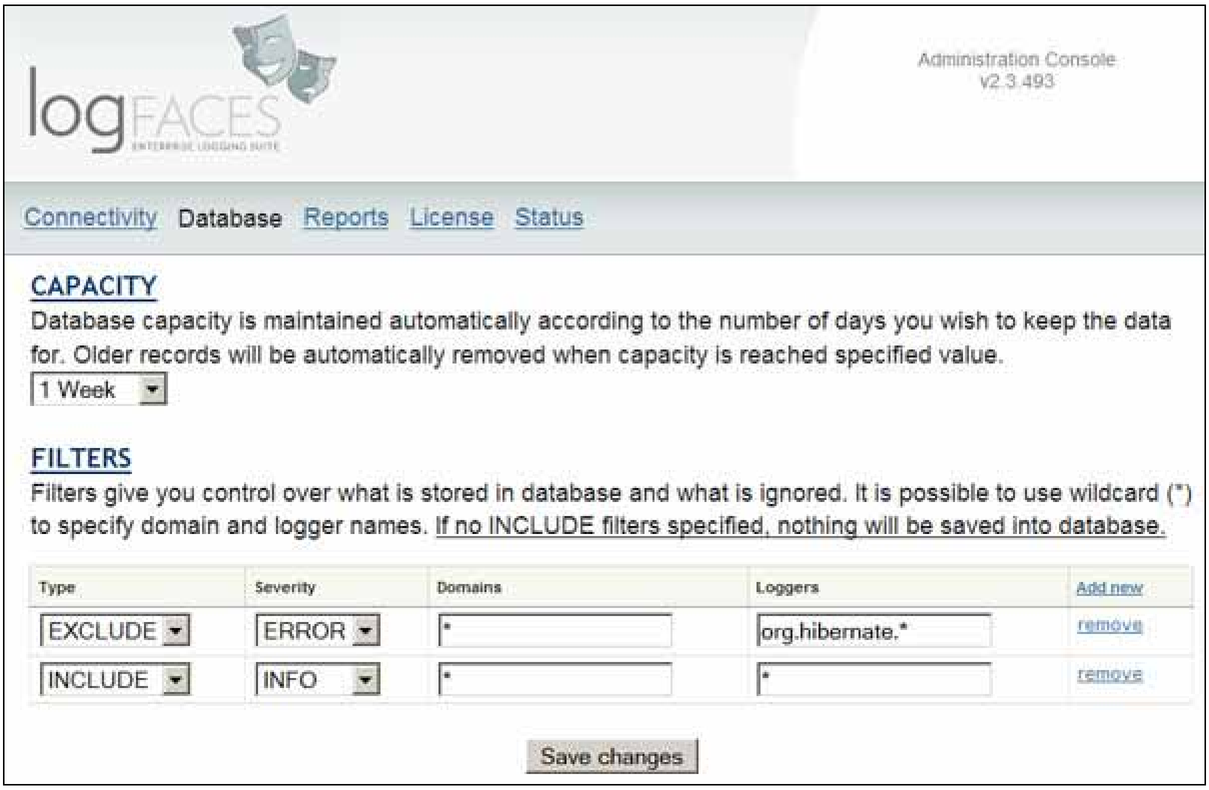
Figure 2-1 Connectivity panel

External port defaulted to 8050 is for Clients and Administration Console connections. This is HTTP connection port which you can modify to fit your environment, but Server will have to be restarted for the changes to take affect.

Internal ports are for the applications to log into logFaces. There are two ports available – TCP and UDP which you can modify or enable/disable. **Changes are effective immediately**. Make sure your applications appenders are adjusted to meet those changes. Socket appender port property named **log4j.appender.LFS.Port** in your configuration should match the ones specified here.

2.4.2 Database

In Database section we specify how logFaces Server database should operate. There are two major things we need to specify – **how much** log to keep and **what** log events should be saved in database (if any).



CAPACITY

Database capacity is maintained automatically according to the number of days you wish to keep the data for. Older records will be automatically removed when capacity is reached specified value.

1 Week

FILTERS

Filters give you control over what is stored in database and what is ignored. It is possible to use wildcard (*) to specify domain and logger names. If no INCLUDE filters specified, nothing will be saved into database.

Type	Severity	Domains	Loggers
EXCLUDE	ERROR	*	org.hibernate.*
INCLUDE	INFO	*	*

Figure 2-2 Database panel

Database **capacity** is specified in days of log. If you specify "1 week" for example, then at least one week of data will always be available. As time goes, older records are automatically removed while new ones are appended. You should carefully specify this value according your needs; it affects overall performance as well as disk space usage.

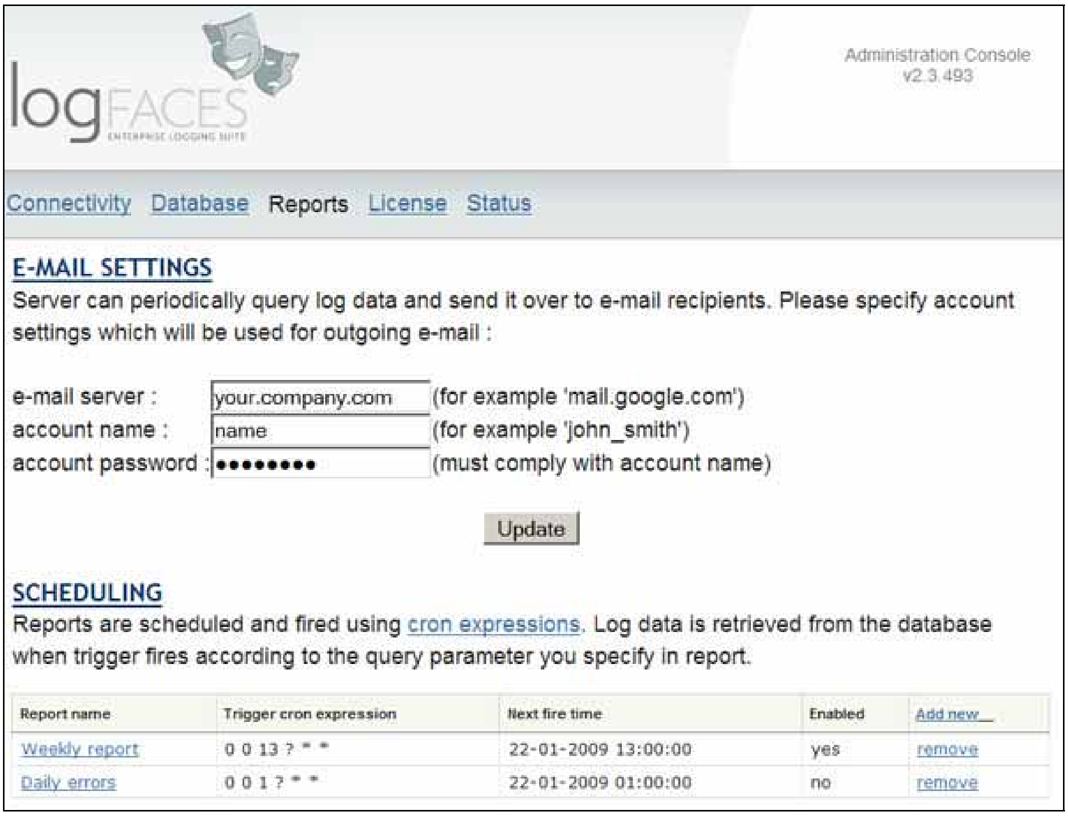
Filters let you specify which logging events to be stored in database. Normally we don't need debug information to be floating around and bloating our server, but sometimes we do need it too. There are two types of filters you can specify – **INCLUDE** and **EXCLUDE**. Filtering is done by **severity** of logging events going through logFaces Server, the **domain** (application) of the events and the **logger names**. You can use wild cards to specify names. The example above means that we want to include ALL events whose severities are higher (and including) INFO, but we don't want events which come from hibernate packages, except those which are higher (and including) ERROR. So, even though we specified the exclude filter, we still want problems from hibernate packages to be saved. You can create fairly complicated filters here; the order of the filters is not important.

If you would like to disable database persistence, simply remove all INCLUDE filters; this will ensure that nothing is saved.

2.4.3 Reports

Reports are custom log files that server generates according to schedule and rules you specify. Reports are sent by email to the recipients you specify. When defining report we need to specify what log information we need, the scheduling rule to trigger the report creation and who to dispatch the email to.

logFaces Server can handle unlimited amount of reports based on complex scheduling rules and content. Let's see what needs to be done to setup the reporting.



E-MAIL SETTINGS

Server can periodically query log data and send it over to e-mail recipients. Please specify account settings which will be used for outgoing e-mail :

e-mail server : your.company.com (for example 'mail.google.com')
account name : name (for example 'john_smith')
account password : ●●●●●●●●● (must comply with account name)

SCHEDULING

Reports are scheduled and fired using cron expressions. Log data is retrieved from the database when trigger fires according to the query parameter you specify in report.

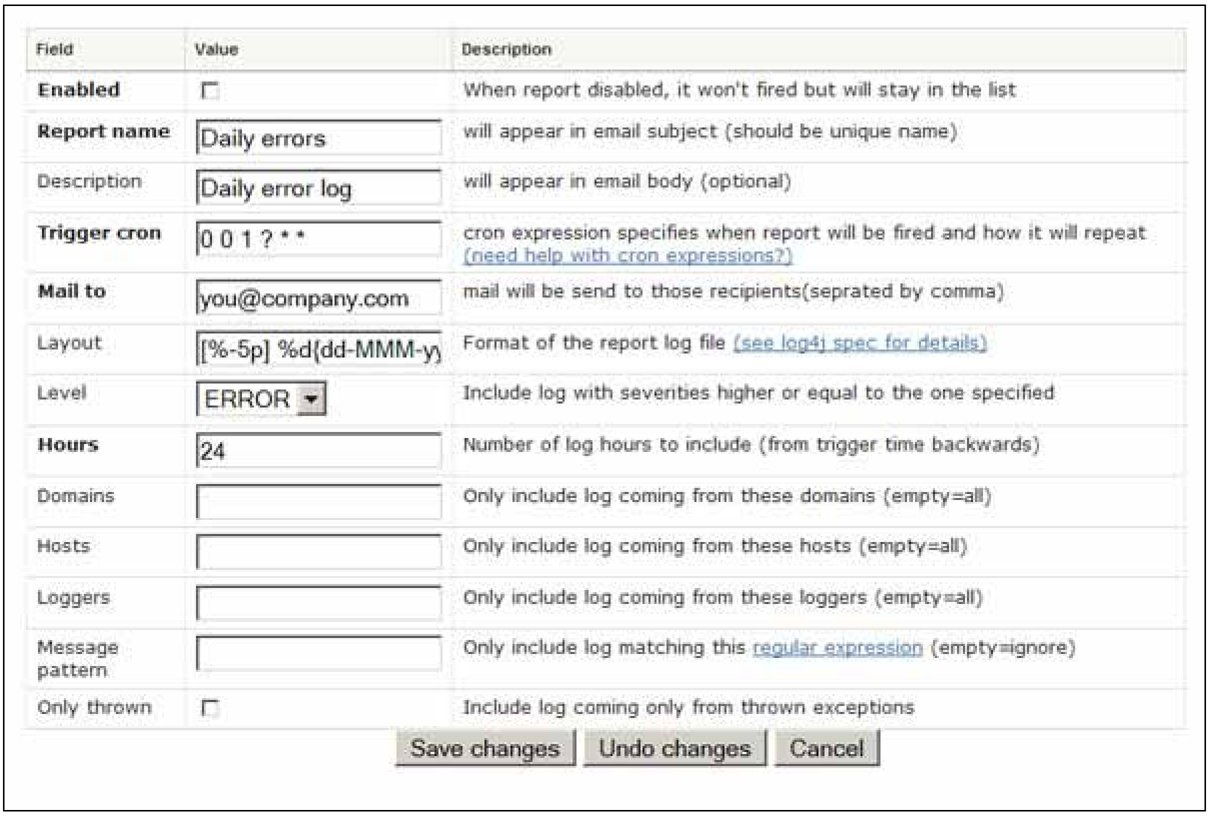
Report name	Trigger cron expression	Next fire time	Enabled	Add new
Weekly report	0 0 13 ? * *	22-01-2009 13:00:00	yes	remove
Daily errors	0 0 1 ? * *	22-01-2009 01:00:00	no	remove

Figure 2-3 Reports panel

First of all, to be able to send emails logFaces Server needs to know **outgoing email** account settings. This includes email server **host**, email **account** and **password**. Please make sure you specify those settings correctly and validate them before use. If those settings are incorrect, the reports won't be emailed, but you will be able to figure out the reason in logFaces internal log which we will show later.

Scheduling section lists all currently defined reports. As you can see, reports are named, have cron expression with next fire time stamp and an indication if report is enabled. Having report enabled or disabled is handy when you want too keep report definitions but don't want it to fire. When you remove the report from the list, it is permanently deleted and can't be recovered. However, it's not a big deal to create one anew – just click on the "Add new" link at the right top header of the list and it will create new dummy records which you can populate and enable by clicking on the report name.

Below is an example of report details. The fields in the bold are mandatory and must be specified.



Field	Value	Description
Enabled	☐	When report disabled, it won't fired but will stay in the list
Report name	Daily errors	will appear in email subject (should be unique name)
Description	Daily error log	will appear in email body (optional)
Trigger cron	0 0 1 ? * *	cron expression specifies when report will be fired and how it will repeat (need help with cron expressions?)
Mail to	you@company.com	mail will be send to those recipients(seprated by comma)
Layout	[%-5p] %d{dd-MMM-y	Format of the report log file (see log4j spec for details)
Level	ERROR	Include log with severities higher or equal to the one specified
Hours	24	Number of log hours to include (from trigger time backwards)
Domains		Only include log coming from these domains (empty=all)
Hosts		Only include log coming from these hosts (empty=all)
Loggers		Only include log coming from these loggers (empty=all)
Message pattern		Only include log matching this regular expression (empty=ignore)
Only thrown	☐	Include log coming only from thrown exceptions

Figure 2-4 Report details

- **Enabled -** when unchecked, the report will stay in the system but will never fire. You can enable it any time and it will fire at the next schedule.

- **Report name** – reports are uniquely identified by name, in fact each report is a scheduling job which is referred by the scheduling system by this name. Name of the report will appear in the subject of the email.

- **Description** – some text you might want to include in the email body. This is an optional field.

- **Trigger cron** – is an expression which specifies when and how to fire the report. Cron expressions are very flexible and used to make fairly complex scheduling schemas. In the picture example the expression means "fire every day at 1AM", but you can easily specify something like "fire on 10AM on the third Friday of the month". You can get more information about cron expression online, for example here.

- **Layout** – specifies how to layout the text in the log files. LogFaces is using log4j formatting rules; you can find more details here.

- **Level** – allows cutting down the severity of log we want in our report. For example, if you specify ERROR, then the report will only include ERROR and FATAL messages while DEBUG, INFO and WARN will be ignored.

- **Hours** – is the amount of hours the log data should cover since the time report is fired. For example, if report is fired at 1AM and amount of hours is set to 4, then the report will include log starting from 9PM of the previous day until 1AM of the current day.

- **Domains** –specify what logging domain (application) should be included in report. You may use wildcards here. If this field is left blank, then log from ANY domain will be included in the report.

- **Hosts** – specify from what hosts we need the log. You may use wildcards here. If this field is left blank, then log from ANY host will be included in the report.

- **Loggers** – specify from what loggers we need the log. You may use wildcards here. If this field is left blank, then log from ANY logger will be included in the report.

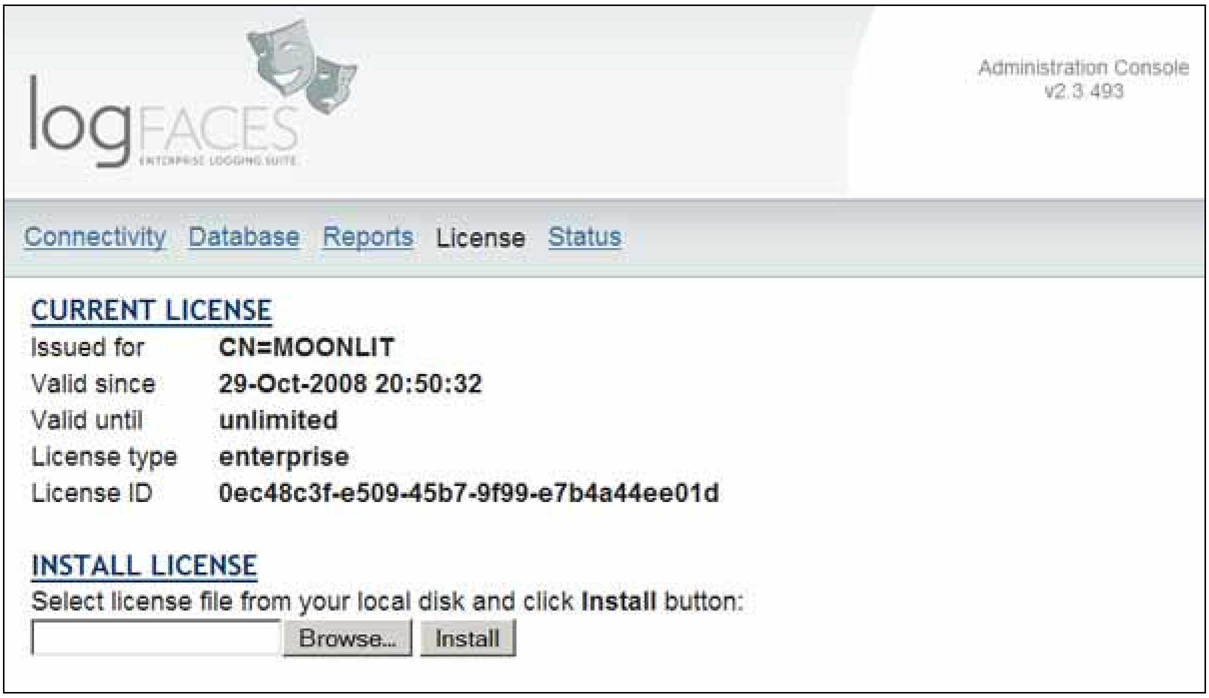
- **Match text** – is the regular expression pattern which allows matching of the log messages. If this field is left blank, the matching will not be applied, but when this field contains a pattern, the logFaces Server will inspect every fetched log event from database for matching this pattern. Regular expressions are a very powerful; you can learn more about them here.

- **Only Thrown** – when checked, the report will only contain log events which are thrown exception.

After you finish specifying the report and click "**Save Changes**", the last thing to do is to "**Save and Reschedule**" which becomes visible every time you modify something in the report list.

2.4.4 License

License tab displays currently installed license information as well as allows you to install new license file. When you install logFaces Server for a first time, it automatically activates one time trial evaluation period for 30 days. If you decide to purchase a license, the license file should be submitted through this form:



CURRENT LICENSE

Issued for CN=MOONLIT
Valid since 29-Oct-2008 20:50:32
Valid until unlimited
License type enterprise
License ID 0ec48c3f-e509-45b7-9f99-e7b4a44ee01d

INSTALL LICENSE

Select license file from your local disk and click **Install** button:

Figure 2-5 License Panel

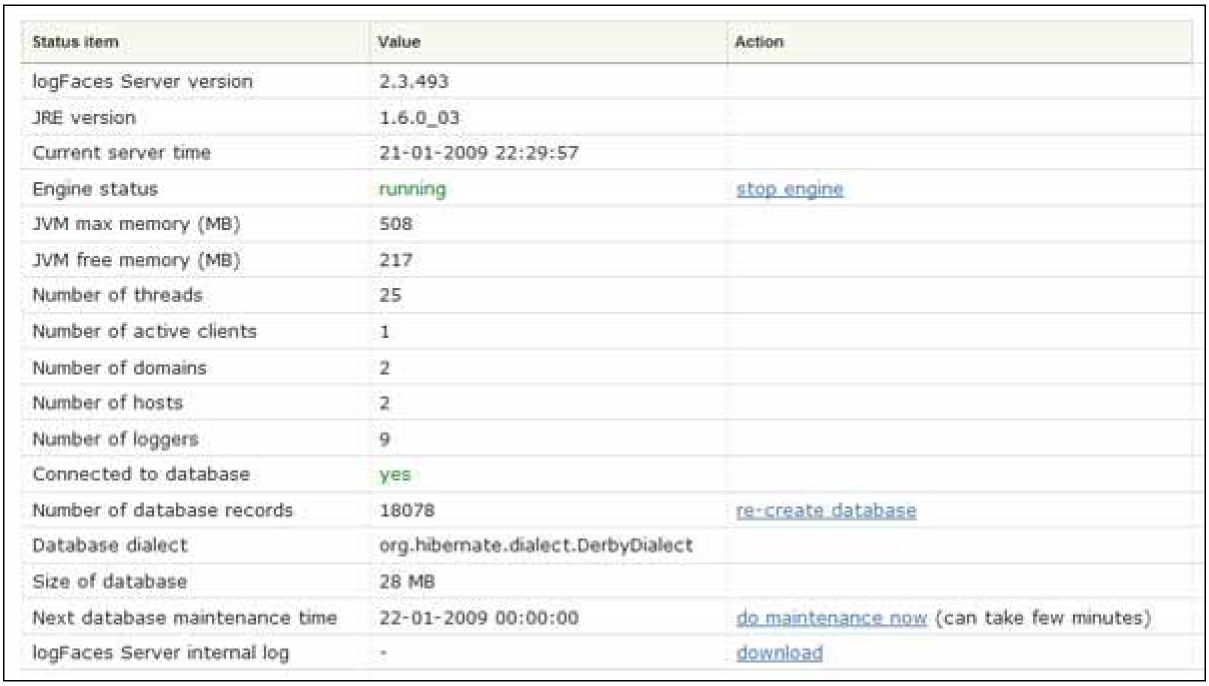
What happens when evaluation license expires?

LogFaces Server will shutdown its engine and only allow Administration Console access; applications will not be able to append to logFaces and clients won't be able to connect to it.

When you install new license, the engine should be started manually. This can be done by simply restarting the whole service from command prompt or control panel, or from the Status panel link – see the next section.

2.4.5 Status

Status tab contains useful health monitoring information of the Server and allows basic instrumentation tasks.



Status item	Value	Action
logFaces Server version	2.3.493	
JRE version	1.6.0_03	
Current server time	21-01-2009 22:29:57	
Engine status	running	stop engine
JVM max memory (MB)	508	
JVM free memory (MB)	217	
Number of threads	25	
Number of active clients	1	
Number of domains	2	
Number of hosts	2	
Number of loggers	9	
Connected to database	yes	
Number of database records	18078	re-create database
Database dialect	org.hibernate.dialect.DerbyDialect	
Size of database	28 MB	
Next database maintenance time	22-01-2009 00:00:00	do maintenance now (can take few minutes)
logFaces Server internal log	-	download

Figure 2-6 Status panel

Most of the information in this tab should be self explanatory for technical people. We will just mention the important instrumentation actions which are available from the links on the right side of the table:

- **Engine start/stop;** sometimes it's required to put the server down without actually shutting the process down. One of the typical uses of this option is when trial license expires. In such case, the logFaces Server will start so that you would be able install proper license, but its engine will be down and no logging will be taken from applications.

- **Re-create database** allows to remove all database records; be careful with this operation, it is not recoverable and can't be undone

- **Database maintenance** allows compacting (compressing) the data and rebuilding the indexes. This action is only available when you use embedded database. By default this operation is performed **every midnight**, but sometimes we want to do it manually. Maintenance can often take several minutes (it depends on the size of the database and speed of the computer), please note that during maintenance database is locked – no commits and queries will go through until operation is complete.

- **Download internal log** allows fetching the log produced by the logFaces itself. In case you experience problems, this information will be very valuable to provide support. When you click on the **download** link, the server will create internal log file and prompt you to save or open it in text editor. This file can be sent to our support team.

2.5 Advanced configuration "how to"

There are some things which can not be configured through the Administration Console and need to be done manually.

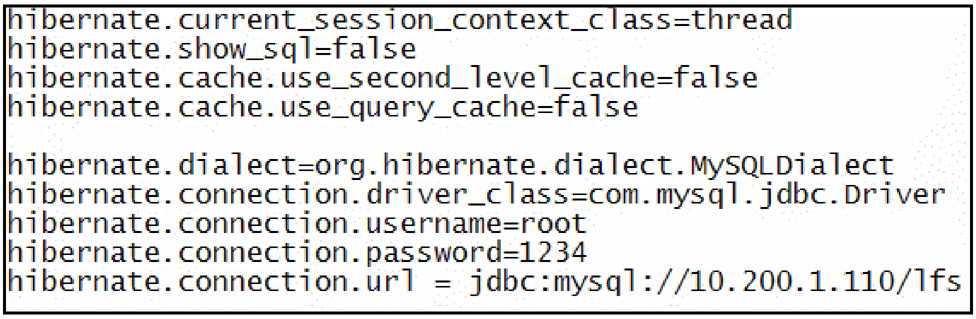
2.5.1 How do I change user name and password for administration login?

- Stop the Server
- Open /conf/realm.properties file in text editor. There is only a single record in this file which has the following format: [userName: password, admin]. Change first two fields as you like
- Start the Server; you should be able to login into the Console with new credentials now.

2.5.2 How do I work with external databases?

LogFaces Server by default is using embedded database (Apache Derby). Although its performance is adequate for most environments, there is an option to use an external database. Version 2.x was tested to work with Oracle, MySQL and PostgreSQL databases. If you already have one of those databases in your disposal and want to reuse it with logFaces all you need to do is to replace one of the configuration files and adjust to the settings your database requires.

LogFaces Server uses Hibernate ORM framework; it is recommended to have a basic knowledge of its configuration before you decide to make reconfiguration. Everything is configured in **/conf/hibernate.properties** file. To make life easier, \conf directory also contains examples of other property files for each of the supported databases. What you need to do is to replace the content of **/conf/hibernate.properties** file with the corresponding database settings. You can also rename the specific file into **hibernate.properties**, but make sure to keep the backup of the original just in case. For example, MySQL configuration looks like this:

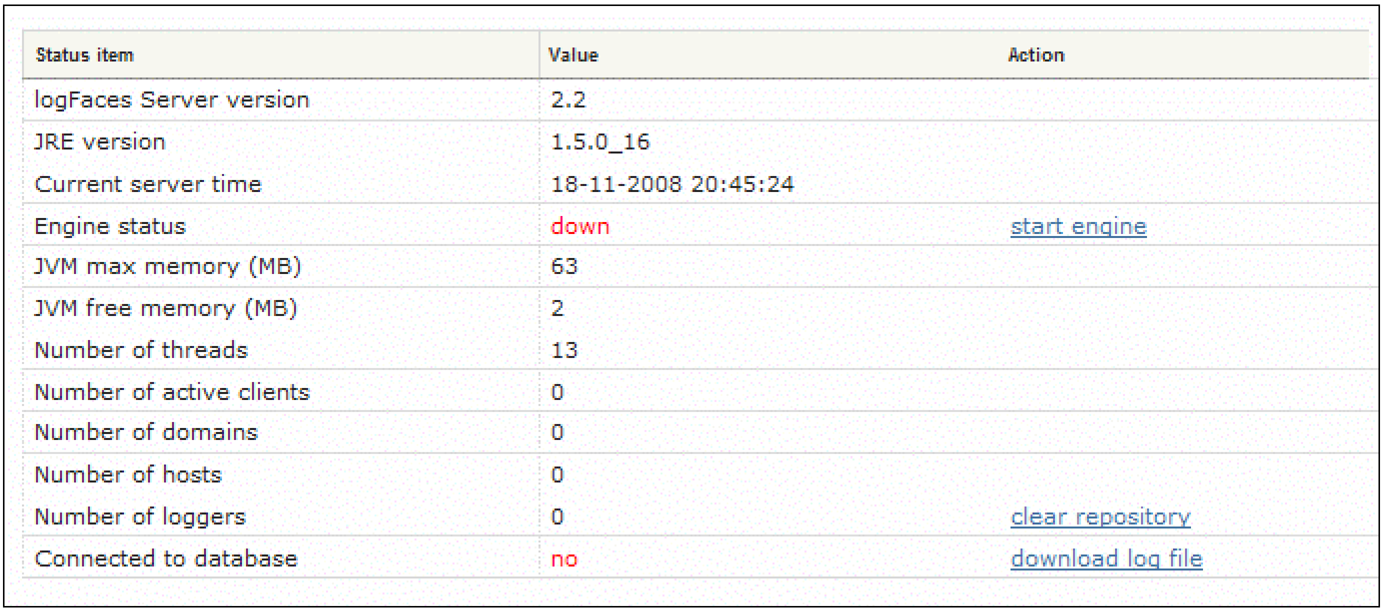


```
hibernate.current_session_context_class=thread
hibernate.show_sql=false
hibernate.cache.use_second_level_cache=false
hibernate.cache.use_query_cache=false

hibernate.dialect=org.hibernate.dialect.MySQLDialect
hibernate.connection.driver_class=com.mysql.jdbc.Driver
hibernate.connection.username=root
hibernate.connection.password=1234
hibernate.connection.url = jdbc:mysql://10.200.1.110/lfs
```

Installation also contains the latest tested database drivers in **/lib/dbdrivers** directory for the databases we currently support, so all you need is to specify the dialect, driver, user name, password and connectivity URL.

Once you prepared the configuration, simply restart the logFaces Server, this can be done either from services control panel of windows or by typing "**net stop lfs**" and "**net start lfs**" in command prompt. During the startup it will publish new schema automatically. After the startup open Administration Console and navigate to the **Status** tab. If everything went well, you should see that engine is started, database connection is ok and there are some database related items in the table. If something goes wrong and there are problems with database connection, you will see the status like this:

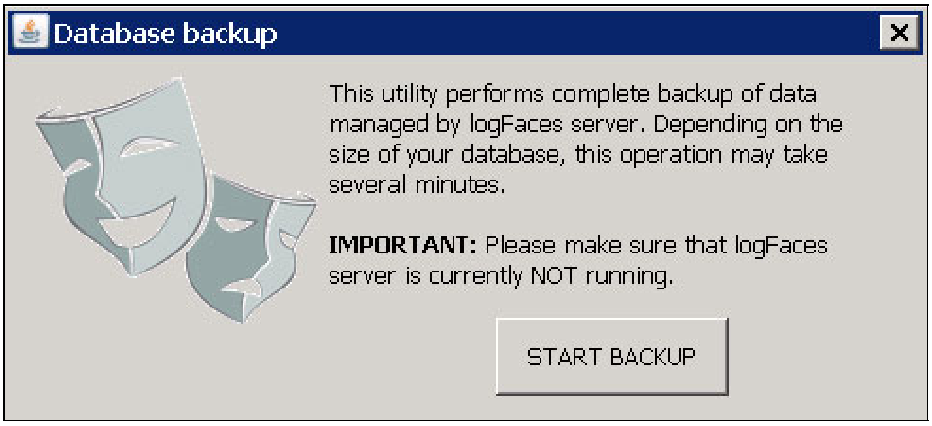


Status item	Value	Action
logFaces Server version	2.2	
JRE version	1.5.0_16	
Current server time	18-11-2008 20:45:24	
Engine status	down	start engine
JVM max memory (MB)	63	
JVM free memory (MB)	2	
Number of threads	13	
Number of active clients	0	
Number of domains	0	
Number of hosts	0	
Number of loggers	0	clear repository
Connected to database	no	download log file

To see what happened, click on "download log file" link; this will download the internal logFaces log file to your desktop. Normally, this log should not have any exceptions or errors. The problem usually is related to some configuration error, typo or perhaps your database is not responding as logFaces expects. If you're unable to figure out the problem yourself, submit this log file to our support site and we will try to help.

2.5.3 How do I backup my database storage?

While server is stopped, run backup utility located in /bin/lfsbackup.jar, this can be done either by double clicking on it, or executing "**java –jar lfsbackup.jar"** command from the command prompt. If you want to do silent backup execute **"java –jar lfsbackup.jar your-file-name"** command.



This utility performs complete backup of data managed by logFaces server. Depending on the size of your database, this operation may take several minutes.

IMPORTANT: Please make sure that logFaces server is currently NOT running.

Figure 2-7Database backup utility

Backup file can later be used for importing it into another database or simply revert the situation to the backup point. Use the client to import backup data, see "Import and export" section for more details.

2.5.4 How do I increase server JVM memory?

Open **\bin\ lfs.conf** file – this is a bootstrap configuration file.

JVM memory is setup with those attributes:

```
wrapper.java.initmemory=256
```

```
wrapper.java.maxmemory=512
```

Those values are default, if you experience extensive memory usage, try to increase maxmemory property value.

3 Getting started with logFaces Client

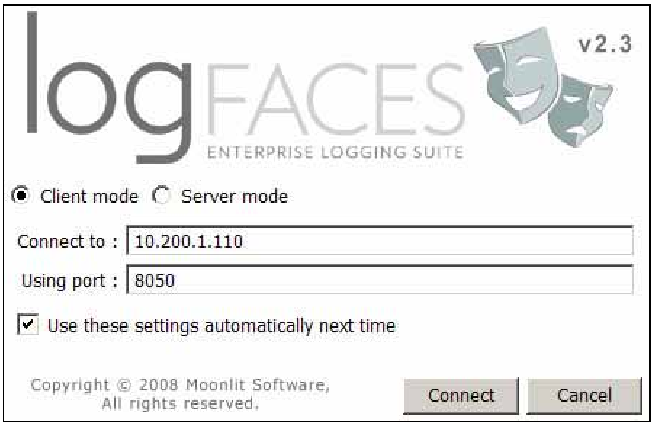
LogFaces Client is a rich GUI application which can be installed anywhere on the network to visually work with log data coming either from logFaces Server or directly from your applications.

3.1 Installing logFaces Client

LogFaces Client installation is almost identical for both Windows and Linux platforms, just make sure you have Java™ Runtime Environment (JRE) 1.5 or later installed. On Windows – simply run the executable installer. On Linux execute command **java –jar lfs-client-linux.jar** to start the installer process. On Windows, the installer will create shortcuts in Start menu for running the application and for the uninstaller. If you choose, the shortcut will also be created on the desktop. On Linux run the **logfaces** executable file from installation directory.

3.2 Modes of operation

LogFaces Client can work in two modes - **Client Mode** or **Server Mode**. You select the mode during application startup. In Client Mode the application connects to and works with remote logFaces Server instance.



● Client mode ○ Server mode

Connect to : 10.200.1.110

Using port : 8050

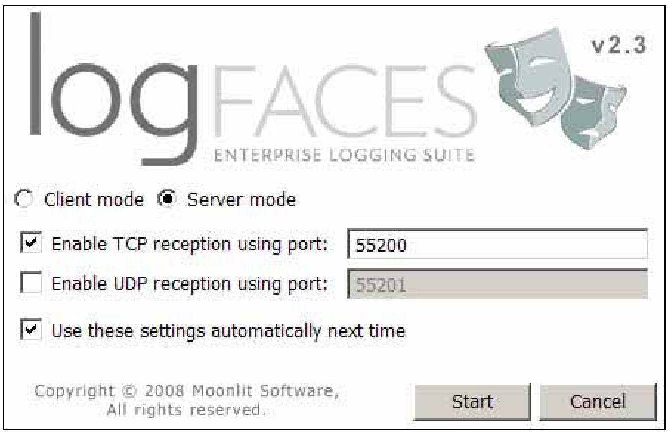
☑ Use these settings automatically next time

Copyright © 2008 Moonlit Software,
All rights reserved.

Figure 3-1 Client Mode

In **Client Mode** we specify logFaces Server **host** name and **port** number. Application will remember these options for the next time you run it, but you can also indicate to use those settings automatically in the next time and not asking again. It is possible to modify these settings in the File/Preferences menu later. The communication between logFaces Client and Server is one way (from client to server) and is HTTP based, so normally there shouldn't be a problem with firewalls. Of course, the access through the given port should be allowed by your network administrator.

In **Server Mode** the application runs with embedded **compact** version of logFaces Server. This is a limited (and less expensive) version of server and client combined into single application which provides only **real time viewing** of log data. You can use it when you don't need database and other features available in standalone logFaces Server.



○ Client mode ● Server mode

☑ Enable TCP reception using port: 55200

☐ Enable UDP reception using port: 55201

☑ Use these settings automatically next time

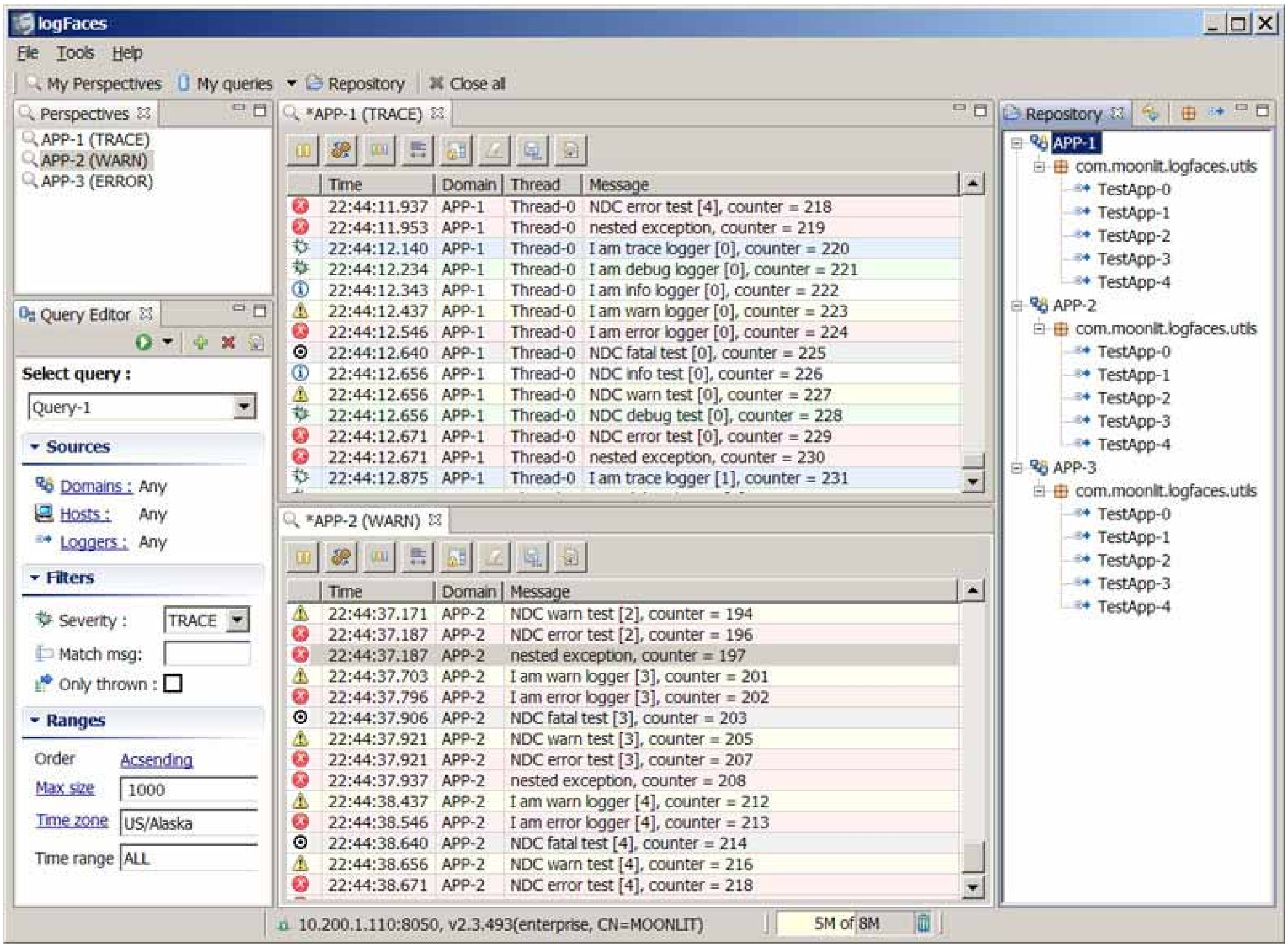
Copyright © 2008 Moonlit Software,
All rights reserved.

In order to run in **Server Mode** we need to specify at least one of the ports which will be used by the application to receive log data from appenders in your applications. You can specify either TCP, or UDP or both, just make sure those ports are available and your application appenders are configured to log into this host and those ports.

Both modes look very much alike from user experience point of view, except that in Server Mode there is no database and querying features.

Note that in order to run in ***Server Mode*** *you need to install the license on the computer where you run it. As with the logFaces Server, the first time launch will automatically activate 30 days free trial. Note also that this is not the same license type as installed on the logFaces Server. This license needs to be purchased and installed separately unless you hold OEM license.*

3.3 Layout



Select query : Query-1

Sources

Domains : Any
Hosts : Any
Loggers : Any

Filters

Severity : TRACE
Match msg:
Only thrown : ☐

Ranges

Order Acsending
Max size 1000
Time zone US/Alaska
Time range ALL

Time	Domain	Thread	Message
22:44:11.937	APP-1	Thread-0	NDC error test [4], counter = 218
22:44:11.953	APP-1	Thread-0	nested exception, counter = 219
22:44:12.140	APP-1	Thread-0	I am trace logger [0], counter = 220
22:44:12.234	APP-1	Thread-0	I am debug logger [0], counter = 221
22:44:12.343	APP-1	Thread-0	I am info logger [0], counter = 222
22:44:12.437	APP-1	Thread-0	I am warn logger [0], counter = 223
22:44:12.546	APP-1	Thread-0	I am error logger [0], counter = 224
22:44:12.640	APP-1	Thread-0	NDC fatal test [0], counter = 225
22:44:12.656	APP-1	Thread-0	NDC info test [0], counter = 226
22:44:12.656	APP-1	Thread-0	NDC warn test [0], counter = 227
22:44:12.656	APP-1	Thread-0	NDC debug test [0], counter = 228
22:44:12.671	APP-1	Thread-0	NDC error test [0], counter = 229
22:44:12.671	APP-1	Thread-0	nested exception, counter = 230
22:44:12.875	APP-1	Thread-0	I am trace logger [1], counter = 231

Time	Domain	Message
22:44:37.171	APP-2	NDC warn test [2], counter = 194
22:44:37.187	APP-2	NDC error test [2], counter = 196
22:44:37.187	APP-2	nested exception, counter = 197
22:44:37.703	APP-2	I am warn logger [3], counter = 201
22:44:37.796	APP-2	I am error logger [3], counter = 202
22:44:37.906	APP-2	NDC fatal test [3], counter = 203
22:44:37.921	APP-2	NDC warn test [3], counter = 205
22:44:37.921	APP-2	NDC error test [3], counter = 207
22:44:37.937	APP-2	nested exception, counter = 208
22:44:38.437	APP-2	I am warn logger [4], counter = 212
22:44:38.546	APP-2	I am error logger [4], counter = 213
22:44:38.640	APP-2	NDC fatal test [4], counter = 214
22:44:38.656	APP-2	NDC warn test [4], counter = 216
22:44:38.671	APP-2	NDC error test [4], counter = 218

10.200.1.110:8050, v2.3.493(enterprise, CN=MOONLIT)   5M of 8M

Figure 3-2 Application layout

LogFaces Client is an Eclipse based application; not only it has a look and feel of the Eclipse but is based on the Eclipse. So, if you're familiar with Eclipse, then using the application will be a breeze.

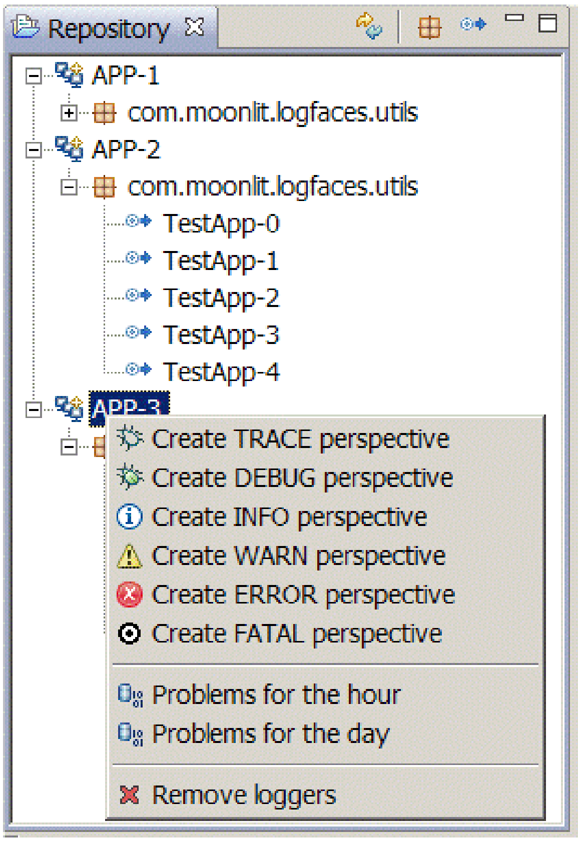
Top left part by default contains Views - for accessing log perspective you create. You can use mouse right click for activating, adding, removing or modifying those perspectives.

Bottom left part by default contains Query Editor for launching and managing database queries.

Rightmost part by default is occupied by Repository which is a list showing which loggers and what applications are known to the logFaces Server at the moment.

The middle part of the screen will be containing active log perspectives and/or database queries you execute. All parts of the screen can be organized to suit your taste; the top toolbar under menu gives you a quick access to the views.

3.4 Repository



Repository is a view containing loggers which are known so far to the logFaces Server. It has two purposes. First of all, it's informative; it shows you what your system looks like log-wise. Secondly, Repository is a quick start for creating perspectives, getting fast access to the errors and deleting unwanted loggers. Right click on it's content to get available operations.

Repository structure is tree-like. The top level is called Domain (or Application). This information is extracted by the server from incoming events by looking into specific property, called "Application". This property is specified when you integrate your application with logFaces. If this property is not specified, the Server will automatically create "Default Domain" and associate all unknown events under this domain. It is highly recommended to setup your application that it provides unique Domain name – it will further help you with quires and give more organized view on your system.

Under Domain resides a Package – leftmost part of the logger name. Some applications might not have any package, it is not a requirement.

Under Package we have loggers - actual sources of log data.

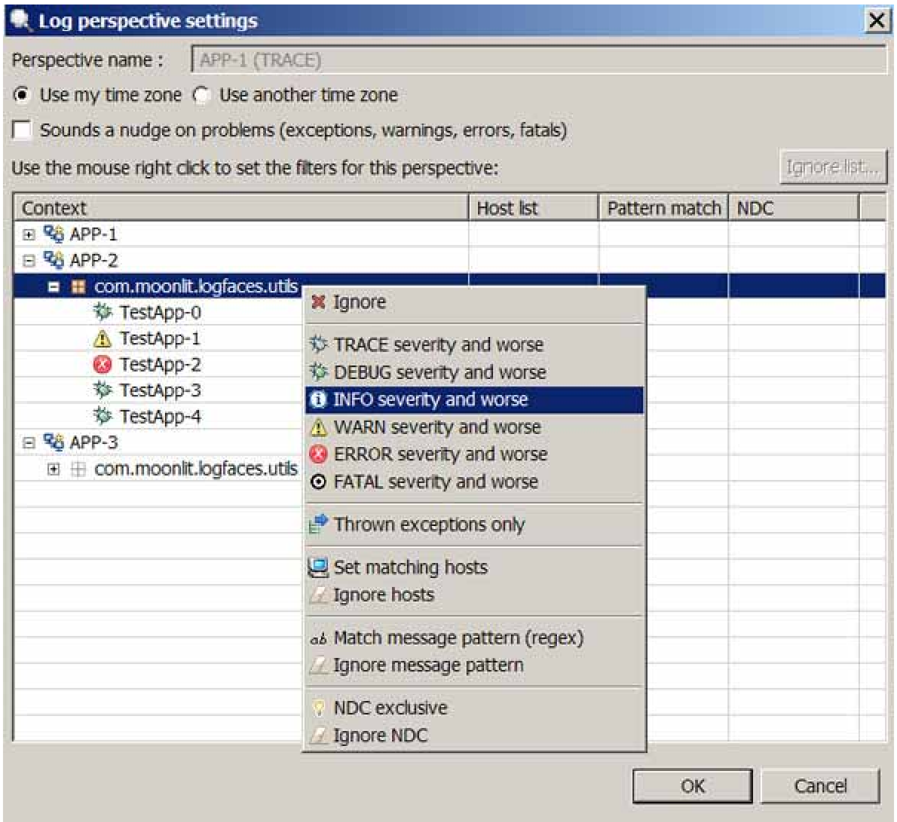
IMPORTANT: When you run your system for the first time, the Repository might not contain all the loggers you actually have in your system – the Server will know your loggers only when they log at least one statement. It is recommended to let your system run for a while before you actually start using the Repository. Also, as your system evolves and there are new loggers (classes?) introduced, you might want to do the refresh to make them appear in Repository. Otherwise, new loggers will not be visible by the Client (unless it gets restarted).

3.5 Creating log perspectives

Log perspective is a real time view on the log stream going through the logFaces Server. Perspectives are organized by name so that you can refer to them later. Each perspective is simply a set of filtering rules telling the Server what logging events should be routed to it. Server has no limitation on amount of perspectives you create. Only when you actually activate the perspective the Server deals with it, other than that Server doesn't know about perspectives. Number of active perspectives is only limited by the physical memory the Server is using. There are two ways the perspectives can be created – "quick" and "advanced"; both ways are actually trivial, but we differentiate them here for the sake of clarity.

The "quick" way is from Repository - select loggers, right click on selection and choose the severity option you need. Then simply give the name to your perspective and you're done. This will create a perspective whose loggers are all set into the severity you specified. Later on these settings can be fine tuned.

The "advanced" way allows to fine tune the perspective not only by severity, but also using various filters. You create the perspective from the Perspectives view right click menu.



Log perspective settings

Perspective name : APP-1 (TRACE)

● Use my time zone ○ Use another time zone

☐ Sounds a nudge on problems (exceptions, warnings, errors, fatals)

Use the mouse right click to set the filters for this perspective:

Figure 3-3Perspective settings dialog

In perspective settings dialog you can select loggers individually and specify the filtering rule for each one of them separately. When you right click on the tree, which is actually the snapshot of Repository, the drop down menu will provide several options. Note that those options will be applied recursively to the tree, so that if you click on the package or domain node, the changes will propagate to all its child nodes. And the options are:

- **Severities** – from TRACE (the lowest) to the FATAL (the highest). This is basically a threshold which discards events with severity lower than you specify.
- **Thrown exceptions only** – will include only those events which are thrown exceptions. Note that thrown events can be with any severity.
- **Matching hosts** – will include only log statements from the host list you specify. Very useful when application is spread on several machines and have loggers with the same names.
- **Match message pattern** – will do regular expression matching to the log messages. Only those which match the pattern will be passed to the user.
- **NDC exclusive** – will include only those messages which have NDC context attached. Useful for debugging.

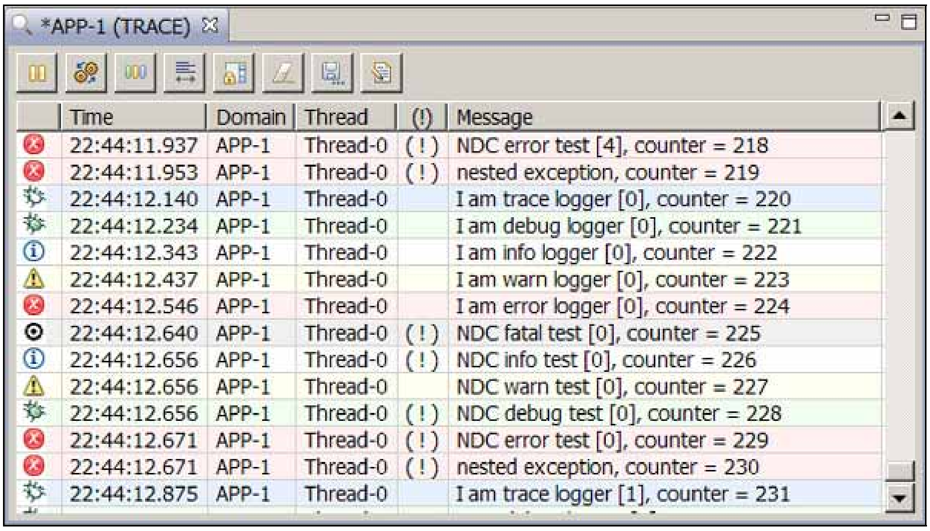
The check box "**Sound a nudge on problems**" is very useful if you want to have sound notifications. Your perspective will make a gentle beep when messages with WARNING+ severities will be going through. The frequency of the sound notification is limited to 5 seconds; in any case, if your system is not stable and there are lots of errors, this feature would be annoying. Its best usage is when every error in the system requires your attention and it doesn't happen too often.

You can specify **Time Zone** settings. Server keeps log events in UTC time format; if you're accessing the server from another time zone and want to see everything related to the original source time – select the appropriate time zone from the combo box. If "**Use my time zone**" option is selected, all time stamps will be shown relatively to the time zone of where the Client is running.

Ignore List is a collection of source locations which will be filtered out even if the logger passes through the filters applied to it. Ignore list is convenient when you want to shut off some monotonous messages from particular location in the code. Each entry in the list is described by file name, class, method and line number from which the log was made.

3.6 Real-time perspective view

Perspective view is actually the log which goes through this perspective.



	Time	Domain	Thread	(!)	Message
⊗	22:44:11.937	APP-1	Thread-0	(!)	NDC error test [4], counter = 218
⊗	22:44:11.953	APP-1	Thread-0	(!)	nested exception, counter = 219
✲	22:44:12.140	APP-1	Thread-0		I am trace logger [0], counter = 220
✲	22:44:12.234	APP-1	Thread-0		I am debug logger [0], counter = 221
ⓘ	22:44:12.343	APP-1	Thread-0		I am info logger [0], counter = 222
⚠	22:44:12.437	APP-1	Thread-0		I am warn logger [0], counter = 223
⊗	22:44:12.546	APP-1	Thread-0		I am error logger [0], counter = 224
⊙	22:44:12.640	APP-1	Thread-0	(!)	NDC fatal test [0], counter = 225
ⓘ	22:44:12.656	APP-1	Thread-0	(!)	NDC info test [0], counter = 226
⚠	22:44:12.656	APP-1	Thread-0		NDC warn test [0], counter = 227
✲	22:44:12.656	APP-1	Thread-0	(!)	NDC debug test [0], counter = 228
⊗	22:44:12.671	APP-1	Thread-0	(!)	NDC error test [0], counter = 229
⊗	22:44:12.671	APP-1	Thread-0	(!)	nested exception, counter = 230
✲	22:44:12.875	APP-1	Thread-0		I am trace logger [1], counter = 231

Figure 3-4 Perspective view in real time

Logging events are organized in a table and displayed in real time as events arrive from logFaces Server. Lines are colored by the severity which is also emphasized by the icon on the left. Severity icons and colors denoted as follows:

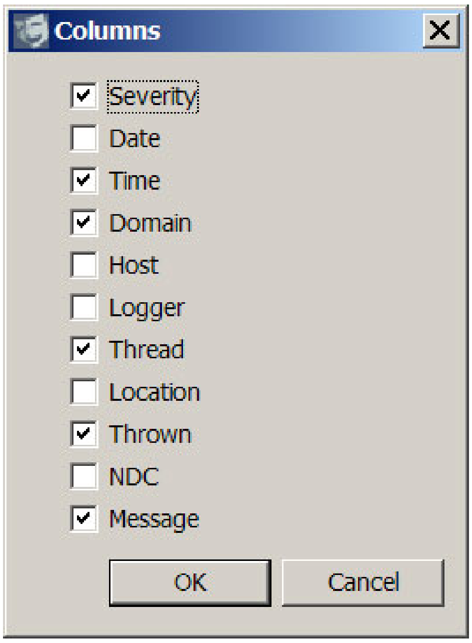
- ✲ TRACE; blue background
- ✲ DEBUG; green background
- ⓘ INFO; white background
- ⚠ WARNING; yellow background
- ⊗ ERROR; red background
- ⊙ FATAL; gray background

Each perspective has its own set of controls on the top of the panel:

- ⏸ - Pause/resume; when perspective paused, the flow of events will be temporally stopped until perspective is resumed.

- ⚙ - Access to the settings where you can modify perspective filters. This can be done any time and applies immediately in real time.

- – Allows you to specify what columns of logging events you want to be visible in this view. Those are the options :



☑ Severity
☐ Date
☑ Time
☑ Domain
☐ Host
☐ Logger
☑ Thread
☐ Location
☑ Thrown
☐ NDC
☑ Message

Figure 3-5 Columns selection

- – Automatically arranges the column width to fit the content.

- – Toggles the scrolling of the view. When view is unlocked the amount of log events is limited to 20000 in order to prevent memory overflow in the client. However, when locked, the perspective will not be rolled and will accumulate all the events; beware that this can exhaust the memory of the application.

- – Erases everything from the view

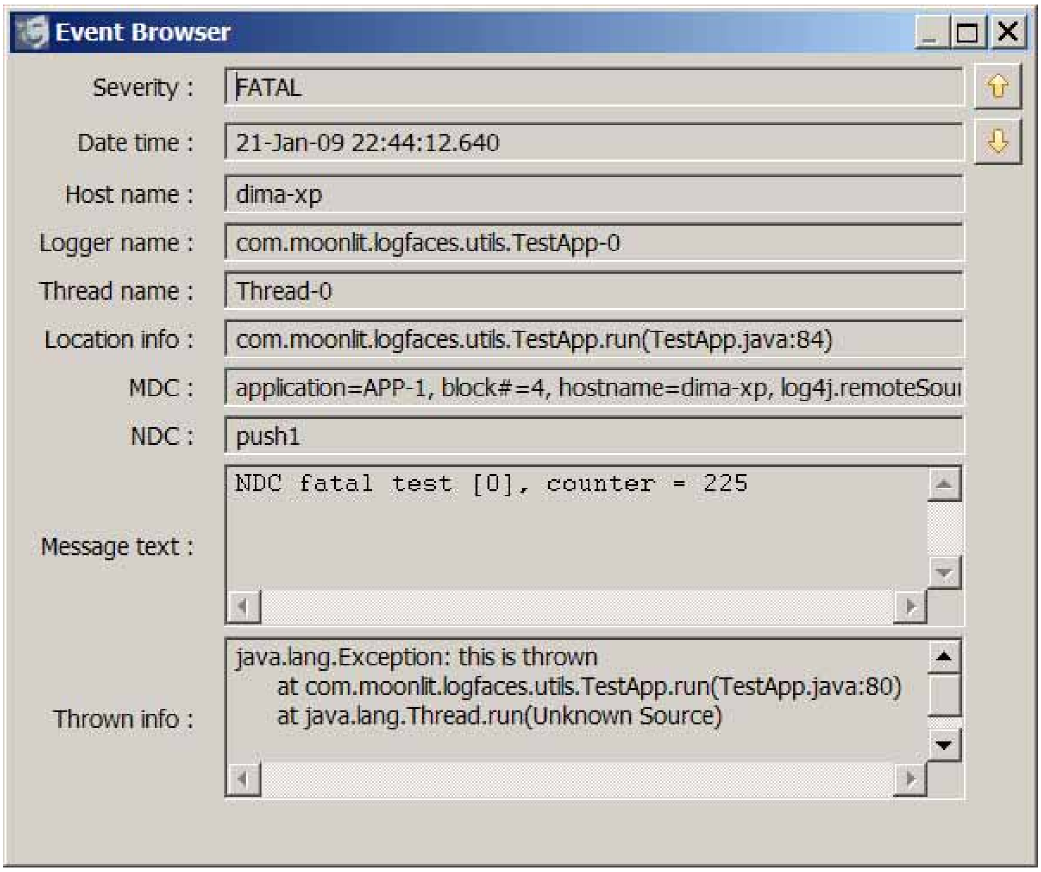
- – Allows saving the content of the view into a text file – everything you have in the view will be saved into the text file.

- – Automatically opens view content in the text editor. In order to work with files you first should specify which text editor you would like to use as shown below; this is done only once.

Note that file operations like saving to a file or opening log in external editor use the format which you can modify in Preferences/Files section. The layout format defined there will be applied to all file related operations. The format itself is specified in Apache Log4j documentation, you might want to read about it here.

3.7 Event browser

To see the details of logging event simply double click on it in the view. Event browser looks like this:



Severity : FATAL

Date time : 21-Jan-09 22:44:12.640

Host name : dima-xp

Logger name : com.moonlit.logfaces.utils.TestApp-0

Thread name : Thread-0

Location info : com.moonlit.logfaces.utils.TestApp.run(TestApp.java:84)

MDC : application=APP-1, block#=4, hostname=dima-xp, log4j.remoteSou

NDC : push1

Message text :

```
NDC fatal test [0], counter = 225
```

Thrown info :

```
java.lang.Exception: this is thrown
      at com.moonlit.logfaces.utils.TestApp.run(TestApp.java:80)
      at java.lang.Thread.run(Unknown Source)
```

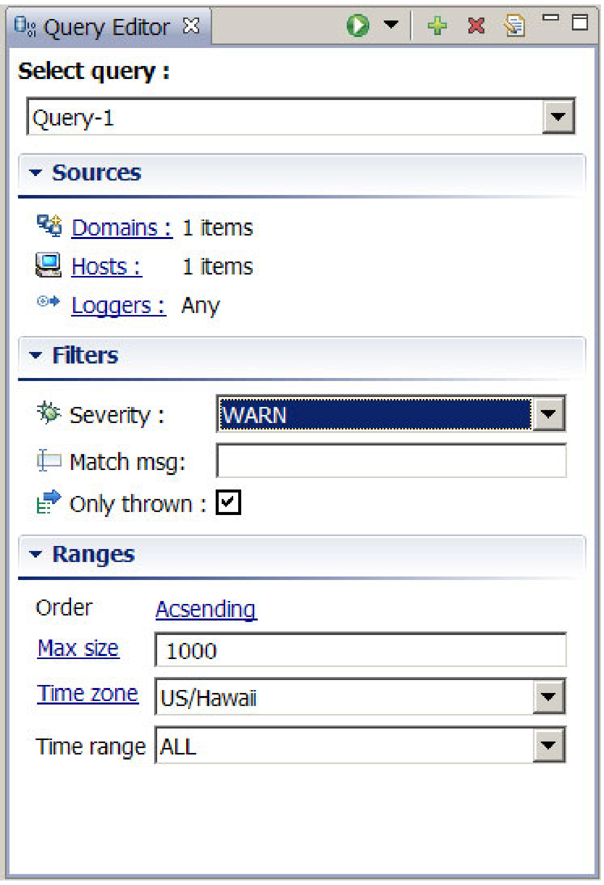
Figure 3-6 Event browser

Here you can see complete information about the event. Use ⇧ and ⇩ buttons to navigate up and down in the view

3.8 Using query editor

Query Editor is for creating database queries. The queries you create then can be accessed quickly from the top toolbar, or simply by launching them in place by clicking ▶.

You create a new query by clicking ✚ button. Queries are named and displayed in the list combo. In the query we specify criteria by which we want the logging events to be retrieved from the database. The options are:



Select query :
Query-1
Sources
Domains : 1 items
Hosts : 1 items
Loggers : Any
Filters
Severity : WARN
Match msg:
Only thrown : ☑
Ranges
Order Acsending
Max size 1000
Time zone US/Hawaii
Time range ALL

- **Domains** – defines the origin domains of logging event. The query will include only those events which come from the domains you specify or from any domain if nothing is selected.
- **Hosts** – defines the origin hosts of logging event. The query will include only those events which come from the hosts you specify or from any host if nothing is selected.
- **Loggers** – defines the origin logger names. The query will include only those events which come from the loggers you specify or from any logger if nothing is selected.
- **Severity** – the threshold which will filter the events by their severity where DEBUG is the lowest and FATAL is the highest.
- **Match msg** – text matching string; will be applied to include only those events whose message text match the specified expression. Works exactly as SQL "like" statement.
- **Only thrown** – when selected will include only those events which originate from exceptions
- **Order** – the order of the results to display. Events are organized in database by time; you have a option to display the result in ascending or descending order.

- **Size** – defines the size of results to display. You never know how much data will get back from the query, so in order to be on the safe side and not to exhaust the memory of the client application, we have a limit.

- **Time zone** – if you are accessing the Server located in another time zone and want to receive queries related to that time zone, then you should specify your Server time zone here. By default the query is set to use client's time zone.

- **Time range** – specifies the time range of the logging events. There are several predefined setups, like "1 Day from now" and the like. But you can also specify **Custom Range** stating the exact start and end time.

If you want to save results of the query directly into the file, select one of the options from the combo. You can save results to :

- Text log file; the pattern of the file will be as specified in File/Prefferences
- XML file; the schema is according to log4j DTD
- Binary file; can be used for later import operations on other logFaces servers.

Once you define the query, it stays in the list and is also available in the top toolbar like this:

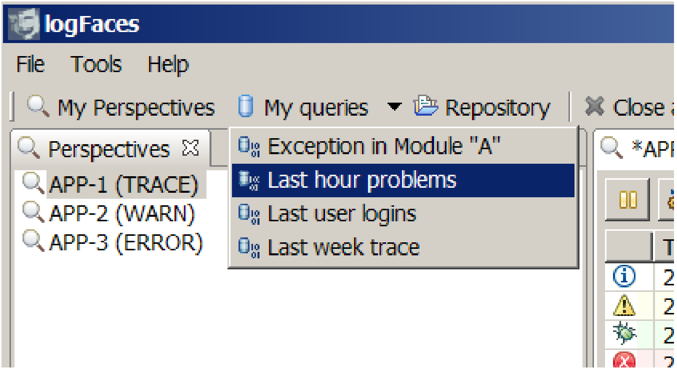
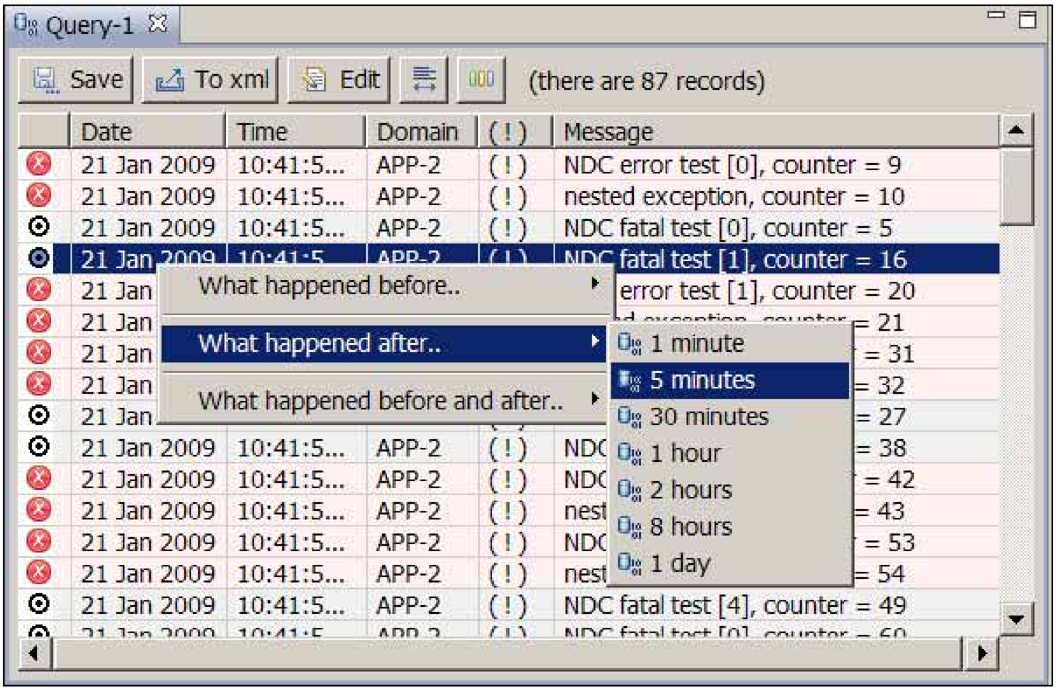


Figure 3-7 Query quick run

All you have to do now is to select it, the results will appear in the view similar to the one of the real time perspectives we have already seen. As with real time perspectives, the query can be saved to file or opened with external text editor. This way you can create log files dynamically at will instead of having your application to manage them on local disk.

3.9 Drill down

One of the most efficient ways to dig into problems is to first grab them all and then see what happened before or after that point of time. For example, you can define a query which grabs only thrown exceptions occurred yesterday and then use right click on the query results; the drop down menu will let you drill into that particular moment. You can go backwards, forward and both ways while specifying how wide should be the gap:



(there are 87 records)

	Date	Time	Domain	(!)	Message
⊗	21 Jan 2009	10:41:5...	APP-2	(!)	NDC error test [0], counter = 9
⊗	21 Jan 2009	10:41:5...	APP-2	(!)	nested exception, counter = 10
⊙	21 Jan 2009	10:41:5...	APP-2	(!)	NDC fatal test [0], counter = 5
⊙	21 Jan 2009	10:41:5	APP-2	(!)	NDC fatal test [1], counter = 16
⊗	21 Jan				error test [1], counter = 20
⊗	21 Jan				= 21
⊗	21 Jan				= 31
⊗	21 Jan				= 32
⊙	21 Jan				= 27
⊙	21 Jan 2009	10:41:5...	APP-2	(!)	NDC = 38
⊗	21 Jan 2009	10:41:5...	APP-2	(!)	NDC = 42
⊗	21 Jan 2009	10:41:5...	APP-2	(!)	nest = 43
⊗	21 Jan 2009	10:41:5...	APP-2	(!)	NDC = 53
⊗	21 Jan 2009	10:41:5...	APP-2	(!)	nest = 54
⊙	21 Jan 2009	10:41:5...	APP-2	(!)	NDC fatal test [4], counter = 49

Figure 3-8 Drill down

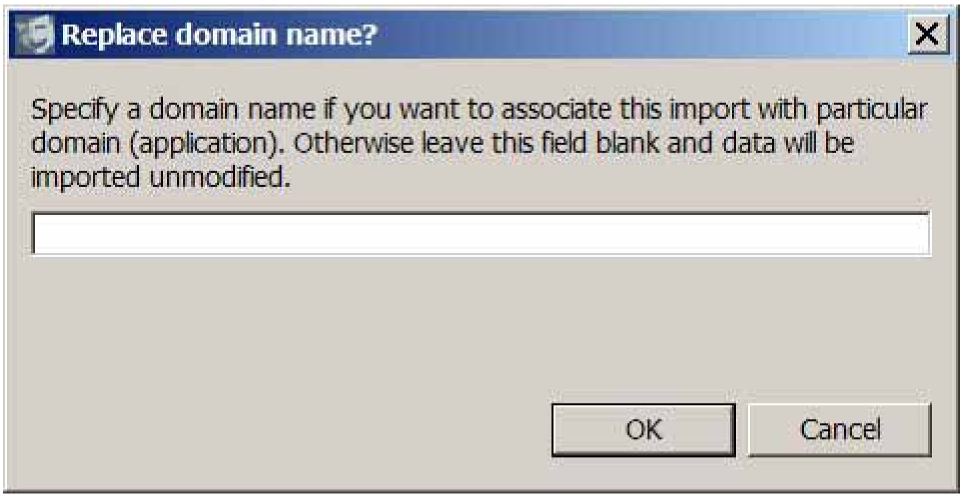
The drill down query will open up in another view and will contain logging events related to originator's Domain covering all severities and time span you defined – this way you will see what other parts of the system were "saying" when this problem took place.

3.10 Import and Export

Data can be exported from one server and imported on the other, we use binary data image for those operations. As mentioned earlier, you can export query results directly into binary file and import it later on somewhere else.

To export all data from the database, go to Tools menu and select "Export data from database". You will be prompted for the file name. This operation is identical to what is done by Backup utility.

To import data into database, go to Tools menu and select "Import data into database". You will be prompted for the **.lfb** file name – binary data exported previously from logFaces. Before the import actually starts, you will be asked to specify a Domain name to which imported data should be associated:


Replace domain name?

Specify a domain name if you want to associate this import with particular domain (application). Otherwise leave this field blank and data will be imported unmodified.

If you leave this field blank, the data will be imported without any modifications. Otherwise, the Domain name of all events will be replaced to the value you provide. This is very convenient when you exchange data between different logFaces servers.

Please take into account, that depending on the dataset size and the speed of the database, those operations may take up to several minutes while taking considerable amount of CPU and database I/O.

3.11 Workspace

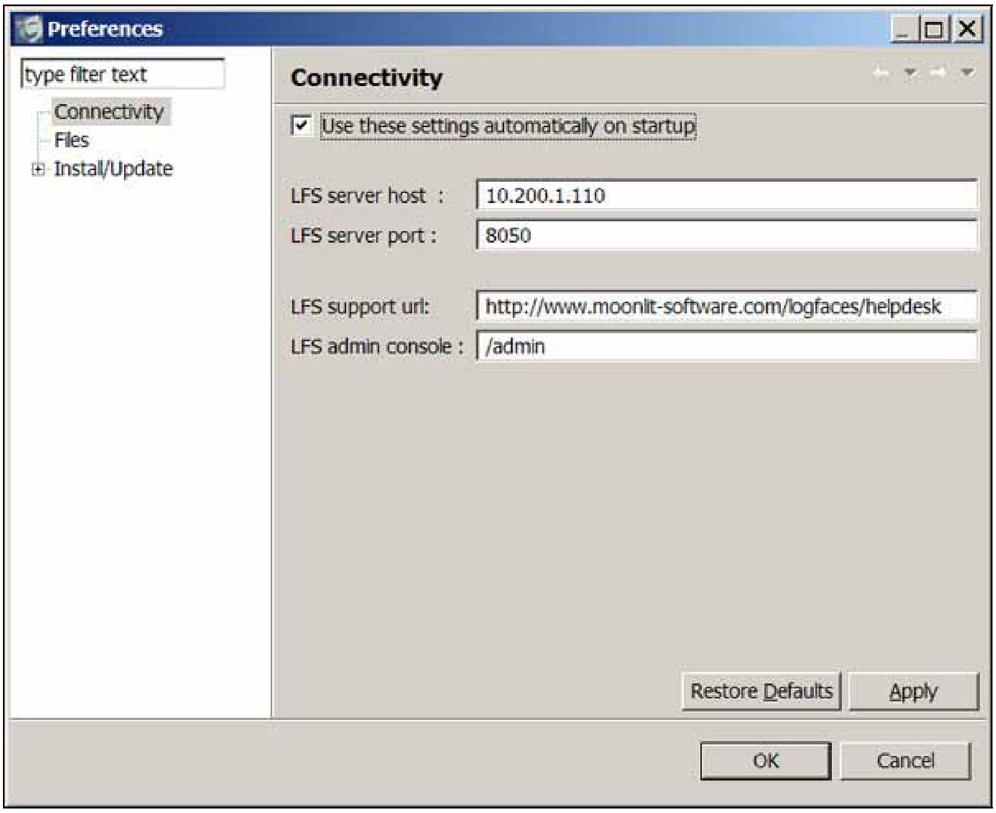
All settings you do in the application including visual layout, perspectives and queries are saved locally on your disk. Every time the application is run those settings are restored to the latest.

You can **export** the workspace configuration into a text file and **import** it on another computer. Import/export operations are available through the **File** menu.

3.12 Preferences

There are several global preferences which are also accessible through the File menu – **Connectivity**, **Files** and **Install/Update**

In **Connectivity** section you can specify the server connection endpoints, support site URL and path to the administration console. Connection endpoints (host and port) will be affective next time you start the application. Support site URL is used for submitting bugs, feature requests or questions – you will be taken to this URL automatically when selecting **Help/Bug report** menu. Admin console parameter is the path of administration console which also can be access from the help menu – it will simply open a browser view to allow you to administer the server without leaving the workbench.



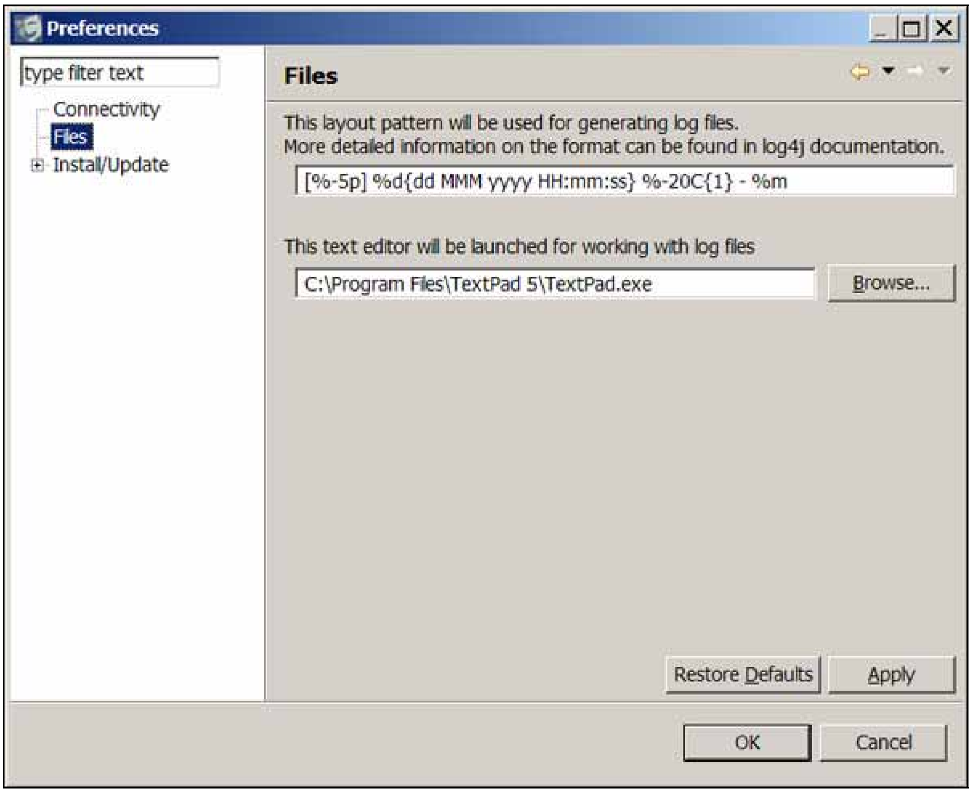
☑ Use these settings automatically on startup

LFS server host : 10.200.1.110
LFS server port : 8050
LFS support url: http://www.moonlit-software.com/logfaces/helpdesk
LFS admin console : /admin

Figure 3-9Connectivity Preferences

In **Files** section you specify the layout format of log files and external editor to be used with the application. The layout format will be applied to all file related operations you do when saving queries to a file or saving the perspective views to a file. The format itself is specified in Apache Log4j documentation, you might want to read about it here.

External text editor is launched every time when you want to display the content in the text editor for doing some text operations. By default after installation the editor is not specified, you will have to manually do it by clicking on a "Browse" button here in preferences:



This layout pattern will be used for generating log files.
More detailed information on the format can be found in log4j documentation.

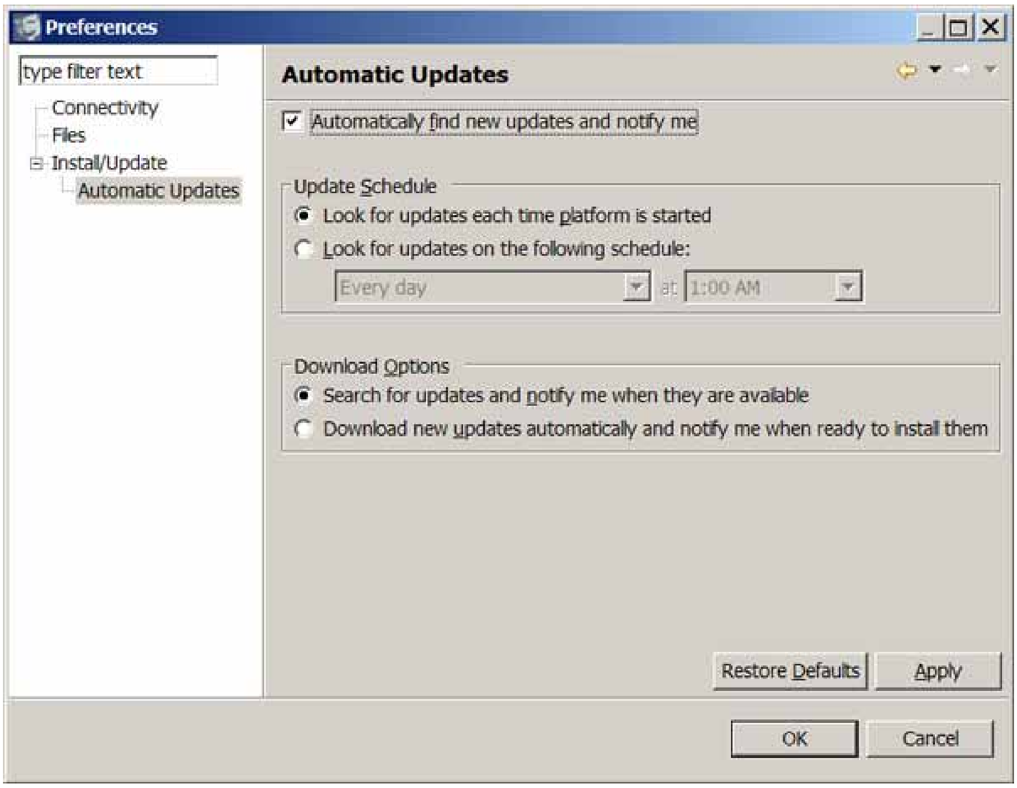
[%-5p] %d{dd MMM yyyy HH:mm:ss} %-20C{1} - %m

This text editor will be launched for working with log files

C:\Program Files\TextPad 5\TextPad.exe

Figure 3-10 Files Preferences

In **Install/Update** section you specify how you would like to obtain software updates. There are several options which specify the update policy, for example, you can request to check for the updates every time the application is run and notify when they're available for installation:



Automatic Updates

☑ Automatically find new updates and notify me

Update Schedule

● Look for updates each time platform is started

○ Look for updates on the following schedule:

Every day at 1:00 AM

Download Options

● Search for updates and notify me when they are available

○ Download new updates automatically and notify me when ready to install them

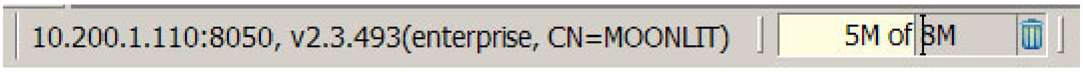
Figure 3-11 Update Preferences

You may as well disable the automatic updates and do it manually from the Help menu on the main menu bar. When new version of the software will be released, the application will display notification dialog and ask your permission to install the updates. Normally this will require consequent restart of the application, but in many cases it won't be necessary.

3.13 Status bar

Status bar displays current logFaces Server host:port, software version and license information.

The gadget on the right displays heap memory usage.



10.200.1.110:8050, v2.3.493(enterprise, CN=MOONLIT)

5M of 8M

Figure 3-12 Status Bar