



Installation Guide

@TASK 4

Copyright 2007 AtTask, Inc.
All Rights Reserved

TABLE OF CONTENTS

Chapter 1: System Prerequisites	3
About Installing @task	4
System Requirements	5
Database Setup	6
Database Setup for MySQL	6
Database Setup for Oracle	7
Database Setup For Microsoft SQL Server 2005	9
Chapter 2: Installing @task	11
Installing @task	12
Accessing @task	19
Upgrading from Earlier Versions of @task 4	20
Running the Java Install Packages	21
Instructions (Unix or Unix-like operating systems)	21
Instructions (for other platforms)	21
Chapter 3: Advanced Installation	22
Using Clusters	23
Basic Overview of Cluster Functionality	23
Cluster Support Checklist	23
JBoss Cluster Setup	24
JBoss and @task Clustering Setup	24
Client Configuration Changes Using The @task API With Clustering	25
FAQ	25
Running @task as a Service	26
Running @task at Startup Using Mac OS	27
Secure Sockets Layer Setup for JBoss	28
Setup	28
Secure Certificate	28
JBoss Configuration	28



Chapter 1

System Prerequisites

This chapter contains the following sections:

- [About Installing @task](#)
- [System Requirements](#)
- [Database Setup](#)

ABOUT INSTALLING @TASK

@task is a robust, exhaustive project management tool that can help you manage your projects efficiently and effortlessly. You can download @task's installation application from the @task Web site:

<http://www.attask.com>

@task consists of 2 components: the database and the @task application. You must install the database prior to installing the @task application. The @task application and Web server are installed by the @task installer. For generic platforms (Generic Unix and the Java-based installer), you must download an appropriate JDK 1.5.0_11 prior to installation.

This manual contains basic procedures for setting up a database. For detailed instructions, consult vendor documentation.

[See "Database Setup" on page 6.](#)

@task runs on JBoss and has its own built in web server. While integration with other Web servers such as Apache or IIS may be possible, @task does not support other Web server configurations.

If you plan to install @task behind a firewall, or you plan to share your @task server with other applications, you should know the ports that JBoss uses. You need to know these so that you can open the appropriate ports in your firewall, or so that you don't have port conflicts with other applications on your server. Table 1.1 lists these ports.

NOTE: While you can install @task on a shared server for very small installations, @task is designed to run on a dedicated server.

TABLE 1.1: JBOSS PORTS

PORT	NUMBER
RMI Port	1099
JNP Port	1098
RMI Object Port	4444
JNDI Port	8083
AJP Port	8009

SYSTEM REQUIREMENTS

Table 1.2 outlines the system requirements that you need to install @task version 4.0.

TABLE 1.2: SYSTEM REQUIREMENTS

REQUIREMENT	REQUIRED	SUGGESTED
Operating Systems	Windows XP Windows 2000 Windows 2003 Server Mac OS X 10.4 and later Linux UNIX HP/UX AIX Solaris Java-enabled Operating Systems	You should use a dedicated server for your @task application.
CPU	1.5 G	75 - 150 users: 2G 150+ users: dual 2G
Disk Space	15 GB free space	40+ GB free space
RAM	512MB	1 - 20 users: 768 MB 20 - 75 users: 1 GB 75 - 500 users: 2 GB 500+ users: Contact @task Customer Services.
Databases	Oracle version 9i and later MySQL version 5 and later Microsoft SQL Server 2000 and 2005	Use Oracle or MSSQL 2000/2005 for installations with more than 500 users.
Java Development Kit (JDK)	Version 1.5.0_11	

DATABASE SETUP

Before installing @task, you must install and configure your database. @task currently supports three standard databases natively: MySQL, Oracle, and Microsoft SQL Server 2000/2005. Installation instructions for each of these databases can be found on subsequent pages:

[See “Database Setup for MySQL” on page 6.](#)

[See “Database Setup for Oracle” on page 7.](#)

[See “Database Setup For Microsoft SQL Server 2005” on page 9.](#)

DATABASE SETUP FOR MYSQL

A file called `attaskSchema.sql` is included. Before @task can communicate with a database, the database needs to allow @task access to it and needs to contain all of the tables and fields that @task needs to store its data. The procedure you should follow is: create the database, import all of the setup data, and give access rights to the web application.

SETUP MYSQL

Once you have downloaded and installed MySQL you need to make sure the security settings grant @task access to the database. ‘Access denied’ errors are usually due to MySQL’s security structure. If you are not currently using MySQL for anything else, this is the recommended procedure:

1. Open MySQL with a command prompt:

```
mysql -uroot mysql
```

If you have a default unsecured installation, that command should work. You can also use the MySQL Command Line Client that is bundled with the MySQL installation.

2. Reset all of MySQL’s security.

```
delete from user;  
delete from db;
```

3. Create a user with full access to the database from “localhost” and `<webserver_hostname>`, where ‘`webserver_hostname`’ is your Web server host name. Give this user a user name of ‘root’ and a password of ‘rootpass’ or substitute your own password. This user is your ‘DBO user’. For example if you were creating a user with a name of root and a password of rootpass, and you installed MySQL on a computer with an IP address of 10.10.100.1 you would use the following command:

```
GRANT ALL on *.* to root@'240.14.28.1' IDENTIFIED BY 'rootpass' with grant option;
```

You can also use the computer name in the previous command.

If you plan to host the web server and application server on a separate machine from the database, then you also need to do the following command. For this command assume that the computer name for your Web server is `RexTheWebServer`. You can use the computer name or the IP address.

```
GRANT ALL on *.* to root@'RexTheWebServer' IDENTIFIED BY 'rootpass' with grant option;
```

CREATE THE DATABASE

The name of the database is encoded in the license key. The database name that you provide when you get your license key is that name of the database that you need to create. Assuming that the database name that you choose is ‘attask4’, to create the database, type the following in the MySQL command prompt:

1. Use the following command at the MySQL prompt:

```
CREATE DATABASE attask4;
```

CREATE THE @TASK USER

Now you need to create an @task user. When you install @task, this is the user name and password that you provide so that @task can connect to the database. If you use root and the root password, you do not need to create another user.

1. After deciding on a user name and password, ensure that the @task web.xml file, located at <TOMCAT_HOME>/attask/WEB-INF/, references the same user name and password that you will use in this next step.

Type the following line into the MySQL prompt, replacing 'attaskuser' with your chosen database user name and 'attaskpass' with your chosen database password:

```
GRANT ALL on attask4.* to attaskuser@'webserver.hostname' IDENTIFIED BY 'attask_user_pass';
```

2. Flush MySQL's privileges by typing the following:

```
FLUSH PRIVILEGES;
```

3. In a second command prompt window verify that you can log in. Replace *webserver_hostname* with the correct value:

```
mysql -uattaskuser -pattaskpass -hwebserver_hostname  
mysql -uattaskuser -pattaskpass
```

4. In the web.xml file that comes with @task, make sure you enter the dbUser and dbPass that you used to identify the @task user.
5. You must restart Tomcat after editing web.xml. Restart Tomcat when you are finished with this section.

BACKING UP YOUR MYSQL DATABASE

A backup of a MySQL database can be done using more than one method:

1. Dump the database information to a text file that can be imported to recreate the database. This is done with the mysqldump tool in a manner similar to that shown here:

```
mysqldump -udbUser -pdbPass -c --add-drop-table myAttaskDatabase > backup.sql
```

You can access help for using the mysqldump tool by typing mysqldump with no parameters. The mysqldump tool is in the 'bin' directory of the MySQL installation.

2. Copy the actual data directory where MySQL stores the information about the database. Look in <MYSQL_INSTALLATION>/data to find a folder for every database created in your MySQL instance. Simply copy the desired directory to a safe place. You may want to use a tar or zip file for the directory for convenience and storage.

DATABASE SETUP FOR ORACLE

Before @task can communicate with a database, the database needs to allow @task access to it and needs to contain all of the tables and fields that @task needs to store its data.

DATABASE SPACE REQUIREMENTS

While every instance of @task stores different information and has unique disk space requirements, there is a general estimate on needs. As your particular instance exceeds your space allocations, you may need to increase the size of your tablespace extent or datafile size, or create additional datafiles to store in the tablespace. It is assumed that there is sufficient expertise within your organization to accomplish these tasks.

@task uses two tablespaces, ATTASK_DATA, for storing the data tables, and ATTASK_IDX for storing the indexes. As a general rule, these should start with at least 80 MB each.

These sizes are estimates based on data that includes 100,000 tasks, 1000 projects, 500 users and 100,000 notes while still allowing for growth. Because some organizations will create more notes and others will add

more tasks, it is impossible to determine exact memory requirements for inexact data sizes. Therefore, it is important to monitor the sizes of the tablespaces, tables and extents within your local instance and extend the datafile size or the number of datafiles if necessary to avoid data insertion problems and unnecessary downtime in the future.

LOCALIZATION NOTE

The @task schema needs to be installed on an Oracle database that uses the LANG environment setting 'en_US.UTF-8'. If you are installing @task on a database server that is outside of the United States, it is likely that the default LANG setting on the database is something other than 'en_US.UTF-8'. For example, in Germany, the default LANG setting is most likely 'de_DE.UTF-8'. In these cases, you will need to set the LANG environment setting on your @task database to 'en_US.UTF-8' before creating the schema.

DATABASE CREATION

The steps to install @task on Oracle are: create the database, create the tablespaces, create tables within the database, import all of the setup data, and give access rights to the Web application.

NOTE: The instructions contained here assume a high level of expertise with Oracle and should be performed by an Oracle DBA.

CREATE THE DATABASE

Create a database for @task. The name chosen must follow the Oracle naming conventions for database names. This documentation assumes the name 'attask'.

1. At an Oracle console that is connected to the instance where the database is to be created, type:
 - a. Create database attask;
 - b. Switch to the newly created attask database by reconnecting to the database using the new SID.

CREATE THE TABLESPACES

The @task database uses two tablespaces, one for the data tables and one for the indexes. These are described below:

1. ATTASK_DATA - The tablespace that holds the @task data tables
2. ATTASK_IDX - The tablespace for the index tables

It is recommended that you do NOT change the tablespace names as the automatic update features of @task assume these two tablespace names. We recommend that each tablespace have an initial size of at least 80 megabytes for the core system install.

NOTE: It is also highly recommended that your tablespaces have autoextend=true.

You can create tablespaces through the Oracle Enterprise Manager or through an Oracle console using a script.

Included in the installation package is a script that can be modified to create the tablespaces in your environment. This file is called createTablespaces.sql. To modify this file:

1. Open 'createTablespaces.sql' in a text editor. You will see two 'CREATE TABLESPACE' commands.
2. You need to modify the values for the DATAFILE parameters to match your local system. These parameters use standard OFA directory structures for maximum portability. You need to make sure that the directory structure that you specify is compliant with the operating system Oracle is running on (for example, forward slashes '/' for Unix-based operating systems and back slashes '\' for Windows-based operating systems,) and that the directory structure exists.

The basic structure of the OFA directory should be something like:

```
/[mount_point]/APP/[application_name]/PRODUCT/[version]/application
```

This example uses a more simple form (from a Windows computer):

```
c:\u01\oradata\attask\
```

More information on OFA can be found in your Oracle documentation set.

3. The tablespace commands here specify default table storage sizes for tables that are created within the tablespace, but don't specify their own storage parameters.

Once the file is modified, you should be able to execute the script to create the necessary tablespaces.

SET UP DATABASE ACCESS RIGHTS

You can choose any user name and password. However, you will need to enter this user name and password in the @task installation wizard.

From an Oracle console you need to enter access rights to the user that @Task will use to connect to the database. Replace 'USERNAME' in the following statement with your information and replace the host name with the host provided as part of your @task License Key:

```
GRANT INSERT,SELECT,UPDATE,DELETE ON ALL_TABLES TO <USERNAME>;
```

DATABASE SETUP FOR MICROSOFT SQL SERVER 2005

This guide assumes that Microsoft SQL Server 2005 has already been installed, that users have a general knowledge of the SQL Server Configuration Manager, and Microsoft SQL Server Manger Studio applications are included in the install.

CREATE THE DATABASE

Using Microsoft SQL Server Management Studio, perform the following steps:

1. In the Object Explorer window on the left side, expand the correct server.
2. Right-click on **Databases**, select **New Database...** from the pop-up menu.
3. Enter a database name. The name can be anything. This documentation assumes the name is 'attask'.
4. On the Owner dropdown, select the **DB Login** that you want to use for the @task database. This will ensure that this Login has all the privileges needed. You must use the username and password associated with this Login in the @task install wizard.
5. On the Database files table, set an initial size of at least 50 MB for the Data file and 25 MB for the Log file.
6. Select **OK** to create the database.

ENSURE PROPER NETWORK SUPPORT

@task is a Java-based application and uses JDBC over TCP/IP to communicate with Microsoft SQL Server 2005. It is important that SQL Server 2005 is configured to support TCP/IP communication. Using the SQL Server Enterprise Manager, perform the following steps to ensure TCP/IP support is enabled:

1. In the console pane of the SQL Server Configuration Manager, expand SQL Server 2005 Network Configuration, then expand Protocols for <instance name>, and then double-click **TCP/IP**.
2. In the TCP/IP Properties dialog box, on the IP Addresses tab, several IP addresses appear, in the format IP1, IP2, up to IPAll. Scroll down and set the TCP Port of IPAll to the default value of 1433. This value is used in the @task configuration file (attask-config.xml).
3. Switch to the Protocol tab, and change the Enabled option to Yes. Click **OK**.
4. In the console pane, click **SQL Server 2005 Services**.

5. In the details pane, right-click **SQL Server (<instance name>)** and then click **restart**, to stop and restart SQL Server.

DATABASE SETUP FOR MICROSOFT SQL SERVER 2000

The same assumptions as detailed for Microsoft SQL Server 2005 apply for Server 2000.

CREATE THE DATABASE

Using Microsoft SQL Server Management Manager, perform the following steps:

1. Select a host. Select **New-> Database...** from the popup menu.
2. Enter the database name. The name can be anything. This documentation assumes that the name is 'attask'.
3. On the **Data Files** tab, select an initial size of at least 50 MB. Select the **Automatically Grow File** option, and specify a **Growth By Percent** value of at least 25%.
4. Select **OK** to create the database.

ENSURE PROPER NETWORK SUPPORT

@task is a Java-based application and uses JDBC over TCP/IP to communicate with Microsoft SQL Server. It is important that SQL Server is configured to support TCP/IP communication. Using the SQL Server Enterprise Manager, perform the following steps to ensure TCP/IP support is enabled:

1. Select the host on which the 'attask' database resides.
2. Select **Properties**.
3. On the **Server Properties** popup dialog, select the **General** tab.
4. On the General tab, select the **Network Configurations...** button.
5. Make sure that the TCP/IP protocol is in the **Enable Protocols** list.
6. Select **TCP/IP** from the Enabled Protocols list and select **Properties**.
7. The Default Port is the TCP port number that the SQL Server database communicates on. The default value is 1433. This value is used in the @task configuration file (web.xml).
8. Select **OK** until all popup dialogs close.
9. You may need to restart Microsoft SQL Server 2000 for changes to take effect.



@task

Chapter 2

Installing @task

This chapter contains the following sections:

- [Installing @task](#)
- [Accessing @task](#)
- [Upgrading from Earlier Versions of @task 4](#)
- [Running the Java Install Packages](#)
- [Configuring @task](#)

INSTALLING @TASK

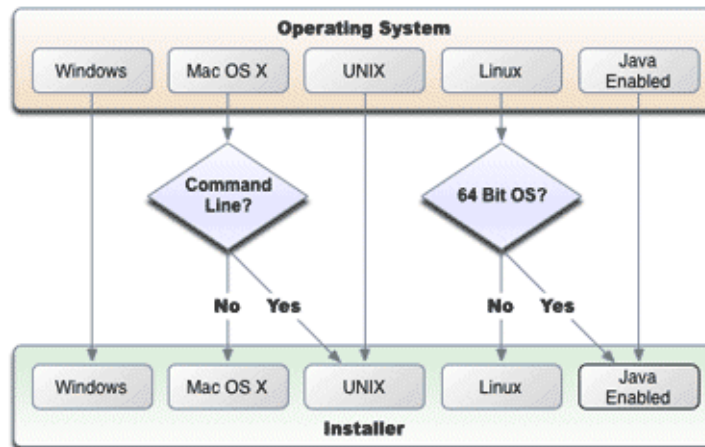
@task uses an installation wizard for the installation process. The following procedure explains how to install @task.

To install @task

1. Download and run the application files from:

www.attask.com/atv4

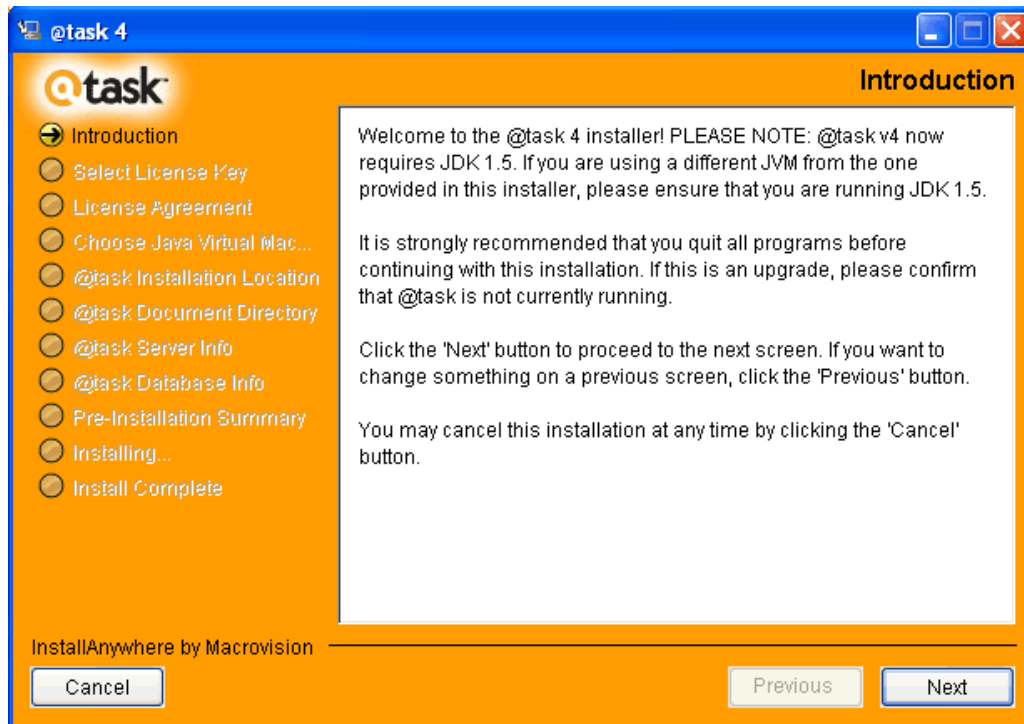
The following figure will help you determine which installer you should download



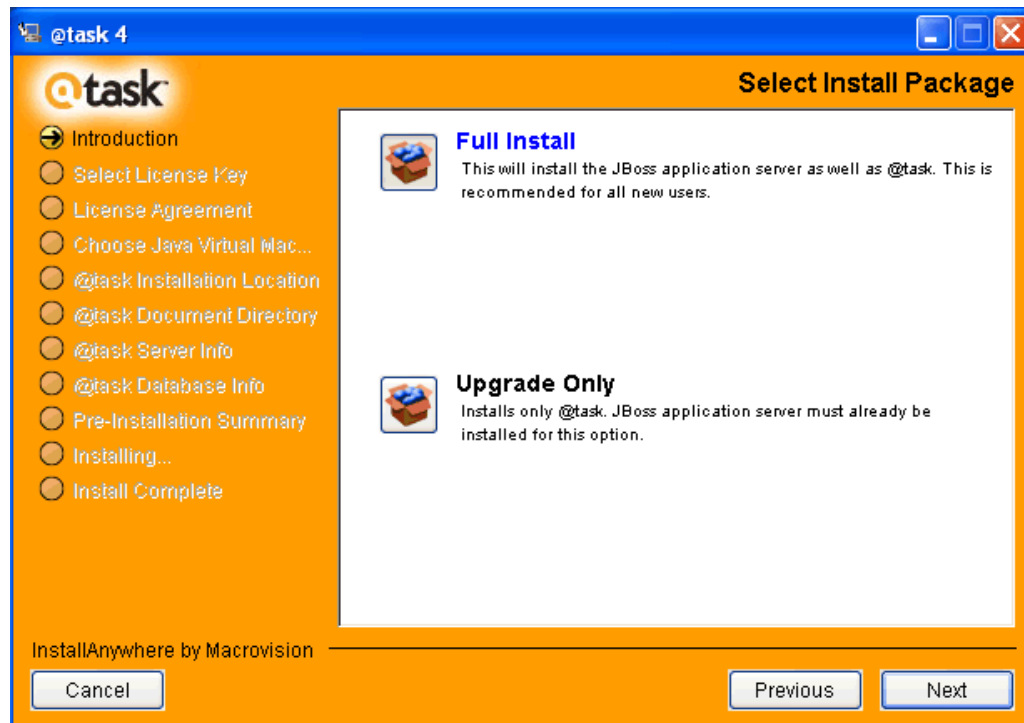
The file names are:

- Windows - attask-install.exe
- Linux/UNIX/AIX/Solaris - attaskinstall.bin
- Mac OS - Attask-install.zip
- To run the installer from a Linux command line, or to do a remote install, type **attaskinstall.bin -i console**. The steps and information are identical, but they use a command line format rather than a graphical user interface.
- For other Java-enabled operating systems use the following instruction set:
[See "Running the Java Install Packages" on page 21.](#)

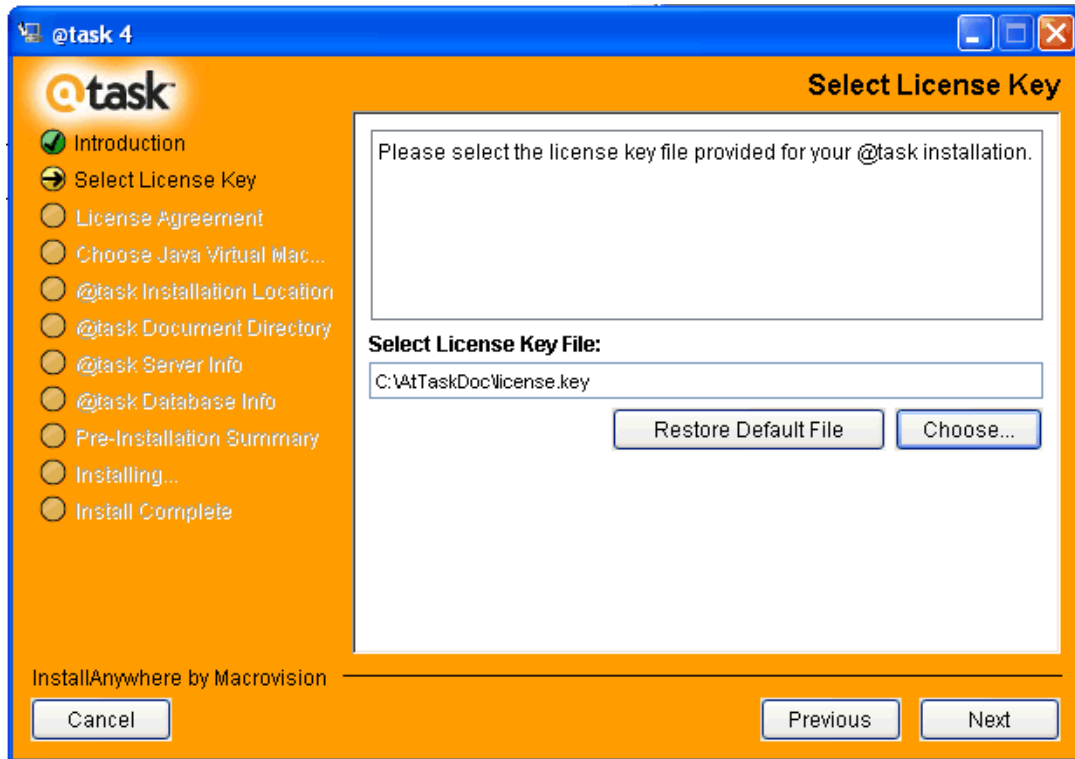
2. Read the Welcome screen then click **Next**.



3. Select **Full Install** to install @task Version 4. Select Upgrade Only to install using settings from a previous version. The upgrade feature is for @task version 4.x and later. It is not supported for versions prior to 4.0.



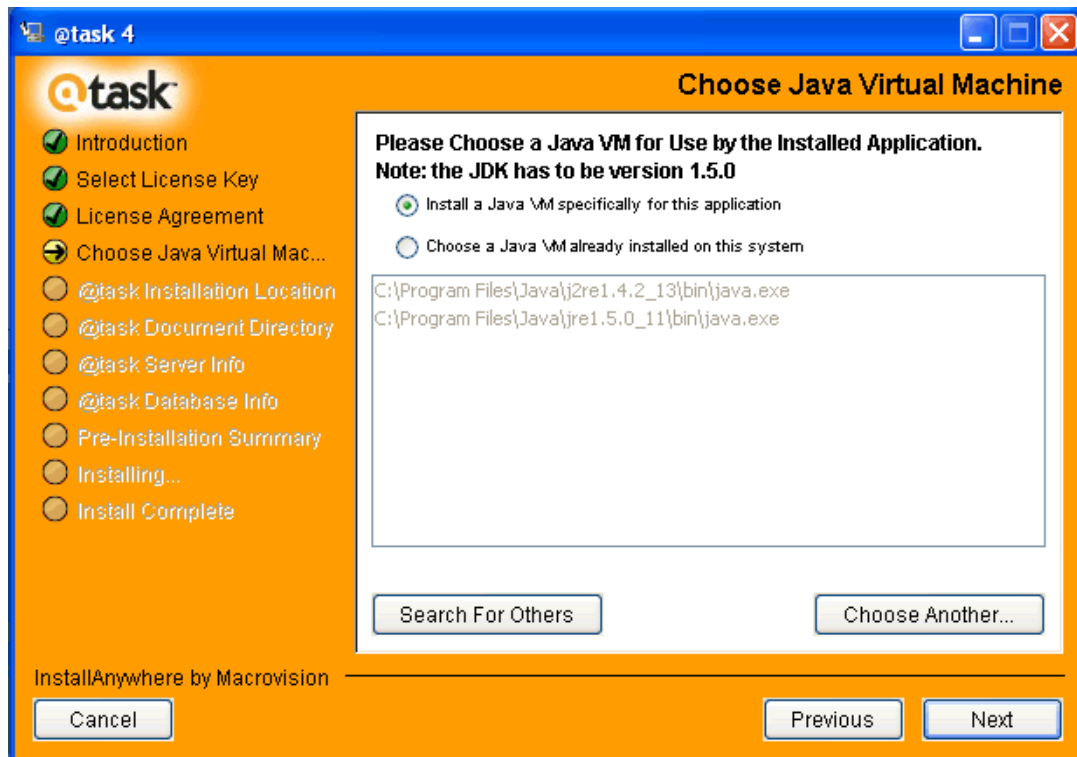
4. Type the path to the license key that @task has provided to you. Click **Choose** to navigate to the file on your computer. The license key name is License.key.



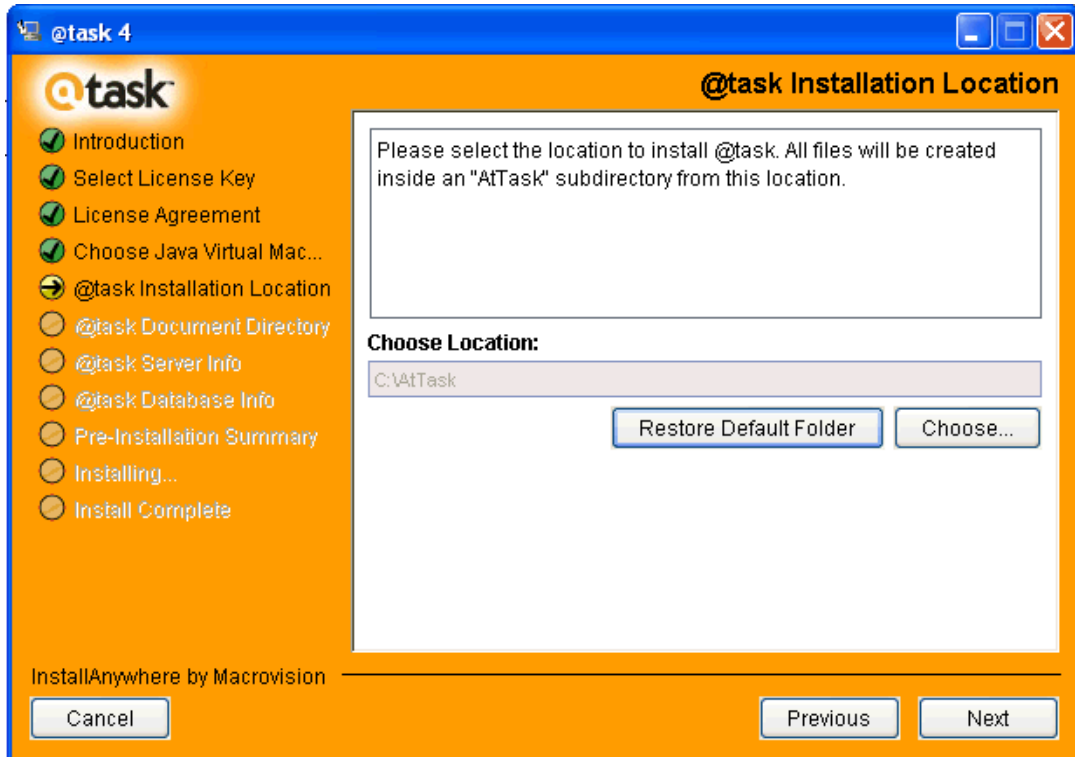
5. Ensure that you have read and that you understand the License Agreement before you click **I accept the terms of the license agreement**.



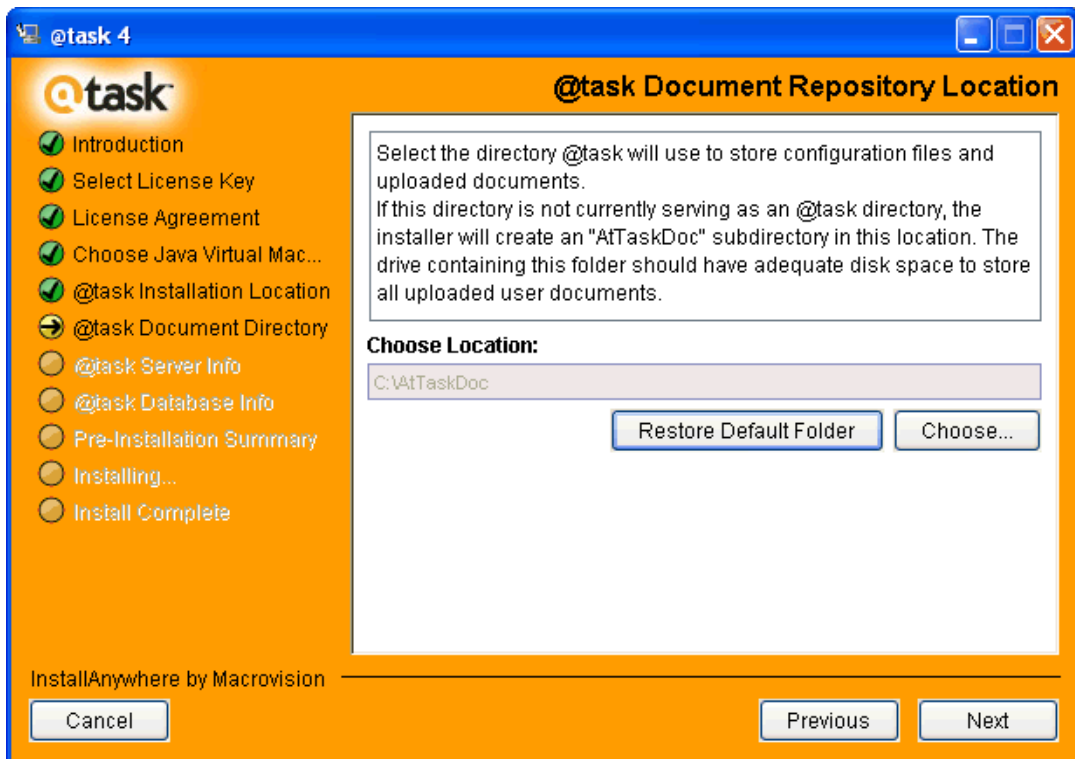
6. For Windows and Linux, select Install a Java VM specifically for this application. For Mac OS X and UNIX-related operating systems, click **Choose a Java VM already installed on this system**. You need to use Java version 1.5.0 or later.



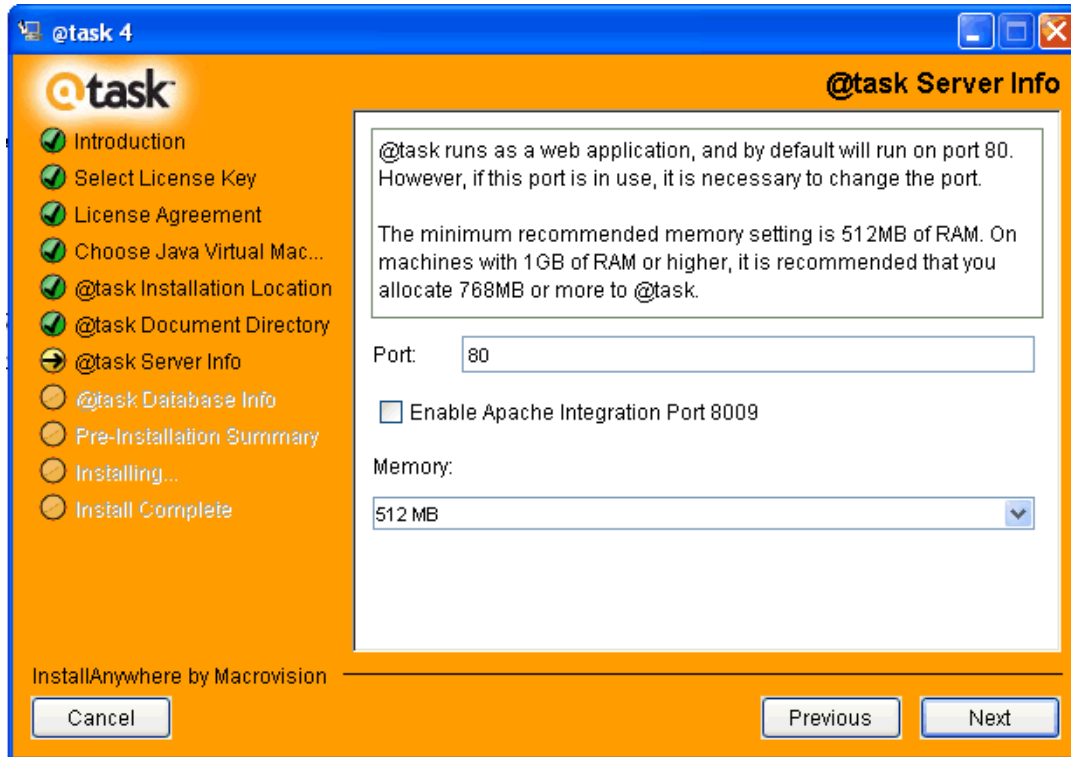
7. Click **Choose** to navigate to the folder to where you want to install @task. Click **Next** to accept the default location.



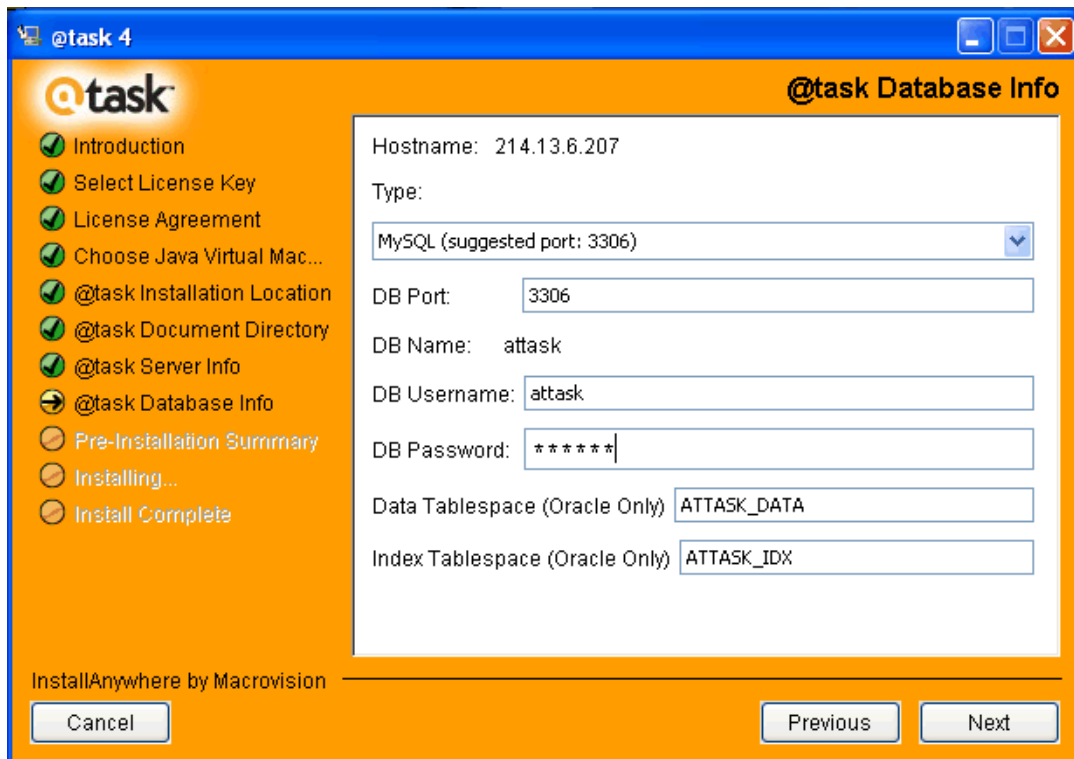
8. Click **Choose** to navigate to the folder to where you want to store @task configuration files. Click **Next** to accept the default location.



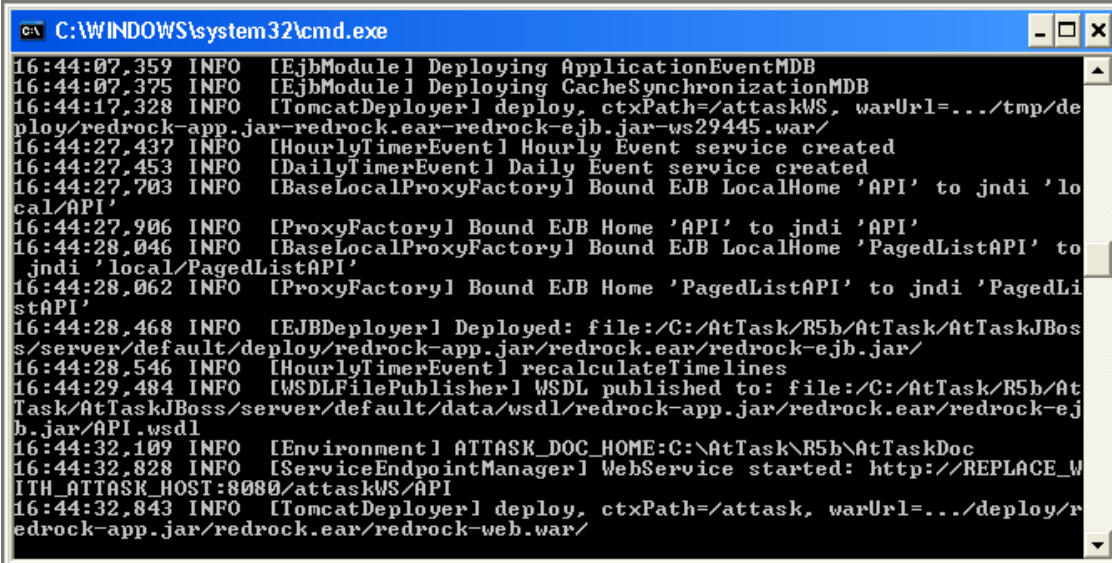
9. Select the port number that you want to use for you @task Web server. From the drop-down menu, select the amount of RAM that you want to allocate for @task to use.



10. Select the database type. Type a port number. Type a user name and a password for your database. If using Oracle, type the tablespace information.



11. Review the installation information, and then click **Install**.
12. When the installation process has completed, a screen appears that indicates that you need to start the @task Web server. Run **RunAtTask.bat** or **RunAtTask.sh** to start the Web server. A terminal window displays data about the install process. The process may take 30 - 90 seconds. Leave the terminal window open. This is the @task program. When the terminal window stops displaying new data, @task and the @task Web server are fully functioning. Near the bottom of the output you will see a line that contains 'ATTASK_DOC_HOME: C:\AtTaskDoc'. When this appears you will know that @task is loaded and can be accessed from a web browser.



```
C:\WINDOWS\system32\cmd.exe
16:44:07,359 INFO [EjbModule] Deploying ApplicationEventMDB
16:44:07,375 INFO [EjbModule] Deploying CacheSynchronizationMDB
16:44:17,328 INFO [TomcatDeployer] deploy, ctxPath=/attaskWS, warUrl=.../tmp/deploy/redrock-app.jar-redrock.ear-redrock-ejb.jar-ws29445.war/
16:44:27,437 INFO [HourlyTimerEvent] Hourly Event service created
16:44:27,453 INFO [DailyTimerEvent] Daily Event service created
16:44:27,703 INFO [BaseLocalProxyFactory] Bound EJB LocalHome 'API' to jndi 'local/API'
16:44:27,906 INFO [ProxyFactory] Bound EJB Home 'API' to jndi 'API'
16:44:28,046 INFO [BaseLocalProxyFactory] Bound EJB LocalHome 'PagedListAPI' to jndi 'local/PagedListAPI'
16:44:28,062 INFO [ProxyFactory] Bound EJB Home 'PagedListAPI' to jndi 'PagedListAPI'
16:44:28,468 INFO [EJBDeployer] Deployed: file:/C:/AtTask/R5b/AtTask/AtTaskJBoss/server/default/deploy/redrock-app.jar/redrock.ear/redrock-ejb.jar/
16:44:28,546 INFO [HourlyTimerEvent] recalculateTimelines
16:44:29,484 INFO [WSDLFilePublisher] WSDL published to: file:/C:/AtTask/R5b/AtTask/AtTaskJBoss/server/default/data/wsdl/redrock-app.jar/redrock.ear/redrock-ejb.jar/API.wsdl
16:44:32,109 INFO [Environment] ATTASK_DOC_HOME:C:\AtTask\R5b\AtTaskDoc
16:44:32,828 INFO [ServiceEndpointManager] WebService started: http://REPLACE_MITH_ATTASK_HOST:8080/attaskWS/API
16:44:32,843 INFO [TomcatDeployer] deploy, ctxPath=/attask, warUrl=.../deploy/redrock-app.jar/redrock.ear/redrock-web.war/
```

NOTE : Closing this terminal window will stop the service and disable access to @task through the web browser.

NOTE : If you want @task to run at startup you can run it as a service (see [“Running @task as a Service” on page 26](#))

ACCESSING @TASK

You must now access @task using a Web browser. You need to create an initial schema from the Web browser.

To connect to @task

1. In a Web browser type `http://<the IP address or URL of the Web server>:<port number>/attask`. For example, `http://214.13.6.207/attask`
2. On the initial screen, click **Create Schema**. You may need to wait several minutes for @task to create the schema.



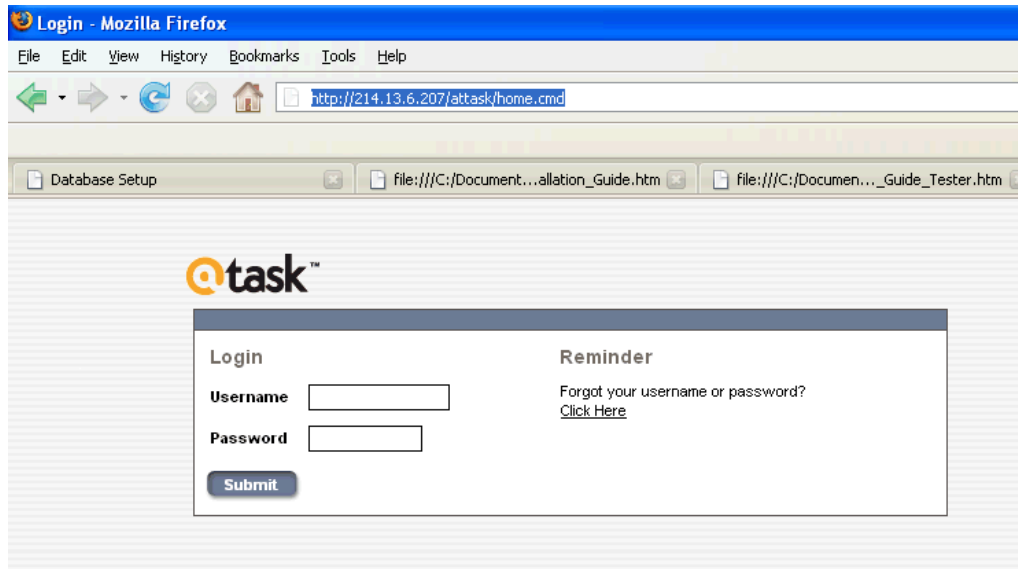
Schema Not Installed

Attribute	Value
Database Username	attask
Database TCP/IP Port	3306
Database Hostname	214.13.6.207
Database Name	attask

The database connection was made successfully, but no schema was found in the database. You need to create the database schema.

[Create Schema](#)

3. When the schema is complete, @task displays the Login screen. To log in, type the default user name: **admin**, and the default password: **user**, then click **Submit**.



UPGRADING FROM EARLIER VERSIONS OF @TASK 4

You can upgrade from the previous version of @task 4 to the current version without losing any @task data. To ensure that you don't lose your data, use the following procedure when installing @task 4. As a precaution, backup your database and your @task documents directory before starting the upgrade process.

1. Acquire a new license key and copy it over to old license key in the @task documents folder. This is only necessary for @task 4 clients upgrading from Professional to Enterprise editions.
2. Install the new version of @task to a location other than the current folder where you installed the previous version. **NOTE:** Do not nest the upgrade inside of the existing AtTask and AtTaskDoc folders. If you do you will not be able to run your instance of @task.
3. Run the installer, but select **Upgrade Only** rather than Full Install in the second screen. Please note that certain upgrades may require a full install.
4. During the install, provide the install wizard with the location of your new license key (if necessary) and the database that you have been using.
5. Delete the directory where you installed the old version of @task. The @task documents directory should not be in this folder.

RUNNING THE JAVA INSTALL PACKAGES

If using 64 bit Linux versions, or other Java-enabled operating systems that are not listed in the system requirements, you must install @task using the Java installer. You must execute the Java files from the command line.

INSTRUCTIONS (UNIX OR UNIX-LIKE OPERATING SYSTEMS)

- For Java 2, after downloading, type
`java -jar attaskinstall.jar`
- For Java 1.1, after downloading, type
`jre -cp attaskinstall.jar install`
- If that fails, try the following:
`java -classpath [path to]classes.zip:attaskinstall.jar install`
- If that fails, try the following:
`cd [to directory where attaskinstall.jar is located]`
`CLASSPATH=attaskinstall.jar export CLASSPATH java install`
- For csh-like shells, try the following:
`cd [to directory where attaskinstall.jar is located]`
`setenv CLASSPATH attaskinstall.jar java install`

INSTRUCTIONS (FOR OTHER PLATFORMS)

- Be sure you have Java 1.5.0_11 or later installed. You can download Java from java.sun.com.
- In a console window, change to the directory in which you downloaded **attaskinstall.jar** before running the installer.

N O T E : Your operating system may invoke Java in a different way. To start the installer, add **attaskinstall.jar** to your **CLASSPATH**, then start the main class of the installer named **install**.

CONFIGURING @TASK

To configure @task, refer to Chapters 13 and 14 of the *@task User's Guide*.



Chapter 3

Advanced Installation

This chapter contains the following sections:

- [Using Clusters](#)
- [Running @task as a Service](#)
- [Running @task at Startup Using Mac OS](#)
- [Secure Sockets Layer Setup for JBoss](#)

USING CLUSTERS

The ability to run multiple @task eservers against the same database is not supported by default. Unless each server is aware of other servers in the cluster and coordinates its activities with these other instances of @task, deploying multiple servers could result in data corruption and unpredictable behavior. Additional setup is required to make this work.

An @task cluster consists of 2 or more instances of @task running on different hosts that access the same database.

When installed properly, an @task cluster can dramatically increase scalability by distributing the workload across multiple servers to improve performance and availability by allowing seamless fail-over when a host goes down.

BASIC OVERVIEW OF CLUSTER FUNCTIONALITY

The first server to start becomes the master node. The master node is responsible for controlling the cluster. All cluster synchronization and communication is routed through the master node. If it goes down (or the connection is lost) a new node is elected as the master node. When the original master node is restarted it rejoins the cluster as a new node, but not as the master node. After a node is elected as master node, it remains the master node until it is shutdown.

Fail-over, in the case when a server goes down, happens automatically. @task API requests are automatically rerouted. There is no need to modify client code. HTTP requests will have to be retried manually by web server software or a network load balancer depending on the customer installation.

If the server that goes down is the master node, there may be a delay as JBoss elects a new master node and recreates the messaging hooks to support cluster communication. However, requests to the cluster are queued during this period, and requests will be serviced normally once the master node configuration is complete.

CLUSTER SUPPORT CHECKLIST

When setting up a clustered environment, all machines in the cluster must:

- Have the same operating system. JBoss clustering will not work unless all machines in the cluster have the same operating system installed.
Please note if your database is running on a separate server, it DOES NOT have to be running the same operating system as the servers running JBoss and @task
- Have the same versions of JBoss and @task. It is important to ensure that all servers in the cluster have identical copies of both JBoss software and @task.
- Share the same network subnet address and allow for IP Multicast.
- Are isolated from other potential instances of JBoss. The network should be properly isolated so that nodes outside your production cluster are not “discovered” by nodes running inside your production cluster. Enforcing this will eliminate any crosstalk that could occur with other machines on site being used for QA or staging purposes.
- Have access to a common file system. Because @task relies on direct file system access for many of its functions, it is critical that all servers have access to the ATTASK_DOC_HOME directory using the same path. This means that the ATTASK_DOC_HOME environment variable on each server should

contain a file system path that points to the SAME network-mounted directory shared by all servers in the cluster.

JBOSS CLUSTER SETUP

@task software is bundled with 6 separate cluster support folders. The naming convention for the folders is as follows: <operating system>-<database type>-cluster-support. It is important to match up the correct folder for the database.

Supported operating systems include:

- **unix** – refers to any version of Unix (Solaris, HP/UX, AIX, etc.), Linux (Red Hat, Debian, etc.), or Apple Macintosh OS X.
- **windows** – refers to any version of Windows (XP, 2000, 2003, etc.)

Please note that operating system refers to the operating system of the servers running JBoss and @task.

Supported databases include:

- **MSSQL** – Microsoft SQL Server 2000/2005
- **Oracle** – Oracle 9i or later
- **MySQL** – MySQL 5.0 or later

Please note that the database can run on a different operating system from that of the servers running JBoss and @task. It is not important which operating system the database is running on.

The included directories are:

- /unix-mssql-cluster-support
- /unix-oracle-cluster-support
- /unix-mysql-cluster-support
- /windows-mssql-cluster-support
- /windows-oracle-cluster-support
- /windows-mysql-cluster-support

JBOSS AND @TASK CLUSTERING SETUP

1. Backup your current JBoss installation. Find your JBoss installation directory (JBOSS_HOME) and copy the “JBOSS_HOME/server/default/” directory and its contents to another location on the file system outside JBOSS_HOME. This will allow you to revert the cluster configuration should you choose to do so.
2. In JBOSS_HOME, delete the “JBOSS_HOME/server/default/deploy/jms” directory on each server in the cluster. This directory, if left on any of the servers in the cluster, will conflict with the JMS setup used by the cluster.
3. Shutdown @task Make sure, if you have multiple instances already running, that all instances of @task that will be in the cluster are shutdown.
4. Copy the contents of “/<operating system>-<database type>-cluster-support” to the “JBOSS_HOME/server/default” directory on each server in the cluster. It is important that each server have the identical configuration files.
5. Please note, that the contents of this directory are meant to copy over the top of the JBOSS_HOME/server/default.

6. Start @task
7. Login to @task using an account with administrator privileges. Enable clustering in the @task Configuration by changing the configuration parameter 'Cluster Enabled' to 'yes'.
8. Restart all @task servers

CLIENT CONFIGURATION CHANGES USING THE @TASK API WITH CLUSTERING

This instruction assumes you are already familiar with how to connect to the @task API in a single-instance mode.

EJB API

The JNDI parameter "java.naming.provider.url" must be changed in order to connect to an @task cluster.

This value should be changed to a comma-delimited list containing the values <host address>:1100.

For example, suppose you have an @task cluster with 3 servers with hostnames host1, host2, and host3. The java.naming.provider.url property should have a value of:

```
java.naming.provider.url=host1:1100,host2:1100,host3:1100
```

This property will enable the API client to try each of these hosts, in order for an initial connection to the EJBHome interface. Once a connection is made, the API client will handle its own load balancing and fail-over. No additional code is needed on the client side to take advantage of these features.

SOAP API

Currently, there are no tools that allow for SOAP clients to automatically discover multiple hosts in an @task cluster or that handle automatic fail-over.

Connecting to an @task server in a cluster is exactly the same as connecting to a single server. Any rollover or load balancing policies must be implemented by the client.

FAQ

Clustering is not working and I'm seeing the following error in my logs:

```
ERROR [HypersonicDatabase] Starting failed
jboss:database=localDB,service=Hypersonic
java.sql.SQLException: General error: java.lang.NullPointerException
    at org.hsqldb.jdbc.jdbcUtil.sqlException(Unknown Source)
    at org.hsqldb.jdbc.jdbcConnection.<init>(Unknown Source)
    at org.hsqldb.jdbcDriver.getConnection(Unknown Source)
    at org.hsqldb.jdbcDriver.connect(Unknown Source)
    at java.sql.DriverManager.getConnection(DriverManager.java:512)
    at java.sql.DriverManager.getConnection(DriverManager.java:171)
```

This means that the Hypersonic database (the internal datastore that handles JMS messages) has become corrupt. To fix this problem, shutdown all @task servers in the cluster and delete the contents of the JBOSS_HOME/server/default/data/hypersonic directory. These files will be recreated once JBoss is restarted.

This will cause any outstanding messages to be deleted. If this problem persists, contact customer support.

RUNNING @TASK AS A SERVICE

If you install @task on a Windows computer, you can run @task as a service. This option has many advantages, including allowing @task to run on a computer without a user having to log in.

To start @task as a Windows service

1. Open the following file:

```
..\install\InstallAtTaskService.bat
```

NOTE : When running @task as a service, if you stop the service, it may take a few minutes to restart.

To update the Windows Service

1. Stop the service.
2. Uninstall the service from the command line. **NOTE:** Uninstallation must be done from the command line.
3. Reinstall with the update.

RUNNING @TASK AT STARTUP USING MAC OS

Use the following procedure to run @task at startup using MacOS

To run @task at startup

1. Open the terminal application.
2. Change directories to your AtTask installation:
`'cd PATH/TO/AtTask/Mac\SystemStarter/'`
3. Make a directory in StartupItems:
`'sudo mkdir /Library/StartupItems/AtTask4'`
4. Move the AtTask and StartupParameters.plist files into that directory:
`'sudo mv ./ * /Library/StartupItems/AtTask4/'`
5. Change the ownership of those files to root:
`'sudo chown -R root /Library/StartupItems/AtTask4/*'`
6. Change the group of those files to wheel:
7. `'sudo chgrp -R wheel /Library/StartupItems/AtTask4/*'`
8. Change the permissions of those files:
9. `'sudo chmod -R 755 /Library/StartupItems/AtTask4/*'`

@task will now start by itself when you restart your computer.

You can test this functionality by issuing the command:

```
'sudo SystemStarter -vdn start AtTask4'
```

You can manually start @task with the command:

```
'sudo SystemStarter start AtTask4'
```

You can manually stop @task with the command

```
'sudo SystemStarter stop AtTask4'
```

SECURE SOCKETS LAYER SETUP FOR JBOSS

This section describes the process of setting up JBoss to allow @task to run through SSL (Secure Sockets Layer) for encrypted network communications.

SETUP

Secure Sockets Layer, or SSL, is the standard for encrypted communication over the web. @task is designed to operate through SSL.

There are two necessary components to setup SSL:

1. Secure Certificate
2. JBoss Configuration

SECURE CERTIFICATE

SSL requires a signed certificate (X509 Certificate) to verify to clients the identity of the server. These signed certificates are obtained from a Certificate Authority, or CA.

If you want to generate your own “unsigned” certificate (keystore) file here is the command:

```
keytool -genkey -keystore chap8.keystore -storepass rmi+ssl -keypass rmi+ssl -  
keyalg RSA -alias chapter8
```

N O T E : JBoss uses ‘chap8’ as a default name feel free to change it. You may also change the alias ‘chapter8’
The chap8.keystore needs to be placed in the conf folder such as ‘jboss-4.0.5/server/default/conf’

JBOSS CONFIGURATION

SERVER.XML

Inside of the deploy/jbossweb-tomcat55.sar/ folder there is a server.xml file. You will need to edit this file to enable the https connector.

1. Inside of the server.xml file, you will need to find the SSL/TLS Connector configuration. It will look similar to the code below

```
<Connector port="8443" address="{ jboss.bind.address}"  
  maxThreads="100" strategy="ms" maxHttpHeaderSize="8192"  
  emptySessionPath="true"  
  scheme="https" secure="true" clientAuth="false"  
  keystoreFile="{ jboss.server.home.dir}/conf/chap8.keystore"  
  keystorePass="rmi+ssl" sslProtocol = "TLS" />
```
2. Uncomment the Connector code.
3. If your certificate keystore is not the default, “chap8.keystore”, then change the keystoreFile to point to your keystore file.

TESTING CONFIGURATION

After restarting JBoss, you should see something in the JBoss log that looks like this:

```
12:14:02,667 INFO [Http11Protocol] Initializing Coyote HTTP/1.1 on http-o.o.o.o-8443
```

SSL should now be working with jboss on port 8443.

Example: <https://235.46.18.79:8443/jmx-console/>

FAQ

Q: Is it possible to only use SSL?

A: Yes, after you enabled the SSL connector you may delete (comment out) the old non ssl connector.

Q: Is it possible to change the port SSL is on?

A: Yes, just change the port="8443" to the desired port number. Remember some systems require special permissions to open ports below 1024.