



Journyx Technical Document

System Administration and Disaster Recovery Guide

For Journyx Timesheet version 9.0

Document Version: 3.5 (October 2013)
For product versions prior to 9.0, please see an earlier version of this guide.

The latest version of this document is always available at:

<http://community.journyx.com/product-documentation/>

Introduction	3
Direct Database Modification	3
Support	3
General Architecture of the Application	4
Linux Directory Layout	4
Windows Directory Layout	5
Logging Locations	5
Administrative Topics	6
Installation Notes	6
System Hardware and Software Requirements	6
External Database Setup	8
Linux Installation Notes	9
External FastCGI web server (Apache) on Linux	10
Windows Installer Digital Signature	11
Windows Installation Notes	12
Configuring the Web Site on Windows	12
Multiple Application Servers (Clustering)	15
Clustering Overview	15
Understanding the Journyx Installation	15
Installing on Multiple Application Servers	16
Multiple Application Servers on Windows	17
Load Balancing	17
Configuring Journyx products to properly report URLs	17
Easier installation on Unix variants	18
Security Issues	18
General Security Issues	18
Internal Database (PostgreSQL) account security	19
External Reporting	19
HTTP Trace and Track verbs	21
Secure Sockets Layer (SSL) configuration	22
Configuration and Tuning	24
General Options	24
Performance and Memory Options	24
Configuring External Memcached	25
Database Tuning	26
Custom Error Handlers	27
Starting, Stopping and Restarting the Service	28
Linux Service Control	28
Windows Service Control	28
Other Command Line Tasks	28
Disaster Recovery	29
Purpose	29

Backup and Restore Summary.....	29
Disaster Recovery Planning.....	29
Backup Techniques.....	30
Recovery Strategy.....	31
Backup Procedures.....	32
Running backupdb.....	32
backupdb options.....	33
Database Archive (Date Range based backup).....	34
Basic Archive Usage.....	34
Archive Limitations and Considerations.....	35
Understanding Archive Date Ranges.....	35
Archive Preparation Checklist.....	36
Archive Options.....	37
Restoring Timesheet Databases.....	39
restoredb options.....	39
Other Backup / Restore Topics.....	41
Automating Backups.....	41
Native Backup Formats.....	42
Oracle.....	42
Microsoft SQL Server 2005 and later.....	42
PostgreSQL Native Backups.....	42
Change Log.....	44

Introduction

In general Journyx Timesheet does not require significant systems administration attention. It is a turn-key system and can be downloaded and installed complete with database and web server. Once the product is up and running, especially with the built-in database option, administrative overhead is low. However we recommend you put a full disaster recovery strategy in place once the system is up and running but *before* an emergency happens such as a hardware or disk failure, or a natural disaster at your site such as a flood, fire, or tornado.

This document is intended to give System Administrators an overview of the application infrastructure and a few tips on how to diagnose the occasional problem, tweak certain system parameters, as well as how to design and implement a disaster recovery strategy.

This version of the document is for both Windows and Linux based systems. Linux is the only Unix-like operating system supported. Journyx supports different flavors of Linux for Intel processors in both 32 bit and 64 bit editions. Both 32-bit and 64-bit versions are available for Microsoft Windows. When running a 64-bit operating system, you must use the 64-bit version of Timesheet, and likewise when running a 32-bit operating system you must use the 32-bit version of Timesheet.

The latest version of this document is always available at:

<http://community.journyx.com/product-documentation>

Direct Database Modification

There are two options for manipulating the database or performing administration outside of the interface provided through the Timesheet Administration Web Interface and command line tools.

One is to use the jxAPI XML/SOAP/WSDL programming interface. See the link on the product Sitemap page for the latest jxAPI documentation.

The other is to contact Journyx for Professional Services for help in meeting your needs. For information on Journyx Professional Services, contact your Journyx Sales Representative. <http://journyx.com/contact-us>

Under no circumstances should you attempt direct database manipulation on the Journyx Timesheet database. Doing so can void your Maintenance Contract for any problems caused by the alterations. Read-only access to the database is allowed for the purposes of external reporting software. Please read the Security chapter of this document for more details about external SQL reporting.

Support

The Journyx website has developer resources and commonly requested documents at the Support page at <http://community.journyx.com/support> .

Support is also available via the public knowledgebase and product documentation:

<http://community.journyx.com/support/knowledgebase>

<http://community.journyx.com/product-documentation>

Additional support is available via the help desk for paid customers:

<http://community.journyx.com/support> (Select Helpdesk from the menu under the Support tab.)

If you have further questions, you can contact support@journyx.com via email or through the online helpdesk link above.

General Architecture of the Application

The application server uses the typical three-tier architecture – a front-facing web server, a back-end SQL database, and an application server in between which runs business logic. The client side is a common web browser such as Microsoft Internet Explorer (version 8.0 or higher recommended), Safari or Firefox. The browser communicates to the application via a standard HTTP Web Server. In the case of Linux installs, the default Web Server is Lighttpd, but you can also run under any server which supports external TCP based FastCGI servers, such as Apache. The standard FastCGI protocol is used to communicate between the web server and the separate Python application server processes (also called “daemons”.) These daemons manage the database connections and use the HTTP form data, application logic and database records to process the request. The result is channeled back through the web server and then to the client’s web browser.

Under Windows you can run under either the Microsoft IIS web server or else the included Lighttpd server. When running Lighttpd it uses FastCGI to talk to the “daemons.” When running under IIS, the daemons are managed directly by IIS in an Application Pool environment. An ISAPI plug-in is used to run Python inside the application pool. IIS is the recommended web server for Windows.

In addition to the 3 major components listed above – a web server, a database server, and the application service, Timesheet also uses a central data caching system called “memcached” (memory cache daemon) to speed up certain aspects of the application. Timesheet supplies an “internal” copy of memcached that is used automatically and reserves (by default) up to 128 MB of memory for this purpose. However you can also use an external memcached process under your control. This process can reside on the application server host, or on a different host where it can use a greater share of memory. Configuring an external memcached is described below as well as how to change the amount of memory the internal memcached uses.

The Timesheet application logic is implemented in Python modules found in three main directories. Another directory is used as a stepping point for using the command-line interface (CLI) to the application to perform manual starting and stopping of the application and other administrative tasks.

The tables below shows a few directories that the System Administrator should be aware of and gives a short explanation of their purpose.

Linux Directory Layout

All paths are from the install directory where the jinstall script was directed to place the installation, which is saved as the \$WTHOME environment variable in the setup file.

Location	Purpose
\$WTHOME/pi/bin	Location of the setup file and CLI tools
\$WTHOME/pi/pylib/timesheet	Application Framework modules
\$WTHOME/pi/pylib/cgi	Application “Screen” modules
\$WTHOME/pi/pylib/wtlib	Application library modules
\$WTHOME/tmp	Timesheet logging directory
\$WTHOME/pi/www	Lighttpd root directory
\$WTHOME/pi/htdocs	Web server static files
\$WTHOME/pi/www/logs	Lighttpd logs
\$WTHOME/pi/db/data	PostgreSQL data directory
\$WTHOME/pi/db/serverlog	PostgreSQL log file
\$WTHOME/pd/Linux/python/bin/python	Python language executable

pi/bin is the directory where most system administrative tasks are performed. The file in that directory called “setup” contains system environment variable values that should be in the install users environment whenever performing operations with the CLI such as backupdb and restoredb. CLI operations must only be performed under the user id that installs Timesheet. You can source the setup file in the install user’s ~/.profile or

~/`.bashrc` or equivalent, or simply source the file prior to performing any system maintenance as the installing user.

```
cd <install directory>/pi/bin
. ./setup
```

That's a dot, then a space, then dot slash setup. Or alternatively,

```
source <install directory>/pi/bin/setup
```

Usually it's easier to add the sourcing of the setup file (the above command) to a shell profile file such as ~/`.bashrc` so you don't have to remember to do it every time you log into this account. You should do this under the Linux account where you installed Timesheet. On Windows everything is keyed off the `%WTHOME%` environment variable instead.

`cgi`, `timesheet` and `wtlib` under `$WTHOME/pi/pylib` are the directories where the application logic Python modules reside. In the event that a patch is required for a Timesheet software problem, the likely resolution will involve replacing a module in one of these directories. Do not attempt to manipulate any files in these directories without the direction of Journyx Support. Normally all patches are provided with an installer script which copies the files to the right directories.

`SavedReportFiles` and `jtfs`, `special` directories used to save user information in previous releases are no longer used as of Version 8.9 See the notes on the `restoredb` command below for more information.

Windows Directory Layout

The product is installed to `c:\Program Files\Journyx` by default. That location is saved as the `%WTHOME%` environment variable. There is no "sourcing the setup file" under Windows. The following directory locations are relative to `%WTHOME%`.

Location	Purpose
<code>jwt\bin</code>	Location of the setup file and CLI tools
<code>python\python.exe</code>	Python language executable
<code>jwt\timesheet</code>	Application Framework modules
<code>jwt\cgi</code>	Application "Screen" modules
<code>jwt\wtlib</code>	Application library modules
<code>jwt\tmp</code>	Timesheet logging directory
<code>LightTPD</code>	Lighttpd web server root directory
<code>jwt\tmp\lighttpd</code>	Lighttpd web server logs
<code>jwt\htdocs</code>	Static web server files
<code>pgsql\data</code>	PostgreSQL data directory
<code>jwt\tmp\pglogs</code>	PostgreSQL logs

`%WTHOME%\jwt\bin` is the directory where most system administrative tasks are performed. A link to open a command prompt in this directory is installed under the Start -> Journyx Timesheet menu.

Logging Locations

Care should be taken with logging since unlimited growth of any log file can cause problems. A log rotation strategy is strongly advised.

There are three locations where log files may be found in order to help diagnose problems.

`$WTHOME/tmp` is where Timesheet logs diagnostic and audit logging. On Windows the directory is `%WTHOME%\jwt\tmp`. Diagnostic logging should only be enabled for short periods, due to the large amount of data that can be generated. Diagnostic logging is usually only useful when performed under the direction of Journyx Support. The logs that may need rotating here are the `audit.log` and if diagnostic logging is enabled `debug.log`.

Lighttpd web server logs are found in `$WTHOME/pi/www/logs` on Linux and `%WTHOME%\jwt\tmp\lighttpd`. Both the `error_log` and `access_log` are kept there.

If the default PostgreSQL database is used, its logging is kept in `$WTHOME/pi/db/serverlog` on Linux, and at `%WTHOME%\jwt\tmp\pglogs`.

The installation process on Windows writes two separate logs:

- The “Installshield” log at `%TEMP%\JournyxTimesheetSetupLog.txt`. This file logs the basic `setup.exe` installation process. If this process fails, please send that log with your support request.
- The “Setup Web server and Database” script, which runs after the Installshield process reboots your computer, has a separate log at `%WTHOME%\IntegrationLog.txt`.

Administrative Topics

Installation Notes

Important note: If you are currently running with an earlier version of Journyx Timesheet and plan to migrate your existing data then it is absolutely critical that you follow the proper migration steps to preserve your existing data and assure a smooth transition to the new Timesheet version. Please read this web page before continuing:

http://community.journyx.com/Files/Upgrading_to_9.pdf

If you do not follow the migration instructions, your critical Time and Expense data may become unavailable or lost!

To avoid interruptions in service, Journyx strongly recommends that you install the new version of Timesheet on a separate machine. This will leave your production site available online while you test the new site. You should also use a separate database user/schema if using an external database connection. Leave your production database intact until the migration to the new version is 100% complete.

System Hardware and Software Requirements

For a minimal installation, for instance to see a sample install of the application and with the DB and application on a single machine, a minimum of 3 GB of RAM is suggested and 4 GB or more is strongly recommended.

Dual CPU or better hardware is also strongly recommended. For production installs please see this page for more detailed hardware recommendations for application servers and database servers.

<http://community.journyx.com/product-documentation>

If the machine will be hosting other web sites or applications besides Journyx Timesheet then the system resources will need to be increased appropriately. If you have any questions please email support@journyx.com.

You should also know that Journyx provides Timesheet in a Cloud or SaaS (Software as a Service) model where the application is hosted on Journyx servers and accessed through a secure SSL Internet connection. If hardware resources or system administration resources are a concern then you might want to discuss this option with your Journyx sales representative and have Journyx take care of all of this!

Operating System Requirements

You MUST use the 64-bit version of Timesheet if your operating system is a 64-bit edition. This applies to Windows and (in most cases) to Linux. Likewise, you must use the 32-bit version of Timesheet if you are using a 32-bit edition of Windows. If your system is large, with more than a hundred or so active users, you will in general get better performance under the 64-bit edition.

Journyx recommends the use of a Microsoft server operating system such as Windows Server 2008 for deployment in your enterprise Timesheet solution, or alternatively an enterprise level Linux system such as Red Hat Enterprise Linux. Installation to a Windows desktop operating system is only recommended for demonstration or test purposes. All versions of Windows and other operating systems may be used as clients to access the software through a web browser.

Journyx Timesheet 9.0 requires one of these Windows platforms to install:

- Windows Server 2003
- Windows Server 2008 or Server 2012
- Windows XP Service Pack 2 or higher (supports Lighttpd web server only)
- Windows Vista
- Windows 7 or 8

See also the “Additional Requirements for Microsoft Windows Servers” section below.

For Linux we currently recommend Red Hat Enterprise Linux (RHEL) version 5.5 or higher, or the equivalent CentOS version. There are a number of different binary distributions of Timesheet available for Linux – in general these are labeled with a “glibc” system library version that they should match up to. For instance RHEL uses glibc version 2.5, and so the Timesheet version for glibc 2.5 should be used. There are several other versions of Timesheet available for different Linux distributions and glibc versions. On Linux, you only need “root” access if you intend to run the web server on a privileged port such as 80. Otherwise the software can run under purely user-level permissions.

Additional Requirements for Microsoft Windows Servers

Web Server: On Windows, Journyx recommends using Microsoft Internet Information Server (IIS) as the web server for Timesheet. This feature can be found on your Windows installation media. Please see the "Configuring the Web Server" section below for details. The Lighttpd web server is included as an alternative and is fully supported. However if IIS is installed, Lighttpd cannot run on port 80 unless IIS is configured to not use that port.

Permissions: You must have local Administrator rights on the Windows system to install Journyx Timesheet. Your user must be a member of the local Administrators group or its equivalent. **Journyx highly recommends that you create a dedicated local administrator account and use that account to install and manage the application. Backups, patch and maintenance installs and general administration can only be performed by the installing user. To protect against problems when an installing user is not available in the future, avoid installing with a personal administrator account.** Once installed under IIS, the application normally runs under a dedicated Application Pool identity automatically generated by the system. When installing patches or updates to Timesheet, you should be logged into Windows using the same user account that you originally installed with (the “install user” account.)

Database Drivers: If you wish to use a Microsoft SQL Server database connection, you must install the **SQL Native Client** drivers on the Timesheet server. You must use at least the **2008 R2** version as mentioned below in “Configuring the Database on Windows”.

Internationalization: The following note applies only if you are installing under a non-Western European edition of Windows, or if you are installing under a Windows account with non-Western characters in the account name (e.g. an account name with Russian or Chinese characters).

If the above applies to you, you must use Windows Server 2008 or higher. In other words, this configuration is not supported under Server 2003 or Windows XP.

Additionally, under this scenario you are restricted to using external SQL Server or external PostgreSQL database connections. In other words, the internal PostgreSQL server option is not supported.

These restrictions do not apply to any data or account names inside the Journyx product itself; only the Windows account name. Journyx fully supports Unicode throughout the application itself.

External Database Setup

Journyx Timesheet requires a relational database to store your Time and Expense information. You may either use the provided enterprise-ready PostgreSQL database, or you may supply your own external database server.

Journyx provides an "Internal Database" option for Windows. The feature is selected by default and installs the PostgreSQL open source database. During the "Setup Web Server and Database" program (which runs after you install Timesheet and reboot) it will ask you whether you want to use the provided internal database or instead select an external database.

Anyone who wishes to use an external database such as Oracle or SQL Server must follow the instructions on this web page before installing. This will tell you how to configure your external database. These are general instructions that apply to all database types – you must also read the instructions for your specific platform linked below.

General Instructions (All Platforms) – Read First

http://community.journyx.com/Files/External_Databases_v9.pdf

Platform Specific Instructions

The currently supported external databases are:

- Microsoft SQL Server 2005 or higher on Windows Servers only. (Note: SQL Server 2000 is no longer supported.) Instructions:

http://ssl.salesforce.com/_ui/selfservice/pkb/PublicKnowledgeSolution/d?orgId=00D00000000hhh2&id=50100000000FkNd

Short form: <http://bit.ly/z72jMg>

Note: you must install the **SQL Native Client** drivers for Microsoft SQL Server **2008 R2** or higher (e.g. the SQL Server 2012 drivers) on the Timesheet server. You can use the 2008 R2 drivers to talk to a 2005 database. See “Configuring the Database on Windows” below for more info and download links.

- Oracle 10g or higher on Windows or Linux servers

http://ssl.salesforce.com/_ui/selfservice/pkb/PublicKnowledgeSolution/d?orgId=00D00000000hhh2&id=50100000000FkNx

Short form: <http://bit.ly/xTal3K>

Note: Oracle Instant Client 11.2 is required. If using a 64 bit operating system you must use the 64 bit Instant Client version.

- PostgreSQL 8.3 or higher on Windows or Linux servers.

<http://ssl.salesforce.com/ui/selfservice/pkb/PublicKnowledgeSolution/d?orgId=00D00000000hhh2&id=50100000000FkO7>

Short form: <http://bit.ly/yucOIW>

Note: Neither IBM DB2, nor MySQL are supported at this time.

Linux Installation Notes

You will download or receive a .tar.gz file containing the appropriate binary for your Linux system keyed to the glibc version. You can check your system's glibc version by running:

```
$ ls -la /lib/libc.so*
```

If you see something like this:

```
lrwxrwxrwx 1 root root 11 2010-08-27 09:58 /lib/libc.so.6 -> libc-2.7.so
```

Then you have glibc version 2.7.

Create a user on your Linux system dedicated to running Timesheet. For instance, create a user named "timesheet". **Journyx highly recommends that you create a dedicated account and use that account to install and manage the application. Backups, patch and maintenance installs and general administration can only be performed by the installing user. To protect against problems when an installing user is not available in the future, avoid installing with a personal account.**

Log onto the system as the Timesheet user. Now "untar" the installation file. You will end up with a directory called "jtime". Change into this directory and run the install script:

```
$ cd jtime
$ ./jtinstall
```

Then follow the interactive prompts. You may also run `./jtinstall --help` for a help screen listing available options. If you don't give these options you will be prompted about them. These are the most commonly used options:

`--path <directory>` Install location. This will become your \$WTHOME.

`--postgresql <DBURL>` Specify an external PostgreSQL database connection. See below for DBURL format.

`--oracle <DBURL>` Specify an external Oracle database connection. See below for DBURL format.

`--port <PORT>` Use the given port as the web server port.

`--license <KEY>` Enter a license key obtained from your Sales Representative.

`--externalfcgi <PORT>` Allow an external web server using the given FastCGI port number.

`--externalmemcache <HOST:PORT>` Use an externally managed memcached server at host:port.

The DBURL describes the database connection. For both external database choices (PostgreSQL and Oracle) they follow the exact same format. The whole thing must be double-quoted. Port is optional.

```
--postgresql "username/password@//host:port/instance"
--oracle "username/password@//host:port/instance"
```

For instance, assume you had an Oracle server running on the default port on the host `ora.company.com`. The Oracle Service Name is "orcl" in this case, and your username/password is `scott/tiger`. The DBURL would look like:

```
--oracle "scott/tiger@//ora.company.com/orcl"
```

If you select the internal PostgreSQL database option, you will be prompted for a database password. This password will be required for all connections to the database. See the Security chapter of this document for more information about internal PostgreSQL security.

External FastCGI web server (Apache) on Linux

You can use any web server on Linux that supports TCP-based FastCGI connections, including Apache and Lighttpd. The FastCGI functionality may be enabled by a separate module such as `mod_fastcgi` (available at http://www.fastcgi.com/mod_fastcgi/docs/mod_fastcgi.html).

First make sure your web server is installed and running at a basic level – for instance that you can view a sample `index.html` file included with it. Obtain and configure your secure SSL service if desired (see below.) Make sure the deflate module is either disabled or else only enabled on specific locations / resources.

Make sure a FastCGI module such as `mod_fastcgi` is installed and working. Read the installation instructions for the FastCGI module. Often the modules come with a simple test script to verify basic functionality.

Typically your Apache configuration file needs a line to load the FastCGI module like this:

```
LoadModule fastcgi_module /usr/lib/httpd/modules/mod_fastcgi.so
```

Install Timesheet using the `./jtinstall` script described above with the `--externalfcgi <PORT>` option. Pick any port (not currently in use) above 1024 such as 5020 for the FastCGI service. You will need to enter that port into the web server configuration file.

Add a virtual host configuration for your web server such as the following:

```
<VirtualHost *:80>
AddHandler fastcgi-script .fcgi .fcg .fpl .py .pyc
DocumentRoot /<WTHOME>/pi/www/htdocs
ServerName <SITE>
FastCgiExternalServer /<WTHOME>/pi/pylib -host 127.0.0.1:<PORT> -idle-timeout 7200
Alias /icons/ /<WTHOME>/pi/www/htdocs/image/
Alias /jtime/ /<WTHOME>/pi/www/htdocs/
Alias /jtcgi/ /<WTHOME>/pi/pylib/cgi/
ErrorDocument 401 /jtime/default401.html
ErrorDocument 403 /jtime/default403.html
ErrorDocument 404 /jtime/default404.html
</VirtualHost>
```

Replace <PORT> with the FastCGI port number you picked above. Replace <WTHOME> with the installation location from the `./jtininstall` script. Replace <SITE> with your desired site hostname, such as `timesheet.mycompany.com`.

The `ErrorDocument` lines ensure that unknown resource requests for this host generate the same responses as unknown resource requests under the `/jtcgi` scripts directory. You can customize these response files; see “Custom Error Handlers” under “Configuration and Tuning” below. If you customize these be sure to follow the instructions in that section and update your `ErrorHandler` lines with the customized locations.

You must also ensure that the “deflate” module is either disabled or else configured to exclude the Timesheet CGI directory `/jtcgi`. The Apache deflate module will interfere with the compression that Timesheet provides internally. This may result in the site being inaccessible, especially on certain browsers. The deflate module compresses content so it uses less bandwidth to transfer to the end user. If all of your users are on a LAN (local area network) then the deflate module may not provide any significant benefit. However if some of your users are accessing the site through the public Internet, then deflate may provide enhanced performance. However you must take care that the `/jtcgi` directory is excluded as that provides its own compression.

To completely disable deflate across the entire site, run this Apache configuration command:

```
a2dismod deflate
```

If you wish to leave deflate enabled, you must ensure that the `/jtcgi` virtual directory is excluded. Typically to do this you must edit `deflate.conf` (often found in `/etc/apache2/mods-available`) and comment out the `AddOutputFilterByType` line. This will disable deflate globally. You must then edit each `Location` directive of your other apps or resources and add a similar line, such as:

```
<Location /myotherapp>
  AddOutputFilterByType DEFLATE text/html text/plain text/xml
</Location>
```

Make sure that deflate is not enabled under the `/jtcgi` location.

Once the configuration is complete, restart the web server and attempt to access the Timesheet site at the `/jtime` URL.

Please also read the entire section marked “Security” below as many of these items apply to external Apache web server setups. In particular read the sections on enabling SSL and also disabling the HTTP TRACE / TRACK verbs.

Windows Installer Digital Signature

Journyx digitally signs the Timesheet for Windows installer files for your protection. However, the digital signature is not checked automatically for you. You can manually check the signature at any time (see below.) The purpose of the signature is to detect any alteration in the setup file, whether by accident (data corruption from network issues) or by malicious behavior.

To check the digital signature, simply locate the installer file in Windows Explorer (or My Computer on your desktop) and right-click on it. Click on “Properties”. Click on the “Digital Signatures” tab in the Properties dialog. Click on the entry for Journyx, Inc. then click Details.

If the file is clean (unaltered) it will say “The digital signature is OK.” If any part of the file has been altered or corrupted it will say “One of the countersignatures is not valid. The file may have been altered.” If you see that, contact Journyx Support immediately and obtain a new copy of the installer. Do not use an installer whose signature does not validate.

Windows Installation Notes

Journyx strongly recommends rebooting your machine before installing Journyx Timesheet. This may avoid problems with previous installations of other products that never completed. In addition, if you have an old version of Timesheet installed, you must:

1. Make a backup file with `backupdb` (described below)
2. Uninstall the old product
3. Reboot the system
4. Delete the old Journyx directory (save your backup file somewhere), such as
`C:\Program Files\Journyx` or in some cases `c:\Program Files (x86)\Journyx`
5. Install the new version.
6. Run the `restoredb` program to restore your data.

You must uninstall any old version of Timesheet, including Timesheet 8.x, and reboot before installing the new version as described above. Be sure to perform a backup of your data with the "backupdb" command. If you are running an external database such as Microsoft SQL Server, you should also perform a "native" database backup from its manager program. Be sure to reboot the server as directed after uninstalling the old version, before you run the new installer.

The Journyx Timesheet installer includes a 'Quality Feedback' feature. This feature sends installation "milestone" information to the Journyx web site as a simple web page request to help Journyx evaluate installer quality and to identify areas of technical concern.

No personally identifying information is collected or transmitted by the Quality Feedback feature. The feature may be disabled by un-checking the box on the first screen on the Installer.

After the installer is complete, you will be required to reboot your computer. You have the option of rebooting later, but you cannot use Journyx Timesheet until you reboot. When you reboot and log in, the installer will run a final setup action that initializes Timesheet. This action will run only once, and you must complete it before you can use Timesheet. Your computer may appear to be "frozen" or "thinking" temporarily when the InstallShield dialog appears after rebooting.

This setup action will launch a script (console) window to configure your web server and database for Journyx Timesheet. This script can be run at any time by clicking Start -> Programs -> Journyx Timesheet -> "Setup Web Server and Database."

Configuring the Web Site on Windows

About Web Server choices

Journyx Timesheet is a web application and therefore requires a web server to run. Journyx recommends using the Microsoft Internet Information Server on the Windows platform. Journyx does **not** provide IIS; it is available on your Windows installation media. See below for installation instructions for IIS.

As an alternative, Journyx provides the open source Lighttpd web server. This allows you to run Timesheet without IIS and is fully supported. However if IIS is also installed, Lighttpd may not be able to grab the default web port (80) and must be installed on a different port. IIS configuration can also be changed to not use port 80.

If the "Setup Web Server and Database" program detects that you do not have IIS, it will automatically use Lighttpd. If you install IIS at a later time, you can rerun this program to choose IIS. You can switch back and forth without affecting your actual Timesheet data..

Installing IIS

To install IIS, first obtain your Windows installation media. Note that IIS version 6 is required. This precludes the use of IIS on Windows XP. Lighttpd can be used there instead.

Once you have your Windows installation media, go to Start -> Control Panel -> Add / Remove Programs. Then select 'Add / Remove Windows Components.' The exact location of the IIS feature may vary depending on your operating system. If you have any doubt, please contact your system administrator or support@journyx.com.

In addition to the core WWW server component, you must install the following IIS components that are not enabled by default:

- ISAPI Extensions
- IIS Metabase and IIS 6 configuration compatibility (Vista, Server 2008, and Windows 7 only)

This component is not required, but highly recommended:

- Dynamic Content Compression

This feature is used in lieu of the GZIP_ENCODING=Yes setting used on Lighttpd. The GZIP_ENCODING setting has no effect when running under IIS – content compression is enabled or disabled through the Dynamic Content Compression feature of IIS.

Installing Timesheet to an IIS server

IIS can serve multiple separate web sites, possibly on different network addresses or ports. Journyx Timesheet will only install on an **existing** IIS site. If you want Timesheet to be served from a particular port number, you **MUST** first configure a web site with that port. Journyx Timesheet is not capable of creating a new IIS web site for you; it can only install to an existing site.

Note that only certain editions of Windows allow you to create multiple web sites. If you only have one web site available, the Timesheet Integration will **NOT** prompt you for a web site. It will automatically select the only available site. If multiple sites exist, you will be prompted to select the site for Timesheet.

Timesheet will create three new virtual directories on your chosen web site:

- /jtime (static files)
- /icons (images)
- /jtcgi (application scripts)

Note that you cannot have any other applications on that web site that use virtual directories of that name, particularly /icons.

You may remove Timesheet from a web site at any time by simply deleting these virtual directories in the IIS Administration tool. You can then re-run the Integration script (see above) and select a new web site where Timesheet will live.

There is only one IIS-specific configuration option you should be aware of. That is the “Maximum Worker Processes” setting of the Journyx Application Pool. As the name implies this controls the maximum number of worker processes (daemons) at any one time. This is the number of requests that can be simultaneously handled – any additional requests are put into a queue (line) and serviced as worker processes become available. By default this is set to 8 processes plus 4 for each additional CPU in your system. Idle worker processes are automatically stopped by IIS. You can set this to as much as 20 processes per CPU but setting the number too high may cause excessive memory usage.

To change this setting, open the “Internet Information Services (IIS) Manager” program from Administrative Tools or Computer Management. Open Application Pools then select the Journyx pool and select Advanced Settings. Maximum Worker Processes is under the “Process Model” category.

Installing Timesheet internal web server on Windows

If IIS is not available on your system, or if Timesheet is unable to install to IIS for some reason, it will fall back to using the included Lighttpd web server.

Lighttpd server does not have any configuration options except for network port. The normal HTTP (web server) port is 80.

However if that port is in use, then you will be prompted to select an alternate port (defaulting to 8080.) If any port besides 80 is selected the URL used to access Timesheet must include the port number. This is true for IIS as well. For instance,

```
http://your-server:8080/jtime
```

In any case, this program will create a shortcut file with the correct address/port for Timesheet. You will find it under Start -> Programs -> Journyx Timesheet -> 'Journyx Timesheet (login)'.

Note that the Timesheet internal web server always binds the given port on ALL available network interfaces; if you need to limit Timesheet to a specific network interface then you must use IIS.

Once the web server is set up, the program moves on to set up the database connection.

Configuring the Database on Windows

Be sure to read the requirements and other information in the “External Database Setup” section of this document above before installing to an external database.

If the Internal Database feature is selected, the only option to configure is a password for the database itself. If using the internal database, please also read the “Internal Database (PostgreSQL) account security” section in the Security chapter of this document.

If using an external database, you will be asked which type (SQL Server, Oracle, or PostgreSQL) and the connection information.

If you wish to connect to a SQL Server 2005 (or later) database, you must install the “Native Client” drivers provided by Microsoft on to the Timesheet server. Currently you must install the **SQL Server 2008 R2** version of these drivers or any later version such as 2012.

You can download the SQL Server 2012 Native Client drivers from this link:

<http://www.microsoft.com/en-us/download/details.aspx?id=29065>

Click on “Install Instructions”. Search for “Native Client” until you find the section titled “Microsoft® SQL Server® 2012 Native Client”. If you have a 32-bit edition of Windows, chose the X86 Package download link. If you have a 64-bit Windows, chose the X64 Package. After the download completes, run the installer on the Timesheet server. If necessary, go to Start -> Journyx Timesheet -> Setup Web Server and Database.

Journyx Timesheet uses ODBC to connect to SQL Server. You can either create a pre-defined ODBC connection, known as a System DSN or Data Source Name, or you can just enter the connection information directly without creating a System DSN. After you select the SQL Server option in the setup program, you will be shown a list of existing DSNs. You can either choose an existing DSN, or chose the special “(None)” option to fill in your connection info directly.

Connecting to an external PostgreSQL or Oracle database does not require any special drivers. Just provide the usual connection information such as host, port, database name, username and password.

Configuring your Timesheet data

After your web site and database are configured, the Integration script will ask you if you want to set up a new 'clean' Timesheet site with the default initial configuration, or if you wish to cancel. Either way, you can always restore a site from an existing backup file later by opening the Timesheet Command Prompt and running `restoredb` as described in the Disaster Recovery section below.

You can also click Cancel at this point, and nothing will be done. It will not modify the database at all, but your Timesheet site will not work correctly in most cases. This option should only be used when following instructions from Journyx Support.

Quickstart

When the Web Server and Database Setup completes, you will be taken to the Timesheet Quickstart Configuration screen. However, you must first log in to the product. If you did a 'clean' install, your Timesheet administrator account login name and password will be the same as your Windows username. If you restored a database, your username and password will be the same as in your previous site.

Multiple Application Servers (Clustering)

From time to time, a single application server may not be robust enough to serve all customer needs. Journyx products support multiple application servers talking to the same database back-end with some custom configuration. This allows you to scale up the number of users by adding additional hardware.

Clustering Overview

The most common configuration consists of multiple physical machines (servers):

- 1) One or more web servers, typically Apache running `mod_fastcgi`. The web servers are mostly used to serve static files such as images. Often a single web server is sufficient even for a very large install.
- 2) One or more memcached servers. Please note that multiple memcached servers only increases performance in some limited scenarios, and does not provide enhanced uptime or failover capabilities. If a single memcached server in your cluster is offline, this may severely impact performance across the entire app cluster. In general we recommend all customers run a single memcached server instance unless directed otherwise by Journyx Support.
- 3) One or more application servers with shared network storage for the WTHOME install. The application servers should be the most powerful systems with the best CPUs available as these systems perform the bulk of the application work.
- 4) A database system, which may be internally clustered or have fail-over systems. Configuring a clustered database is beyond the scope of this document; please consult your DBA and/or database vendor training. When running in a clustered environment (multiple app servers) you must run an external database instance; in other words, you cannot use the "internal" copy of PostgreSQL when using multiple app servers.

Understanding the Journyx Installation

Journyx Timesheet and Journyx ProjectXecute install fully into a single directory known as `%WTHOME%` (on Windows) or `$WTHOME` (on Unix) (hereafter referred to as `WTHOME`). This is usually `C:\Program Files\Journyx\` on Windows and the user-specified directory (usually under the user's home directory) on Unix variants.

Note: This document assumes that you are talking to an external database server (PostgreSQL, SQL Server or Oracle) and not using the internal PostgreSQL database which ships with Journyx products.

Beneath WTHOME are several directories of interest.

WTHOME/pd and WTHOME/pi contain product libraries and logic files which can either be shared (via NFS, DFS or similar) or installed separately on each application server.

WTHOME/doc contains product documentation files which can either be shared (via NFS, DFS or similar) or installed separately on each application server.

WTHOME/tmp contains temporary files and ongoing logs used by the Journyx products. **This directory MUST be shared among application servers. This directory CANNOT be configured.**

There are 2 files of interest within WTHOME itself: config and config.lock. These files tell the Journyx products “how to run,” and **MUST be shared among application servers.**

Additionally, Journyx products utilize memcache server technology. For a shared installation such as this, you will be required to talk to one or more external memcache servers.

Installing on Multiple Application Servers

Installation of Journyx products to multiple application servers is relatively-simple.

- 1) Setup a shared directory on your network which your multiple application servers can access which will be where you install the Journyx product (WTHOME directory).
- 2) Install an external memcached server (memcached.org) on your network where the application servers can talk to it. It is acceptable to use Journyx internal memcached server on one of the application server(s). If you have multiple servers available to dedicate to memcached, you can run one memcached on each server. For large (>1000 user) installations it is recommended to have at least one server fully dedicated to memcached.
- 3) Install your Journyx product on a single application server to an external database server per standard product instructions to the directory noted above. On Windows it is often best to map that shared directory to a “network drive.”
- 4) Install your multiple-application-server license key as provided by Journyx Accounting.
- 5) Repeat step 2 on each application server, installing to the same directory.
- 6) NOTE: When you have completed this, you must select the “master” machine for the purposes of running Journyx-based “scheduled tasks” (cron). Pick a single application server to be the “master” and run this command to enable cron: `setlocalcron --on`
- 7) In the event that the cron host is offline for more than a couple hours, `setlocalcron` can be run to transfer the function to a different machine.
- 8) Once completed, stop each application server and modify the WTHOME/config file line which says:

```
MEMCACHED_SERVERS=1.2.3.4:11211
```

to have the IP:Port of your accessible memcached server. If this line does not exist in your config file, increment the number at the top of the config file by 1, and add the line to the bottom of the file. If you are using more than one memcached server, the addresses of all servers should be comma separated.

Please note: as mentioned in the Security section of this document, please ensure that your firewall is configured to deny any TCP/IP connections to the memcached servers other than from the Timesheet app servers.

- 9) Restart the application server(s).

Multiple Application Servers on Windows

Running multiple application servers on Windows is mostly the same but there are a few differences to note.

1. All patch upgrades including Support hotfixes will need to be installed separately on each node in the cluster.
2. In addition the `/jtcgi` virtual directory under IIS must be set to have Anonymous Authentication where the identity of the Anonymous User is set to "Application Pool Identity". Normally this is done for you automatically but you can check that in the IIS Manager.
3. If you use Windows Authentication to connect to your SQL Server database in a Windows Domain site, you will also need to add `domainname\machinename$` (for each machine in the cluster) to the list of authorized accounts inside SQL Server Manager. If you are NOT using Windows Domains, you'll need to add `IIS AppPool\Journyx` to the list of authorized SQL Server users.
4. You must ensure that "Cron" (scheduled tasks within Timesheet) is only running on a single server. The setting `RUN_LOCAL_CRON=<servername>` must be in the config file on all servers. The value of `<servername>` should be the one server where Cron will be run. E.g. "apphost01".

Load Balancing

Using the example above, let us say that we installed a Journyx product to 3 application servers.

- a) The installation directory is a shared network directory which all 3 application servers can access with full control, and it is mapped to "J:"
- b) The application servers are named "app1," "app2," and "app3."
- c) Your domain is mydomain.com

Now that you have completed the steps above, anyone who can access:

<http://app1.mydomain.com> or <http://app2.mydomain.com> or <http://app3.mydomain.com> can talk to the same Journyx installation which talks to the same back-end database server.

However, if app1 and app2 are unavailable, this installation would require your users to know that and to choose to connect to app3.mydomain.com to have an available application server. Journyx also supports "sitting behind" a load balancer with some minor configuration changes.

Understanding Load Balancing

Using the example above, you would install a "load balancer" to act as your "virtual server," where your 3 "real servers" are app1, app2 and app3. For this example, we will name the load balancer "journyxapp.mydomain.com"

When your load balancer is properly configured, a user requests a resource from journyxapp.mydomain.com, and the load balancer takes the request from there, routing it to one of the 3 available application servers, depending on the rules you have configured.

Configuring Journyx products to properly report URLs

Sometimes in emails or product links, Journyx products will provide HTTP addresses for ease of use. In the example above, your FQDN would be set to app3.mydomain.com (assuming you installed app3 last). Behind a load balancer, however, you would want those links to say "journyxapp.mydomain.com" instead. This is configurable within the Journyx product under System Settings -> Server and Email. An example of these configuration settings is shown below.

SMTP (outgoing) email server:

mymailserver.mydomain.com

SMTP (outgoing) email server port:

25

'From' address for email sent by the system:

timesheetadmin@mydomain.cor

Number of seconds to wait before giving up on a mail server connection:

30

Miscellaneous Settings

'Fully qualified domain name' of this server:

journyxproduct.mydomain.com

Preferred email format:

Both

Alternate prefix for links inside email notifications:

http://journyxproduct.mydomain.com/jtcgi/

Maximum number of rules that can be in a policy (max. 30):

12

Number of remaining license seats under which a warning message is generated on management screens

5

Easier installation on Unix variants

If you are using a Unix variant rather than Windows, and you are sharing WTHOME via NFS or similar, the installation can be simplified greatly. Please note that you must still run an external database in this situation; you cannot use the "internal" PostgreSQL install provided by Journyx when using multiple application servers.

- 1) Install your external memcached server and configure and start it per documentation.
- 2) Install your Journyx product to the shared directory using the --externalmemcache IP Address:port syntax for your external memcache server and the external database flags per Journyx installation documentation.
- 3) Test the installation
- 4) On server 2, if you are using a local user (as opposed to LDAP, NIS or similar), create a user named the same as the user on the first application server with the same UID (user ID).
- 5) Mount the shared directory in the same location
- 6) Start the Journyx application server on the second application server.

Security Issues

General Security Issues

To ensure the continued secure operation of your Timesheet web site, there are a few simple steps that you should take. As with all software, security is achieved by presenting as few opportunities as possible for unauthorized access to your systems. Your IT staff should be familiar with configuring firewalls and other tasks required to implement secure networks and web services. You should keep your operating systems and other software components (especially external web servers and databases) up to date with the vendor's security patches.

In general we recommend running the Timesheet web server exclusively on your intranet (internal corporate network). Off-site or external users who need to access your Timesheet site over the public intranet should use a secure encrypted VPN (virtual private network) solution to first connect to your intranet, and from that access

the Timesheet site. If you choose to configure Timesheet so that it's directly accessible from the public Internet, we strongly recommend using SSL security (see below) and locking down the application server as much as possible.

In particular, make sure that only the public web server port (usually 80, or 443 for https/SSL) responds to connections from the public Internet. Your firewall solution should be configured to block all other requests. In particular the other TCP based services that Timesheet depends on, such as the internal PostgreSQL database, and memcached, should not be allowed by the firewall to accept connections from the outside.

If using an external memcached server, it is essential that you read the "Important Security Note for External Memcache" section below.

Internal Database (PostgreSQL) account security

Timesheet 8.8 and higher now requires a password to be set on the PostgreSQL system when using the internal database option on either Windows or Linux at installation or patch-upgrade time. This new prompt does not apply if you connect to an external PostgreSQL system maintained by you (in which case a password was already required). The password you select is automatically applied to both the standard "journyx" account and to the read-only "jxreport" account. You must select a single password for both accounts at installation time but you may change the password for either account later with the "changedbpw" command-line program. The password will be required for all connections to the internal PostgreSQL server. The password is stored in a lightly-encrypted format in the \$WTHOME/config file. That file should normally only be readable by the OS install user. If you forget the password and need to retrieve the plaintext version, you can run this command as the OS install user:

```
$ python -c "from wtconfig import options; print options['e_DB_PASSWORD']"
```

Although a read-only reporting account named jxreport is created automatically for you, by default you can only log into this account from the same local machine for security reasons. If you wish to grant external reporting access to a different host, please follow the instructions below.

Remember you can use the `changedbpw` command at any time to change the password for either the main account (journyx) or the reporting account (jxreport). Use `changedbpw -h` to see detailed help info. If you are using an external database connection, `changedbpw` only changes the stored password in the Journyx config files, not the actual password on the external DB account.

However if you're using the internal PostgreSQL database, the `changedbpw` command changes both the stored password and the actual 'journyx' account password in PostgreSQL. Use the `changedbpw -R` option to change the actual account password of the jxreport read-only external reporting account. The Journyx application server does not directly use the jxreport account.

External Reporting

Many users will wish to provide direct external reporting access to the Journyx Timesheet SQL database. For instance, you can write your own customized reports using a third-party product like Crystal Reports. It is possible to use the main database account that the Timesheet application server uses. However this practice is not recommended because that account has read/write access and could accidentally modify, overwrite or destroy critical data due to a programming or security error. Journyx does not support, recommend, or allow direct database modification. Performing a direct database modification could violate your support contract with Journyx and render your site inoperable.

Instead, Journyx recommends creating a dedicated read-only account for external reporting access. This account should be set up by your DBA (Database Administrator) and granted only read (SELECT) permissions on all Timesheet tables and views. The exact procedure depends on your database system and version and should only be performed by a DBA who possess sufficient knowledge of database security fundamentals. Consult your database system vendor's documentation for further details on granting read permissions to other users.

External Reporting and the internal PostgreSQL database

As of Journyx Timesheet and ProjectXecute version 8.8 or higher, a read-only external reporting account is automatically created for you as described above when using the internal PostgreSQL database option. However this account can only connect to the PostgreSQL server from the same local host by default. To enable true external reporting from a different host, you must take some additional steps to grant access.

- A. Make PostgreSQL listen on external network interfaces. By default the internal PostgreSQL only listens on the localhost (127.0.0.1) interface for security reasons.
 1. Find the postgresql.conf file.
 - a. On Linux this is in `$WTHOME/pi/db/data/postgresql.conf`
 - b. On Windows this is in `%WTHOME%\pgsql\data\postgresql.conf`
 2. Edit the postgresql.conf file in a text editor (using a programmer's editor like vi, Emacs, Notepad++ or similar is recommended.)
 3. Search for "listen_addresses". You will probably find it on a commented-out line.
 4. If there are no instances of a non-comment listen_addresses line, add a new line that looks like the following. If there is already a non-comment listen_addresses, modify it to look like this:


```
listen_addresses = '*'
```

 That will listen on ALL network interfaces. You may instead want it to only listen on a specific network interface (i.e, your intranet network.) In that case change it to list the IP address of the correct interface, such as:


```
listen_addresses = '192.168.1.5'
```

 You can also use a comma-separated list of interfaces. The exact location of the line within the file is not critical, but you may consider putting it just above the commented-out listen_addresses sample.
 5. Save changes in the postgresql.conf file and exit the editor.
 6. While you are in here, note the port= line. You will need this later to connect. On Windows the port is usually 5532 but on Linux the port varies depending on which ports you selected at install time.
 7. You can either restart the PostgreSQL server now (see below) or wait until editing the pg_hba.conf file in step 2 below.

- B. Edit the pg_hba.conf security file to allow remote connections over the jxreport account. This file controls who is allowed to connect to which database.

1. Locate the pg_hba.conf file.
 - a. On Linux this is in `$WTHOME/pi/db/data/pg_hba.conf`
 - b. On Windows this is in `%WTHOME%\pgsql\data\pg_hba.conf`
2. Edit the pg_hba.conf file in a text editor.
3. There are comments near the bottom of the file that tell you what to do. Follow these instructions. Generally you will add a line like this:

```
host    journyx          jxreport          192.168.1.100    md5
```

This sets a **host** based access role allowing access to the **journyx** database on the **jxreport** account using an **md5** encrypted password from the client host **192.168.1.100**. Usually the only thing you need to change here is the IP address of your external reporting client; i.e, replace 192.168.1.100 with the actual IP address of your reporting host. If you need to enable multiple report clients, you can either create a separate line for each host (updating the IP address as necessary on different lines) or else you can use a single line with an IP netmask to allow a range of addresses. For instance the host specification 192.168.1.0/24 will allow all machines on the 192.168.1.x network to have access.

If you are using Timesheet 8.7 or earlier, the read-only jxreport account will not exist. You can use the journyx account instead, but be careful not to make changes to the database through that account.

This page has complete documentation on the pg_hba.conf file:

<http://www.postgresql.org/docs/8.3/static/auth-pg-hba-conf.html>

- C. Now that you have told PostgreSQL to listen on external network addresses and have added appropriate access rules, you must restart the PostgreSQL service to have the changes take effect.
- On Linux, simply run the command `wstop; wstart`
 - On Windows, open the Services control panel. Typically you can do this via Start -> Run -> `services.msc` or find it under the Control Panel -> Administrative Tools window. Find the service named **PostgresqlJournyx** and right-click it and select Restart (or Start if not already started.)
- D. If the service fails to restart, check that you have the correct syntax in your PG config files. You can check the PostgreSQL service log for error messages. On Linux this is in `$WTHOME/pi/db/serverlog`. On Windows these messages appear in `%WTHOME%\jw\temp\pglogs`.
- E. You should now be enabled for external access. Before running the main reporting program, attempt to verify a simple connection such as with the ODBC driver or `psql` command. If your external reporting host is Windows, you may need to obtain a third-party PostgreSQL ODBC driver which is not provided by Journyx. You can obtain one such driver from here: <http://pgfoundry.org/projects/psqlodbc/> (Note: Journyx does not warranty or provide support for PostgreSQL ODBC drivers.)

When attempting to connect make sure you use the correct PostgreSQL server port number obtained in step 6 above. You will need these parameters:

- **port:** obtained in step 6 above, default is 5532 for Windows servers.
- **host:** the hostname or IP address of your Timesheet / PostgreSQL server.
- **database:** This is always "journyx" for the internal DB.
- **user:** always "jxreport" for the internal DB (version 8.8 or higher) otherwise "journyx"
- **password:** you supplied the password at install or upgrade time. The password may be different on the jxreport account than on the journyx account. The `changedbpw` can be used to change the jxreport account password. See the section on Internal Database Account security above for details.

HTTP Trace and Track verbs

Most security professionals recommend disabling the HTTP verbs TRACE and TRACK as they are not essential to program operation and may inadvertently reveal configuration information about your server.

The Lighttpd web server provided with the Timesheet service automatically disables TRACE and TRACK so there is nothing further to do there.

Microsoft IIS web server also disables these by default. If you have enabled this on IIS you may wish to turn it off. Please consult Microsoft's documentation for instructions.

To disable TRACE and TRACK in Apache versions 2.0.55 or higher, or 1.3.34 or higher, simply insert this line in your Apache configuration file under each virtual host:

```
TraceEnable off
```

Secure Sockets Layer (SSL) configuration

Secure Sockets Layer (SSL) is a feature of your web server that securely encrypts all communications with the end-user. Step-by-step instructions for configuring your web server are beyond the scope of this document. You should obtain and follow the standard instructions for SSL from your web server provider.

For instance, here are instructions for setting up SSL on IIS 7 (Windows Server 2008.)

<http://learn.iis.net/page.aspx/144/how-to-set-up-ssl-on-iis-7/>

Here is the manual for Apache 2.2 and SSL:

<http://httpd.apache.org/docs/2.2/ssl/>

You will need to obtain and pay for a security certificate for your organization.

For Apache you would typically have a configuration file section like this:

```
<VirtualHost *:443>
AddHandler fastcgi-script .fcgi .fcg .fpl .py .pyc
DocumentRoot /<WTHOME>/pi/www/htdocs
ServerName <SITE>.company.com
FastCgiExternalServer /<WTHOME>/pi/pylib -host 127.0.0.1:<PORT>
Alias /icons/ /<WTHOME>/pi/www/htdocs/image/
Alias /jtime /<WTHOME>/pi/www/htdocs/
Alias /jtcgi/ /<WTHOME>/pi/pylib/cgi/
SSLEngine on
SSLCipherSuite
ALL:!ADH:!EXPORT56:RC4+RSA:+HIGH:+MEDIUM:+LOW:+SSLv3:+SSLv2:+EXP:+eNULL
SSLCertificateFile /etc/httpd/ssl/host.cert
SSLCertificateKeyFile /etc/httpd/ssl/host.key
</VirtualHost>
```

Replace <WTHOME> with your \$WTHOME installation path, and <SITE> with the Timesheet site name (such as timesheet.mycompany.com) and <PORT> with the FastCGI port number from the --externalfcgi <PORT> command when you installed Timesheet.

Generating a Self-Signed SSL Certificate

For testing purposes only you may wish to create a self-signed SSL certificate. This allows you to configure and test secure SSL mode without buying a real certificate from Verisign, Thawte, or another Certificate Authority (CA). However a self-signed certificate doesn't provide true security and your users will be warned about an insecure (self-signed) certificate. You should only use a self-signed certificate for testing purposes. Here's how to generate a self-signed certificate on a Linux system.

First, go to a working directory where you will store your self-signed certificate.

```
$ cd $HOME/self-signed-cert
```

Next, generate a server key file:

```
$ openssl genrsa -des3 -out server.key 4096
```

If you do not have the openssl program available, you may need to install it from your Linux distribution.

Now create a server certificate:

```
$ openssl req -new -key server.key -out server.csr
```

Answer the questions as prompted. You can edit the default values in `/etc/ssl/openssl.cnf`

Now sign the certificate making it valid for 365 days.

```
$ openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

Create a version of the key that doesn't require a password. (Note: make sure no one else has access to this file.)

```
$ openssl rsa -in server.key -out server.key.insecure
$ mv server.key server.key.secure
$ mv server.key.insecure server.key
```

Now create the final pem file. This is what you use with a web server like Apache or Lighttpd.

```
$ cat server.key server.crt > server.pem
```

The `server.pem` file can now be used with Apache or Lighttpd. Remember, this is a self-signed certificate and should only be used for testing purposes.

Enabling SSL in Lighttpd on Linux

Journyx provides the Lighttpd “internal” web server with the product installer. Normally this operates in “regular” HTTP mode, but it can be configured to operate in SSL secure mode (HTTPS). However Journyx recommends that for production HTTPS use that you should use either Apache or Microsoft IIS. Journyx provides these instructions for configuring SSL on Lighttpd for your convenience only. SSL cannot currently be used on Lighttpd under Windows. Please also note that using standard HTTPS port of 443 is not currently possible with the internal Lighttpd – your users will have to include the port number in the URL they used to access the site. If you must use the standard HTTPS port of 443, then you must use either Apache or IIS.

1. Make sure you have Lighttpd 1.4.28 or higher for Linux. You can check the Lighttpd version provided with Timesheet by running:

```
$WTHOME/pd/Linux/lighttpd/sbin/lighttpd -v
```
2. Log into the Linux account where you installed Timesheet (“install admin” account).
3. Copy your SSL certificate pem file to:

```
$WTHOME/pi/www/conf/server.pem
```

If you do not have a SSL certificate, you can create a self-signed certificate using the instructions above. However a self-signed certificate should not be used for production servers.
4. Run “`chmod 400 $WTHOME/pi/www/conf/server.pem`”
5. Stop the Timesheet service with the “`wstop`” command
6. Edit the Lighttpd configuration file at `$WTHOME/pi/www/conf/lighttpd.conf`
 Insert the following two lines anywhere in the file (at the end is fine):

```
ssl.engine = "enable"
ssl.pemfile = "/home/jtime/jt/pi/www/conf/server.pem"
```

7. That is assuming your WTHOME install location is `/home/jtime/jt`; edit the above line with your actual install location.
8. Find the line that says “`server.port =`”. All users will have to include the port number in the url, like:
<https://yoursite.example.com:29700/jtime>
9. Save changes in the editor and exit the editor.
10. Start the Timesheet service with “`wstart`”

11. Tell Timesheet that we are now in secure HTTPS mode with this command:
`setkey HTTPS_STATUS=on`

Configuration and Tuning

The master configuration file is kept in the installation directory. Its name is simply `config`. It is advisable to make a backup of this file if you ever modify it and save it as `config.save` or some other name. If the `config` file is lost due to accidental deletion or corrupted the site will be down until it is restored and restoration is difficult.

Normally changes to this file should be made through the `setkey` command line program. `getkey` can print individual keys or the entire config file.

This file can be edited by hand but you must exercise extreme care in doing so. The first line of the file has a count of the number of lines expected in the file and that number must not change or the file will be considered corrupted. Never edit the file when Timesheet is running. Shut down the site before editing the file.

Prior to Timesheet Version 7.1 all configuration options were specified in the file-system resident config file. In Version 7.1 and later, configuration options with application-wide scope are moved into the database to allow better support for multiple application server instances. Thus, configuration options that remain in the config file apply to the local machine only. Options in the database apply application wide. Database connect information also still resides in the config file so the application can perform the initial database connect at start up.

General Options

Any attempt to manually change the configuration parameters below should be performed under the direction of Journyx Support. As mentioned above, in Version 7 and later, some of the configuration options are kept in the database, and cannot be modified via the config file. For these options the `getkey` and `setkey` command line interfaces can be used to set the configuration variable. Issuing the `getkey` command with no arguments gives a list of the values, along with their current setting.

In general, you rarely need to modify the config file once the site is up and running. Values for the `WT_DEBUGLOG`, `WT_DEBUG` and `DATABASE_BUSY` are controlled automatically in response to administrative actions such as turning on logging in the Timesheet Administration interface (Browser interface). The command line operations `backupdb` and `restoredb` which are used to backup and restore (see the Disaster Recovery section below for more information on these) will disable users web access while Timesheet backup/restore are performed and set the `DATABASE_BUSY` flag to do so.

`DATABASE_BUSY` can also be set via the Timesheet Administration Interface when Timesheet Administrators wish to perform operations and configuration using the browser interface while locking out general user access. This is accessed by Timesheet Administrators via the browser interface Administration/Web Interface Lockout link.

Performance and Memory Options

When running under FastCGI configuration (as with Lighttpd and Apache web servers) there are several important settings that have effects on performance and memory utilization. Running under Microsoft IIS uses a different architecture but has some similar settings described below. FastCGI is a protocol where the web server can communicate with the application server processes (the "daemons.") The daemons are pre-spawned so they are ready to handle web requests as they come in. The daemons handle the application web requests and generate the responses. The daemons are optimized for performance and cache various objects that improve application response. Running too few daemons can starve the system for workers to respond to burst web requests. Running too many, however may result in excessive memory utilization. Best performance is achieved by a balance of the number of daemons derived through observation of system behavior.

The following 3 FastCGI settings in the config file are the most critical:

```
FASTCGI_MAX_SPARE=<number, default=6>
```

```
FASTCGI_MIN_SPARE=<number, default=3>
```

```
FASTCGI_MAX_CHILDREN=<number, default=18>
```

These three options work similar in concept to similarly named settings in Apache. `FASTCGI_MAX_SPARE` sets the maximum number of “idle” daemons allowed at any given time. Any extra idle processes (those not actively handling a web request) will be shut down to conserve memory. `FASTCGI_MIN_SPARE` is the minimum number of idle processes. If all of the FastCGI daemons are engaged handling processes, the service will launch additional processes to ensure there are always several idle processes available to handle spikes in traffic. Finally, `FASTCGI_MAX_CHILDREN` controls the upper limit on all FastCGI daemons. It is the absolute maximum number of daemons that will run at any time. You should only increase that number if you are running at least a quad-core CPU with at least 4 GB of memory.

Please note that increasing `FASTCGI_MAX_CHILDREN` can increase the number of connections made to your database. If you are using the internal PostgreSQL database you may need to increase the number of allowed connections in the `postgresql.conf` file. Doing so may trigger additional resource requirements (e.g. `SYSV` shared memory) and may require adjusting your operating system settings per your vendor’s instructions.

Under the IIS web server there are no exactly equivalent settings, but the “Maximum Worker Processes” setting of the Journyx Application Pool controls the maximum number of daemons. See the “Installing Timesheet to an IIS server” section above for details.

Other settings:

`MAX_CHILD_REQUESTS=<number, default=2500>` This is the maximum number of web requests that each individual daemon will handle before being restarted. Normally you do not need to change this, but setting to smaller numbers can help keep the caching memory utilization down. In general, it should not be set lower than 1000

`MEMCACHED_MEM=<number of megabytes, default 128>` This is the amount of memory to allocate to the internal Memcached process. This is a shared cache that the daemons use to store working copies of database records and Python objects to accelerate the web site. 128 MB is sufficient any small to medium size site with approximately 200 active users. Larger sites may wish to set this to a higher number, or run an external memcached on a different host. Do not set this number so large that the system begins “swapping” under normal traffic or using the “page file” excessively. Very large sites may need up to 512 MB.

`LOCAL_CACHE_CLEAR_MINS=<number of minutes, default 5>` - this setting determines how often the “daemon” worker processes clear out their local caches. The default is 5 minutes. Sites with very high traffic may need to lower this. The lowest allowed value is 0.

`GC_MINS=<number of minutes, default 5>` - should normally be set the same as `LOCAL_CACHE_CLEAR_MINS`. This setting determines how often a “garbage collection” call is run which returns unused memory to the operating system.

Configuring External Memcached

Timesheet provides an “internal” copy of the memory cache server “memcached”. However you can run an “external” memcache either on the same host, or a different host, or multiple hosts. An external memcache server is not started and stopped by the Timesheet application service and is under your control. You can either specify external memcache servers during the initial setup (`jtinstall` script with the `--externalmemcache <SERVER LIST>` option) or by using `setkey` after the install:

```
setkey MEMCACHED_SERVERS=<SERVER LIST>
setkey EXTERNAL_MEMCACHED=Yes
```

The `<Server List>` is a comma-separated list of memcache servers in `host:port` format. Note that if you use multiple servers, they are used concurrently and are not treated in a “fail-over” manner. If just one server in the list becomes unavailable, it can make the application perform slowly for everybody.

As of version 8.7, Timesheet will automatically reconnect to an external memcache server that restarted or went offline temporarily. In version 8.0 the Timesheet service must be restarted whenever the external memcache server restarts.

`setkey MEMCACHED_SERVERS=host:port` Set this to configure an external memcached system using host:port notation, or a comma separated list of servers.

`setkey EXTERNAL_MEMCACHED=yes` should be run on any machine that won't be launching a memcached instance. This setting merely disables the launching of a memcache instance; `MEMCACHED_SERVERS` should also be set.

Please note that the meaning of the `EXTERNAL_MEMCACHED` setting changed between version 8.7m3 and 8.7m3P1 (and 8.7m4 and 8.8). The patch upgrade script should automatically migrate your old setting, if any.

The memcache can run under the same host, or a separate host. If using an external memcached you should ensure that your network is configured such that it isn't available outside your firewall to the general Internet, because memcached doesn't have a built-in authentication or access control mechanism.

The "internal" memcached instance only listens on the 127.0.0.1 (loopback) network interface for security reasons. That means it can only be accessed from the same local machine. If you need to host memcache on a separate machine (other than your Timesheet app server) or you have more than one app server host, then you must run an external memcached.

Important Security Note for External Memcached

If running an external memcached, for security reasons you should make sure the memcached host/port is firewalled off from everything except the Timesheet app servers. In addition you should make sure your memcached instance only listens on an internal network interface, if you have multiple interfaces. Use the `-l <IP Address>` option (with a lowercase L) to make memcache only listen on the network interface specified by that IP address. This interface must be reachable by the Timesheet app server(s) but should be closed off from the public internet.

Content Compression

When running under Lighttpd (on either Linux or Windows) then GZIP compression should be enabled for better performance. Use this command to check the current GZIP status from the Timesheet command line console.

```
getkey GZIP_ENCODING
```

Use this command to turn it on:

```
setkey GZIP_ENCODING=YES
```

In the unlikely event that it causes problems with certain older browsers, you can also turn it off:

```
setkey GZIP_ENCODING=NO
```

When running under the Microsoft IIS web server, the `GZIP_ENCODING` setting has no effect. Instead you can turn the Dynamic Content Compression feature on or off from within IIS Manager.

Database Tuning

Journyx makes no particular recommendations for external database tuning. Your local DBAs (database administrators) should be consulted for operational details. Timesheet is designed to be as independent as possible of database particulars. Running the Timesheet application on a separate machine from the Database Server can enhance performance. However, except for certain specialized operations, performance is generally limited by available CPU power on the application server more than any other factor such as database performance. So putting the application server on your fastest available hardware is generally the best bet for overall performance.

Custom Error Handlers

The Timesheet application has several static HTML files that are used as content for HTTP response error codes such as “404 Not Found” and “500 Internal Server Error”. These files can be customized to match the desired look of your site or to match your local policies. By default these only apply to error codes arising out of the /jtcgi script directory, not static files in /jtime or other resources on your webserver. To apply either the default Journyx error response content or your customized version to such requests, see the instructions appropriate to your web server below.

There are five customizable responses for /jtcgi resources:

- 401 Unauthorized. The user wasn’t authenticated (logged in), but might be able to access the resource after authenticating.
- 403 Forbidden. The user may be authenticated but is not authorized to access this resource.
- 404 Not Found. The requested resource does not exist or is inaccessible.
- 500 Internal Server Error. The service experienced an unexpected internal error.
- 503 Service Unavailable. The service is temporarily offline but is expected to return later. For instance, the service process is stopped or the database administrator is performing maintenance.

The content of these responses can be found in the htdocs directory (see the Directory Layout section of this document for the location of htdocs). The default response is contained in a file named defaultCODE.html. For example, default404.html for the 404 response. Do not modify the defaultCODE.html files. If you wish to customize these files, copy them to CODE.html (for example, 404.html) in that same directory and then modify them. Journyx product updates may overwrite default404.html and related files but will never touch 404.html.

In addition, you may wish to make your web server return these customized responses for similar errors that occur outside of the /jtcgi script directory context.

Apache

For Apache, simply add appropriate ErrorDocument statements to the virtual directory configuration section as shown in the “External FastCGI web server (Apache)” section of this document above. If you have customized the content, link to your version instead of default404.html etc. You can do this for 401, 403, 404, and 500 errors. Be sure to restart your web server after doing this.

Microsoft IIS

For Microsoft IIS, select the website in IIS Manager and then select Error Pages. Find the response codes you wish to change and right-click then select Edit.

Choose “Execute a URL on this site” and put in the URL like:

```
/jtime/default404.html
```

Substitute the correct error code for 404. If you have customized the /jtcgi error code responses as above, then instead of linking to default404.html, link to your customized version such as /jtime/404.html

Lighttpd

In Timesheet 8.8 and higher, the internal Lighttpd web server is already configured to return the default404.html error response. If you’ve customized it, or you’re on Timesheet 8.7 or earlier, you may want to configure this line. Open the lighttpd.conf file in an editor. You can find this here:

```
$WTHOME/pi/www/conf/lighttpd.conf (Unix / Linux)
```

```
%WTHOME%\LightTPD\conf\lighttpd-inc.conf (Windows)
```

Look for any uncommented lines that have “error-handler-404” in them and delete them.

Now add a line that says:

```
server.error-handler-404 = "/jtime/default404.html"
```

Replace default404.html with 404.html if you have customized 404.html. Save changes to the file and restart the Timesheet service.

The 404 response is the only one that can be customized under Lighttpd (for resources other than /jtcgi). Custom 401, 403, 500 and 503 responses (outside of /jtcgi) are not available on Lighttpd.

Starting, Stopping and Restarting the Service

Linux Service Control

Starting and stopping the service as well as backup and restore are performed via the command line interface and must be performed when logged in as the install user. When attempting to do any of these operations you must first set the proper environment variables by sourcing the setup file.

```

. ./setup                # Sets environment variables
backupdb                # Backs up the installation. (see below for details)
restoredb               # Restores a backup made with backupdb
wstop                   # Stops the application
wstart                  # Starts the application

```

Note: wstop and wstart only stop the database if the internal PostgreSQL database is in use. If an external Oracle, SQL Server or PostgreSQL database is being used, they must be manipulated using the native DB tools and interfaces.

Windows Service Control

Starting and stopping the “Journyx Timesheet” service on Windows can be done through the normal Services control panel (under Administrative Tools), or otherwise in the Computer Management application. Or you can do it at the command line:

```
net stop journyxtimesheet
net start journyxtimesheet
```

Or from the %WTHOME%\jw\bin directory, these shortcuts are provided:

```

wstop    (stops the Timesheet service)
wstart   (starts the Timesheet service)
wrefresh (restarts both Timesheet and the IIS web server)

```

The wstop and wstart commands on Windows take optional parameters of extra service names to stop/start respectively, such as w3sc for IIS.

Other Command Line Tasks

Aside from database backups and log file rotations, restarting (AKA “Bouncing”) the server is the only other CLI administrative task that might be performed on a regular basis. The Timesheet server daemons cache database results aggressively both in the daemons and through memcached in order to optimize performance. While the cache is flushed on write, some database configurations may allow the daemons to cache large amounts of data over time. If the daemons acquire too large of a virtual memory footprint there can be danger of termination of the daemons due to ulimits or decreased machine performance due to memory thrashing.

In the event the System Administrator wants to schedule a daily or weekly restart they should schedule cron to restart the site. This cron entry could be used to restart the site at 5:05AM and send mail to the system admin. Note the ‘.’ used to source the setup file.

```
5 5 * * * . <install_dir>/pi/bin/setup;wstop;wstart | mail -s nightlyRestart
sysadmin@your_site.com
```

Disaster Recovery

This section of the guide is designed to help you formulate and put into action a plan for backup and restore, also known as Disaster Recovery. The idea is that before a disaster strikes, you will already have a plan in place to enable you to quickly recover to a working state and get the application up and running again.

Purpose

This document describes the Journyx Timesheet database backup and restore utilities including the “Database Archive” feature. These utilities allow you to save your Journyx Timesheet data to a file on disk and restore it at a later time in case of system failure. You should implement a disaster recovery plan and do backups on a regular basis. These backup files should be stored in a safe place separate from your Timesheet server, ideally in a different geographic location. This document explains the different strategies and options for performing backups. All of the command-line options for these utilities are explained in detail along with general documentation and answers to frequently asked questions about backup and restore.

Backup and Restore Summary

To create a Timesheet backup, open a command prompt and run the **backupdb** command and give a filename for the backup file:

```
$ backupdb -v My_Timesheet_Data_January2011
```

A file with the name **My_Timesheet_Data_January2011.jx** will be created in the current directory. To restore that file at a later time run the **restoredb** command and give the filename for the backup file:

```
$ restoredb My_Timesheet_Data_January2011.jx
```

Please be warned that running **restoredb** will **permanently erase** all data in the site and replace it with the data contained in the **.jx** backup file.

The **-v** option the **backupdb** command enables printing of “verbose” status messages. That gives you a better idea of how long the program will take to finish.

You can also create an “Archive” backup file. An archive takes all the time, expense and mileage sheets that fall within a date range that you provide and writes them to a **.jx** backup file. You then have the option to purge (delete) those archived sheets from your live Timesheet site. The archive file can be saved away in a safe place or restored to an alternate server for reporting on archived data. To create an archive simply give the **-A** (capital A for Archive) option to the **backupdb** command. The program will prompt you for the date range and other options.

```
$ backupdb -A My_2004_Timesheet_Archive.jx
```

The complete set of **backupdb** and **restoredb** options are described in detail later in this document.

Disaster Recovery Planning

Any system administrator responsible for a Journyx Timesheet site should become familiar with the contents of this document and implement an effective data recovery plan. Journyx, Inc. is **not responsible** for the loss of any data for any reason from any system under the care and custody of the customer. Journyx strongly recommends that you backup your Timesheet system on a **nightly basis** and maintain backup media and old backup files going back a minimum of one week.

Backups of the Timesheet system should occur independently on at least three levels:

- Journyx backupdb format
- “Native” RDBMS / SQL backup

- Operating system level backup

Your recovery strategy should encompass all three levels of backups.

The Journyx backupdb command creates a cross-platform .jx backup file that is perhaps the simplest way to perform a backup. However for very large amounts of data, this can also be the slowest way to recover as the restoredb command can take a long time to run. The Journyx backupdb format includes professional services scripts and some other objects not directly stored in the database. We recommend you at least schedule a nightly backupdb command.

Journyx Timesheet stores all of your Time and Project data in a RDBMS (relational database management system), also known as a SQL database, for example Microsoft SQL Server, Oracle, or PostgreSQL. In addition to the backupdb .jx backup files, it is highly recommended that your I.T. staff perform regular “native” backups of the RDBMS hosting your Timesheet data whether you are using the "Internal Database" or an external RDBMS that you have provided, such as Oracle. Complete instructions for performing “native” backups are outside of the scope of this document but some guidelines can be found below.

In addition you should make regular operating system-level backups of the Timesheet application server (and the database server, if separate) to ensure the fastest recovery in the event of a total system loss. Remember: in a total loss situation having a copy of the database itself is only one part of the overall recovery strategy. Besides the database itself you'll need a working computer, operating system, network, and an installation of the Timesheet server application. Please consider all of those factors when formulating your recovery plan.

Backup Techniques

There are two common methods for making a backup of Journyx Timesheet data. One method is to make a direct backup of the RDBMS using the native recovery tools provided by your RDBMS software. The other method is to use the **backupdb** and **restoredb** commands provided by Journyx. The “pros and cons” of each backup solution are described in this table with some specific recommendations for a backup strategy following that. In general you should plan on using both strategies to obtain the best level of protection.

A third much less common backup option is to use a program such as Norton Ghost or other operating-system level tools to create a backup image of an entire hard disk partition. If you use this technique, it is highly recommended that you make additional backups with the Journyx backupdb utility and/or the RDBMS native recovery tools.

Timesheet Backup / Restore
(Backupdb and restoredb)

RDBMS Native Backup / Restore
(also called 'Recovery Tools')

Cross Platform File Format

The backup file can be restored on any operating system that Timesheet supports and with any type of database system. This is also useful in case Journyx Support needs to look at a copy of your database.

Non-portable Format

The backup file may only be restored on the same kind of database system – including the same RDBMS version in most cases. You cannot, for example, create a native backup on Oracle and then restore it to a PostgreSQL server.

Automatic Version Migration

Backup files from old versions of Timesheet can be “migrated” into a newer version with some limitations.

No Migration

A “native” RDBMS backup file from an old version of Timesheet may not be restored to the newer version. It can only be used by a Timesheet installation on the same version level as when the original backup was created.

Date Range Archive Supported

backupdb allows you to select a date range for the backup and to optionally delete these archived records.

Uses Compression

The Timesheet backup format automatically compresses the data with the popular and reliable ZIP format. The `.jx` files can be opened by most Zip compatible programs and the plain text backup file inside is easily extracted and read by humans.

Site must be offline to backup

While a backup is being made with the **backupdb** tool, the Timesheet site is offline and unavailable to your users.

Relatively Slow

Timesheet version 8 improved **backupdb** and **restoredb** speed significantly but a backup of a very large site can still take as much as 30 minutes or more.

The **restoredb** program is very slow compared to most RDBMS native restore programs. The speed is improved in version 8 but still relatively slow. A very large site can take hours or even several days to restore.

The best way to make the **restoredb** faster is to have fewer records in the first place; in particular, try to avoid activating the “historical custom fields for time records” feature unless absolutely necessary.

It is also a good idea to use the “Archive” feature on a regular basis (see below) to trim out unneeded old data. This will make your ongoing backups and any restore operations much faster.

As you can see from the table above, the two major disadvantages of the Journyx Timesheet backup format are that it is slow compared to native RDBMS recovery utilities and that the site must be offline to make a backup. (The site must be offline for a restore as well, even if you are using the RDBMS Recovery Tools.) The slower restore time compared to a native RDBMS restore may mean that your site is unavailable for longer periods of time while restoring.

Recovery Strategy

If your I.T. staff is capable of understanding and effectively using the native RDBMS recovery tools we recommend that you use these on at least a nightly basis in addition to making weekly backups in the Timesheet native **backupdb** format. The backup process should be completely automated if possible.

No Archive Feature

No easy or automatic way to archive away records that fall under a certain date range.

No Compression by default

The native backup format of RDBMS's are typically not compressed by default though often the option will be available. Compressed backups can take up as little as 10% of the original file size.

Online (“Hot”) Backups

Most RDBMS's allow you to safely perform a backup while the site is in active use. Sometimes this is called a ‘hot’ backup. This increases site availability.

Very Fast

RDBMS native backup and restore operations are typically very quick. Usually they are limited only by disk input / output capabilities.

If you prefer not to use the RDBMS recovery tools we do suggest that you make a Timesheet native backup at least once a day. Again this will usually be at night or during non-peak hours. Even if you plan to use the RDBMS recovery tools exclusively we recommend that you make Timesheet native backups at least weekly. You may choose a minimally disruptive time to perform the backup such as a weekend night. Automated backups are ideal for that. (See the "Automating Backups" section below.)

One important point to remember when planning your recovery strategy is to never make the current backup overwrite a previous one, especially if that previous backup is your only recent backup. Always backup to a **new** file. Ideally the filename should contain the current date and time as a convenient way of keeping the files distinct.

You can store the backup files on any media that is convenient for your staff whether DAT or other tape formats, recordable optical formats like DVD-R, external hard drives or network storage that is backed up by a separate process. The Timesheet backup utility cannot normally write directly to a tape or DVD-R. You must first create the backup file on disk, then you can copy that `.jx` file to tape or other backup media. Timesheet backups can be written directly to network shares provided that the appropriate access permissions are in place.

If you reuse your backup media, it is recommended that you keep 7 days worth of backups as a bare minimum. It is also highly recommended that you make a full backup in the Timesheet format at least monthly and store that backup in a safe place off-site (away from your place of business) in case of a fire or other disaster at your office. This is good advice for all of your business data, not just the Timesheet database.

The rest of this document describes the Journyx Timesheet native `backupdb` and `restoredb` utilities. A section at the end of the document does have some additional technical notes about performing native RDBMS backups.

Backup Procedures

Journyx Timesheet provides the **backupdb** command-line utility to perform backups. The files that `backupdb` creates have the `.jx` extension. A `.jx` file is a bundle of files compressed in the common "zip" archive format. These files contain your Timesheet database tables as well as other objects, such as professional services scripts and your site `config` file.

Running backupdb

To run `backupdb` on Windows, launch a command window by clicking:

```
Start -> Programs -> Journyx Timesheet -> Timesheet Command Line Prompt
```

Then you may simply type `backupdb` plus any option you need (see below) as well as the name of your new backup file.

On Unix, before you run `backupdb` or any other Timesheet command line utility, make sure you have sourced the Timesheet setup file first. To perform a backup on Unix, you must be logged in as the same account that was used to install Journyx Timesheet.

For example, if Timesheet runs under the `journyx` user, and is installed in the `/home/journyx/jtime` directory, log in as `journyx` and do this:

```
~:$ cd /home/journyx/jtime
```

That directory is also know as `$WTHOME` since it is your installation directory.

```
~/jtime:$ source pi/bin/setup
~/jtime:$ backupdb -v MyBackupFile.jx
```

The `.jx` backup file is always written to the current directory, unless you specify the full path to the output file, such as:

```
~/jtime:$ backupdb -v
/netshares/BackupVolume/2010/September/Timesheet_20100905_backup.jx
```

Please note: when you run `backupdb`, it will automatically take your Timesheet site offline. If any of your users attempt to access the site while you are making a backup, they will get a "Site is Offline for Maintenance" message until the backup completes. Please note that your Timesheet service must be running in order to make a backup if you are using the "Internal Database". Use `wstart` to start the service on Unix or use the Services control panel under Windows.

If the backup fails due to lack of disk space or for any other reason it will prompt you at the console whether or not to remove the database busy flag. Saying "yes" will put the Timesheet site back online. If you are running the `backupdb` command through an automated process such as a timed "at job" or "cron job" this may render your site offline if your script is unable to answer the prompt about the database busy flag. See the section below about automated backups.

backupdb options

These options may be given to the `backupdb` command. In addition to the options listed here, you must provide a filename for the output backup file. If you do not give the filename a `.jx` extension, it will be added automatically. For instance, if you type:

```
$ backupdb myfile
```

A backup file named `myfile.jx` will be created.

The "Archive" options for date range based backups are listed separately in the next section.

```
-? or --help    Show the usage screen
```

This option displays a short screen describing the rest of the options.

```
-v              Show verbose output
```

If you do **not** give this option, the `backupdb` command is silent. It will not print any output or progress indication. If you give the `-v` option, `backupdb` will keep you informed about its progress. In most cases we recommend that you use the `-v` option as it will give you a rough idea of how fast the backup is proceeding and how long it might take to finish.

```
-l              List what would be backed up
```

When you give this option, `backupdb` does **not** actually perform the backup. It simply lists the objects that would have been backed up as well as the source locations for those objects such as the professional services scripts.

```
-p (lowercase) Backup only the professional services scripts
```

When you give this option, the backup will **only** have the Professional Services files. It will not include the actual Time data or anything else from the database. This option is useful if you wish to move your professional services scripts to another system without moving the actual Time data. Please note that this only backs up Professional Service files in the "cgi" directory. To ensure a complete backup of all Professional Services deliverables please do a filesystem backup of the entire Timesheet install directory, such as `C:\Program Files\Journyx`.

`-O` Allow backupdb to overwrite existing files

Timesheet `backupdb` will not overwrite an existing backup file unless you give this flag (capital O.) For safety it is never wise to overwrite your only good backup while in the process of making a new backup, so we do not recommend the use of this option.

`-x` Ignore database busy flag

Normally the `backupdb` command will not perform a backup if the "database busy" setting is active on your Timesheet site. "Database busy" is turned on by certain maintenance utilities such as other instances of `backupdb` or `restoredb` or the `guilockout` command line tool. When you specify `-x` it will ignore this setting and perform the backup anyway. This option does **not** turn off the prompt about the database busy flag that comes up if the `backupdb` program encounters an error. There is currently no way to disable that prompt.

`-b` Use binary file format

Normally the `backupdb` command creates a human-readable "plain text format" backup file inside the compressed zip file. The `-b` (binary) option uses a more efficient format that is not human-readable. This can make the `restoredb` operation run up to 20% faster but otherwise has no effect.

Database Archive (Date Range based backup)

Timesheet version 5.5.2 introduced the capability of archiving only a specific range of dates with the `backupdb` program. This creates a regular `.jx` backup file except that the time, expense and mileage sheets in the file will be constrained to the given date range. There are a number of reasons you might want to do this.

The most important reason to use the archive feature is to get rid of old data from prior years to improve system performance. If you no longer need to report on this data getting rid of it with Archive is a good way to trim the overall size of your database and greatly improve performance. You can still report on the data by putting it on an "archive" server away from your main production Timesheet server.

Another reason to perform an Archive is in order to send a recent copy of your database to Journyx Support for troubleshooting without including all your old data from prior years. If you plan to do this, please see the note below about the `-N` option to disable the inclusion of the entire database in the archive file.

Basic Archive Usage

To make a date range archive of your Timesheet data, just run the `backupdb` command with the `-A` (capital A for Archive) option. The program will prompt you for the range of dates and whether or not to delete the archived records. In most cases, to create a date range archive you do not need to provide any additional options other than the `-A`. The additional options listed below are for certain specialized options, or to avoid being prompted for the archive dates.

The program will create a regular `.jx` Timesheet backup file. The backup file will contain all the time, expense and mileage sheets that fall within the given date range. No other data will be constrained by the date range; your archive file will always have a complete version of the Project tree and the Field Values (such as Pay Types and Bill Types.)

The archive program will prompt you near the end whether you want to delete the archived data from your live site. If you say **yes** to this question the archived sheets that fall within the date range will be *permanently deleted* from your live production Timesheet site. The only copy of that data will be in the newly created archive file.

To review or report on this archived data you will need to restore the archive to a non-production "test" or "archived" Timesheet site. This requires a completely separate Timesheet install. If your Timesheet server is running under Microsoft Windows you will need a separate machine for your "test" install as multiple Timesheet

installs cannot coexist on the same Windows machine. If running under Linux or Unix you can create a separate Timesheet install on the same machine as long as you do it as a different system user. If you are using an external database you must also use a separate database user account.

To access the archived data simply restore the archive file to your “test” archive reporting site with the **restoredb** command. You may then log into the test site and run reports.

Please note that in most cases your separate “test” or archive reporting server will need its own license key. Please contact the Journyx Sales department if you need a separate key for archive reporting; often these keys can be obtained at a significant discount.

Archive Limitations and Considerations

The archive tool has some limitations and other issues that are important to consider when making decisions about archiving old data.

It is not possible to move data from an archive reporting site back onto your main production site. Make sure you do not enter any *new* time data on your archive reporting site. You may wish to configure the login screen of the test / reporting site to have a message warning users that it is an archive site and not to enter any new data. However you can merge new archives into your archive/reporting site using the `restoredb --merge` option described below. In general, data can only flow one way – from your main production server to the archive server.

The primary cause of large database sizes and slow archive operations is using an excessive number of “historical custom fields.” (Also called historical Extra Fields in version 6.0 and earlier.) Every custom field that has the “historical for time records” option set generates a large amount of extra data that must be dealt with by the backup, restore and archive code. These options also affect the overall site performance during peak usage hours. Disable the “historical” feature on any extra fields unless absolutely necessary. As of version 5.6 Timesheet built-in reports can report on the “current” value of extra fields as opposed to the “historical” value.

The archive operation itself can be slow in some circumstances. If you have a large volume of data to purge (delete) after archiving, this can take a very long time in your database server. Experiments have suggested that to delete 1,000,000 records takes approximately 1 hour of database time. A site which has 1,000,000 time records to archive plus 15 “historical extra fields” associated to time records may take up to 16-20 hours just to delete the records. That is in addition to approximately 1 hour of time to perform the archive / scanning process. Please be patient and do not cancel the archive process prematurely. If you are concerned whether the process has “hung up” check your database server for activity. RDBMS’s such as Oracle have management tools which will show you progress in very long running requests such as a massive delete transaction.

See also the notes below about the `-Z` option and transaction / rollback space in your database.

Understanding Archive Date Ranges

It is important to understand how **backupdb** selects which Time, Expense, and Mileage sheets will be included in the archive. For example if you create an archive of all Pre-2002 records do not be surprised if some records from early January 2002 are included in the archive. That is because a user’s 1-week time sheet may run from Sunday, December 29th 2002 to Saturday, January 4th 2003. If you have selected 20021231 (December 31st 2002) as your archive cutoff date some “overlap” sheets may contain January 2003 days.

Normally these overlap sheets will be included in the archive but they will **not** be deleted even if you choose to purge out the archived sheets. In other words, by default Time/Expense/Mileage sheets will only be purged if **all** of the dates of the sheet fall within the selected archive range. The `-Y` option overrides that feature and makes it purge all archived sheets including overlapping ones.

This lets you create archives based on any time period that you choose without worrying about splitting up Time / Expense / Mileage sheets that have already been approved by your staff.

By default overlap sheets will be included in your archive. You can use the `-X` (capital X) option to **eXclude** overlap sheets. This means that a sheet will not be included in the archive unless **all** of the dates in the sheet fall within the given archive range.

Archive Preparation Checklist

When you are considering doing an archive or purge operation please run through the following checklist to make sure that you are fully prepared.

- Have you determined the exact date range that you wish to archive? You can choose either a specific date range (such as January 2008 to December 2008) or you can choose an open ended range such as “all sheets before January 2009.”
- Have you determined what you will do with the archived data? Will you simply keep the file for backup purposes or do you plan to set up the data on a separate legacy reporting site?
- Do you need to purge (delete) the archived sheets from your live production site? Using the purge option to delete archived sheets will greatly extend the total amount of time to complete the archive on very large sites. The speed of the purge operation depends entirely on the speed of your database system in deleting a large amount of data under a transaction. Usually this in turn depends on the speed of your disk(s) and the amount of data involved. It is highly recommended for the best performance that your RDBMS uses separate physical disks or arrays for data, indexes and transaction rollback space. This applies to general Timesheet performance as well as the Archive operation.
- Do you have the latest maintenance release for your version of the application? Maintenance releases often contain performance improvements or other fixes for backup/restore.
- Do you have enough disk space available on the application server? You will need *more* disk space for the backup operation than the final backup file will require by itself. A good rule of thumb is to estimate 175% of the database size will be needed as temporary disk space. That means if you are backing up a 2 gigabyte database then you should have at least 3.5 gigabytes of free disk space on the application server – or more just to be safe. The space must be available on the volume that holds the backup file.
- Do you have enough time available? Archiving and purging several years worth of data from a large site with thousands of active users can potentially take up to 48 hours or more in extreme cases. For these large sites consider starting the archive on a Friday morning and letting it run over the weekend.
- Can you do a test run on a separate test site using a copy of the database? Doing this will give you a better idea of how long it will take in production. The time will be proportional to the overall size of your database and the size of the date range being archived and purged. Doing a test run will also help you be aware of any problems before they affect your production site such as running out of database transaction space.
- If you have done a test run have you restored the archive to a separate test site and verified that the correct data is archived and reportable?
- Have you notified your users that the Timesheet application will be offline for a certain period of time? You should allow them a final chance to get into the system before taking it offline.
- Are you using the interactive mode (`-A`) or are you setting the archive start and end dates manually with the `-S` and `-E` options? If the latter then you must use the `-P` option to purge (delete) records. You will only be prompted to delete records if you use the `-A` interactive Archive mode.
- Have you considered using the `-S`, `-E` and `-P` options instead of `-A` interactive Archive mode? Using these options let you avoid the prompt about purging (deleting) records. This prompt only comes up after the main part of the archive operation has finished; this may happen late at night when no one is around to answer the prompt. If you want to delete archived records you should use the `-P` option to avoid the prompt. This will reduce your total amount of downtime by eliminating time wasted waiting on the prompt.
- Have you checked with your database administrator (DBA)? Have you informed him or her that the purge operation will require heavy database resources and may tie up the database for a period of hours or days in some cases?

- Does your RDBMS system (such as Oracle) have a large amount of free transaction space? (Also known as RBS, rollback segments, or just rollback.) The space you will need is related to the size of the purged records compared to the total size of the database. For instance, if your Timesheet site has 5 years of data and you are archiving 2 years, you will need at least two-fifths (40%) of the data-tablespace size available as rollback space.
- Have you made an RDBMS native backup “just in case”? For instance, have you used the Oracle EXP utility to make a copy of the database?
- Have you double-checked your archive plans with members of your organization who are involved with Timesheet administration, configuration and support?
- Have you studied this document in detail? Have you read about all the different command line options?

Archive Options

This section details all of the **backupdb** command options related to the Archive feature. Note that all of the Archive related options for **backupdb** are capital letters. These options are only available in Timesheet version 5.5.2 and higher.

`-A` prompt for specific dates to Archive

This is the main option to invoke Archive mode. If you give this option you do not need to give the options below. With the `-A` option the backupdb utility will prompt you for the start and end dates of the archive. It will also prompt you at the end whether or not you want to purge (delete) the archived sheets from your live Timesheet site. However if you use the `-S` and `-E` options (see below) to set the date range of the archive, “Archive mode” is implied and you do not need to use the `-A` option itself. Please note that in that case you will not be prompted to delete records and must use the `-P` option to force deleting of archived records.

When prompted for the archive start and end date you must enter the date in `YYYYMMDD` form (4 digit year first, then 2 digit month then 2 digit day. Use a 0 if necessary, such as 01 for January.) Use the `-` character (dash) to indicate “no limit.” The program will prompt you for a new date if the date you gave is not a valid `YYYYMMDD` date.

If you give `-` (no limit) for both the start and end dates of the archive, then a regular backup is performed.

`-S YYYYMMDD` specify a start date for Archive (or `-` for no limit.)

Specify a start date for the archive in `YYYYMMDD` format. You may also use the `-` (dash) character to mean “no limit.” If you specify `-S` but do not specify `-E`, the program will assume you mean the end of the archive should not be limited.

For instance, this command:

```
$ backupdb -S 20090101 myarchive.jx
```

will produce an archive starting on January 1st, 2009 and containing all records up to the current date, since `-E` is not specified.

Please note that if you use either the `-S` or `-E` option then Archive mode is assumed and you do not need to give the `-A` option. More importantly if you use either `-S` or `-E` then you should be aware that you will **not** be prompted to delete any records. If you use `-S` or `-E` then you **must** use `-P` if you want the archived records to be purged (deleted) from the site.

`-E YYYYMMDD` specify an end date for Archive. (or `-` for no limit.)

This specifies the end date of the archive, or - (dash) for no limit. Similar to `-S`, if you specify `-E` but do not specify `-S`, then the program assumes the archive has no limit for the start. For instance, this command:

```
$ backupdb -E 20090101 myarchive.jx
```

will produce an archive including all sheets from the past up to January 1st, 2009.

```
-P (capital)  purge (delete!) archived Sheets from the live site without
prompting. (Use caution with this option!)
```

Use this option to force the Purge (deletion) of all archived sheets without prompting. Normally you will be prompted whether or not to purge the archived sheets. This option skips the prompt and always purges.

```
-Y           Purge archived sheets with some days outside the archive range
```

As noted above, by default, a sheet is only purged if **all** of the dates in the sheet fall within the archive range. This option forces it to purge even these overlap sheets, assuming that purge is enabled. This option has no effect when combined with `-X` (see below.)

```
-X (capital)  eXclude sheets whose dates do not fall entirely within the given
date range. The default is to include all sheets that have at least 1 day within
the archive date range.
```

This option excludes 'overlap' sheets -- those sheets where at least one of the days falls outside of the specified archive date range. When `-X` is enabled, these overlap sheets are **not** included in the archive, and they are **not** purged.

The default is "inclusive mode" -- a sheet is included in the archive if **any** of its dates are within the archive range, even if some of the dates are outside the range.

```
-N           Do not include 'full' backup in Archive (saves space.)
```

Normally backupdb (even in Archive mode) produces a file that contains a **full** copy of your entire Timesheet database. If you have used the Archive option and try to restore an archive `.jx` file, the **restoredb** program will ask you whether you want to restore just the archive date range or the full database. This is done for your convenience and the safety of your data.

However, sometimes you may wish to produce an archive that does **not** contain the entire database, and you can use the `-N` option to do this. One reason to do that is to make the resulting `.jx` backup file much smaller, since it only needs to contain the data for the date range you specified. This is often useful when creating an archive to send to Journyx Support for troubleshooting. Often, Support will only need the last 2 months of data in order to diagnose a problem.

It is also possible to open the `.jx` backup file with a Zip utility such as Winzip to remove the 'full' data from an archive backup. Simply delete the `database.backup.trimmed` file within the `.jx` zip archive. **Do not touch** the `database.backup` file as this contains your primary archived time data. It is best to make an extra backup copy of the `.jx` backup file before you attempt to modify it.

Please note that when you give the `-N` option the `database.backup.trimmed` file inside the `.jx` archive is still created but it will be very small in size and will not contain your time data.

```
-Z           Do not use a database transaction when purging records.
```

Normally you will not need this option. This option is provided in case your RDBMS does not allow very large transactions. Do not use this option unless you have been told to do so by Journyx Support.

Purging archived records normally happens within a single database transaction. This is in case there is any kind of error during the delete process; in case of error, the database will be quickly restored to the state it was in just prior to the backup. In other words, either all of the archived records are safely purged or none of them are.

When you are purging a year's worth of Time and Expense data from your system it often requires a very large transaction. Occasionally a RDBMS such as Oracle can run out of transaction log space. That means you are not allowed to purge your archived records in a single transaction. Even without transaction space problems, a very large delete of millions of records can take 12 to 24 hours in some extreme cases so please be patient. (See notes above about large deletes.)

The best way to handle a transaction space situation is to have your DBA (database administrator) increase the transaction log space in the Timesheet database and then re-run the backup/archive.

This temporary disk space for transaction information is called "rollback segments" or RBS in Oracle. In Microsoft SQL Server, it is just called the transaction log file. Normally, SQL Server adjusts the space for this file automatically. Depending on its configuration, Oracle does **not** adjust RBS space automatically and your DBA must make manual adjustments. The most common Oracle error indicating too little RBS space is `ORA-01555 "snapshot too old"` error.

On Oracle you must have at least one free RBS that is large enough by itself to contain the entire transaction. It will not work to have lots of different small rollback segments. The size required obviously depends on your particular data. The best way to tell is to look at the (uncompressed) size of the `database.backup` file within the `.jx` archive that you just created. In general, you must have a free RBS with at least 80% of the size of your uncompressed `database.backup` file to do a purge.

If it is not possible to increase your transaction log space you can specify the `-Z` option to do the purge in multiple transactions. This avoids the problem of needing one single large transaction logfile. The downside and danger to this option is that if something goes wrong during the purge, some of the archived records may have been deleted while others were not. This may leave your database in an inconsistent state and lead to unpredictable and unsupported errors.

In the unlikely event that this happens it is recommended that you do a full restore of Timesheet from your most recent backup. In most cases you can use the archive file that was just created prior to the purge error.

Restoring Timesheet Databases

The **restoredb** program allows you to restore the backup files that you created with the **backupdb** command. If you are on Unix, be sure to source the Timesheet `pi/bin/setup` file before running **restoredb** as described above in the section for **backupdb**.

In general, you do not need to provide any options to **restoredb** other than the name of the `.jx` backup file. The program will prompt you for any information that it needs. However, a number of options are provided to tweak the restore behavior and in some cases avoid the prompts. Keep in mind that **restoredb** will delete any data currently on the site and replace it with the data in the `.jx` backup file. However the `--merge` option described below changes this, allowing you to restore to a site without erasing the existing data. This should only be used in conjunction with the "Archive" features of **backupdb**.

restoredb options

`-P` Do not restore professional services scripts

This option should be self-explanatory.

`-A` Clean restore of special Database Archive format (this will overwrite any current data)

This option skips the prompt when you attempt to restore an Archive (date range limited) backup. The prompt asks you whether you want to restore the specific date range or the entire backup if available. If you specify this `-A` option, it will skip the prompt and just restore the archived (date-range) data set. This option is only available on Timesheet 5.5.2 and higher.

`-p` Restore professional services scripts without restoring database

Just restore a set of Professional Services scripts without touching the actual database.

`-F` Do not force foreign key constraints after migration

Do not use this option unless instructed to do so by Journyx Support.

If Timesheet is unable to activate a foreign key (FK) constraint due to corrupted records it will automatically delete and log the corrupted records. This option overrides that behavior; unless the FK is mandatory it will be disabled in case of any corrupted records.

`-u` Create an undo file

This option creates a backup file of the current Timesheet database before restoring the file named on the command line. You will be prompted for the name of the 'undo file'. The undo file is just a regular Timesheet `.jx` backup.

`-k [license]` Specify license key

Allows you to specify a new license key for the site on the command line without being prompted.

If you do not specify this option, you will be prompted for a new license key. At that time you may simply press Enter if you wish to keep the existing license key that is currently on your site. The existing key will not be displayed.

`-x` Ignore database busy flag

Restore the database even though the 'database busy' (system maintenance) flag is already set. Be careful when using this flag as some other administrator on a different account may actually be doing maintenance work at the same time.

`-q` No output or warnings

If this option is set **restoredb** will not print any status output to the console.

`-d` Turn debugging on

Normally, "extended tracebacks" are turned off automatically after a restore. This option turns on extended tracebacks (also called 'debugging' in the Logging Preferences screen) after a restore.

`-l` Turn debug logging on

Usually "diagnostic logging" (`debug.log`) is turned off automatically after a restore. This option turns on diagnostic logging for the restore itself, and leaves the option on after the restore.

You should not leave diagnostic logging on for extended periods on your production server unless instructed to do so by Journyx Support. The log file (`debug.log` in your Timesheet `tmp` directory) can quickly fill up all available disk space on a busy site. Leaving Diagnostic Logging turned on will also slow down overall site performance.

```
--merge           Merge the data in the backup file into the current site.
```

This option is currently not shown on the `restoredb help/usage` screen. It merges the data in the backup file into the current site without erasing the existing data. This can be used in conjunction with the "Archive" feature of `backupdb` to merge newly archived data into an existing archive/reporting site.

```
-? or --help      Show this usage screen
```

Shows a short help screen listing the `restoredb` options.

Other Backup / Restore Topics

Automating Backups

These instructions only apply to Microsoft Windows Server 2003 and higher. It is also possible to schedule automated backups on various Unix systems using the "crontab" command. Please consult your Unix System Administrator or documentation for details.

To set up a scheduled task in Windows Server 2003 or higher, right-click "My Computer" and select Manage then choose Scheduled Tasks.

Double-click "Add Scheduled Task." Click Next.

It will ask you to select the program to run. The Timesheet `backupdb` will usually not be on the default list. Therefore, click Browse. Select the `backupdb.bat` file. Assuming you installed Timesheet to the default location, that will be:

```
C:\Program Files\Journyx\jwt\bin\backupdb.bat
```

Next it will ask you to select a name for the task. Name it something like "Timesheet Nightly Backup."

Normally you will want to run the task every night, so select Daily and then click Next.

It will then ask you more questions about when to run the task. Choose a time for the task. We strongly recommend that you test the backup task at least once during normal business hours, and then after you confirm that it works set it to run at an off-peak time such as 4 a.m. After you have selected a run time click Next. After you have created the scheduled task, you can test it at any time by right-clicking on it and selecting 'Run.'

It then will ask you which Windows user should run the task. It is critical that you choose the same user account that you used to install the Journyx Timesheet software. If you use a different account here, it will not work. Be sure to give the appropriate account password for that account and then click Next.

Now you must be sure to set some options on the `backupdb.bat` command line, so be sure to check "Open advanced properties" on the final screen of the wizard.

In the Properties screen, change the Run: line as follows:

(Original)

```
"C:\Program Files\Journyx\jwt\bin\backupdb.bat"
```

(New)

```
"C:\Program Files\Journyx\jwt\bin\backupdb.bat" -Ox ..\..\NightlyBackup.jx
```

Note: you cannot use a path with spaces when editing the command line for a Scheduled Task; therefore we use the `..\..\` relative path as above. An example of a path with spaces is "C:\Program Files\Journyx"

It is important to note that the scheduled task as described above will cause it to overwrite the previous night's backup every time. This is not a good idea. If something goes wrong with the backup, you have just deleted your previous (good) backup. We recommend that you create a separate batch file which saves the old backup file before running backupdb.

For instance, you can create a new file in the %WTHOME%\jwt\bin directory called `autobackup.bat`. Be sure to create the file with a plain text editor.

Put commands like this in the `autobackup.bat` file:

```
c:
cd "c:\Program Files\Journyx"
copy NightlyBackup.jx PreviousNightlyBackup.jx
cd jwt\bin
backupdb.bat -Ox "C:\Program Files\Journyx\NightlyBackup.jx"
```

Then you can change the Windows Scheduled Task to run the `autobackup.bat` file instead of `backupdb.bat`. Be sure to test the Scheduled Task by right-clicking on it and selecting "Run."

Native Backup Formats

In this section we describe briefly how to perform "native" database backups for the PostgreSQL "Internal Database" included in Journyx Timesheet. Please see the "Backup Techniques" chart above to determine whether making native backups is the right strategy for your organization.

Journyx also supports external Oracle and Microsoft SQL Server databases, but the recovery tools for those systems are not described in detail here. Please consult with your DBA (database administrator) if you are using Oracle or Microsoft SQL Server and wish to create native backups.

Oracle

Native backups are normally launched through the "Enterprise Manager" program. Please consult with your Oracle DBA (database administrator) for details. Some online documentation for Oracle backup and recovery is available here:

http://www.orafaq.com/wiki/Oracle_database_Backup_and_Recovery_FAQ

Microsoft SQL Server 2005 and later

Microsoft provides a program called SQL Server Management Studio to manage a SQL Server database. This tool lets you perform native backup and recovery operations from directly within the tool. Please consult your SQL Server DBA (database administrator) or Microsoft documentation for detailed instructions. Here is a link to documentation for SQL Server 2008:

<http://technet.microsoft.com/en-us/library/ms187048.aspx>

PostgreSQL Native Backups

It is quite simple to make a "native" backup with PostgreSQL (PG), whether it is the version provided by Journyx or your own external PG site.

The `pg_dump` command creates the backup. Normally the data is dumped to the console (screen) so you will want to redirect it to a backup file. In this example, assume that your database is named `journyx` and you

wish to backup to a file named `my_dump.sql`. (Please note that when the Internal Timesheet PostgreSQL instance is used, the database is named "journyx", so you would use `journyx` in your command.) The **psql** command restores the backup file.

PostgreSQL on Linux

This command creates the backup from the 'journyx' database:

```
$ pg_dump -c journyx > my_dump.sql
```

The `-c` makes a 'clean' backup including commands to drop old data.

This command restores the backup to the 'journyx' database:

```
$ psql journyx < my_dump.sql
```

After that completes, simply restart your timesheet site (`wstop; wstart`) to access the newly restored data.

PostgreSQL on Windows

Running a native PostgreSQL backup on Windows is very similar to Linux but some additional options may need to be supplied, namely the `-P` (port) option since the copy included with Timesheet doesn't run on the default port.

First open a Timesheet Command Line Prompt, under Start -> Journyx Timesheet. Now change to the `%WTHOME%\pgsql\bin` directory:

```
> cd "%WTHOME%\pgsql\bin"
```

Now run a command like this:

```
> pg_dump -h 127.0.0.1 -U journyx -p 5532 -c journyx > my_dump.sql
```

`-h` sets the host name. `-U` specifies the username and `-p` specifies the port (use 5532.) The `-c` makes a 'clean' backup including commands to drop old data. 'journyx' is the default database name.

This command (run from the `%WTHOME%\pgsql\bin` directory) restores the backup to the 'journyx' database:

```
> psql -h 127.0.0.1 -U journyx -p 5532 journyx < my_dump.sql
```

After that completes, simply restart your Timesheet site (as described in the Windows Service Control section above) to access the newly restored data.

Change Log

1.0 Original document version.

2.0 Converted to Journyx Technical Documentation. Made several revisions and expanded material on database archive.

3.0 Updated for product version 8.0 and merged the Sysadmin Guide.

3.0.4 Added material for clustered application server setups.

3.0.5 Added material about configuring SSL on Lighttpd and creating self-signed SSL certificates for testing. Also added information about how to disable TRACE and TRACK verbs in Apache config.

3.0.6 Added information about customizing HTTP error response content.

3.0.7 Added material in the Security section about External Reporting and the internal PostgreSQL database account passwords.

3.0.8 Added material about new external Memcache settings new in version 8.7m3P1 and additional info about running multiple app servers (clustering) on Windows.

3.0.9 Updated the website links for external database setup instructions and other links.

3.1 Updated for Timesheet 8.8.

3.2 Updated for Timesheet 8.9.

3.3 Updated hardware requirements and support information.

3.4 Updated for Timesheet 8.9m1 and information about SQL Native Client drivers and installing on internationalized Windows.

3.5 Updated support website links. Added recommendations for Windows install user selection. Added some details on effects of config settings and memory utilization.

(End of Document)