INTRODUCTION

Manual for version 1.5.41

Urbancode developed Anthill to make it easier to automate the build process among a group of developers using Apache Ant.

Anthill allows multiple users to work together and consistently access only the latest build, complete with changes from all programmers working on a project. This eliminates messy conflicts that arise when multiple users are working with the same source code!

We decided the best way to unlock the source code assets within a company was to dedicate a portion of an intranet to a project library. This library would contain home pages for every project within the organization.

Anthill creates a central project library that is updated manually and kept current, allowing the sharing of source code and compiled libraries within an organization.

Anthill performs a checkout from the source repository of the latest version of a project before every build and tags the repository with a unique build number. Anthill then updates a project intranet site with artifacts from the latest build.

It's a model developed from Open Source projects. Basically, Anthill instantly internalizes Open Source within an organization.

Anthill is compatible with version 1.3 and 1.4 of Apache Ant. Unlike other Apache Ant build tools, Anthill is designed to be as non-intrusive as possible and does not require cumbersome build additions.

Anthill is Open Source and released under a Mozilla-like license.

CHAPTER 1:

SYSTEM REQUIREMENTS

Anthill Server Requirements

- Anthill uses **Apache Ant**, which is packaged with the program.
- Anthill requires **Java 1.2** or later.
- **Apache Tomcat** is the recommended Java servlet container

Servlet Containers

- Anthill may be configured for any Java servlet container, but has only been tested with Tomcat, (versions 3.2.2, 3.3.1, and 4.0.2).

Anthill Client Browser Requirements

- For the administration of the system, you can use a regular web browser. **Microsoft Internet Explorer** or **Netscape 4+** are recommended, but **Mozilla** and **Opera** will also work.

CHAPTER 2:

DOWNLOADING ANTHILL

You may download the most recent version of Anthill by visiting the Download Directory at

http://www.urbancode.com/projects/anthill/download/.

Anthill is available with or without the source for each public build.

View the date in the right column to select the version you want. Scroll down for the newest version (Latest date will be the newest public version.)

CHAPTER 3:

INSTALLING ANTHILL & RUNNING PROJECTS

Installing Anthill

You've downloaded Anthill and now you're ready to install it.

Depending on whether you have downloaded Anthill with or without the source code, you will have one of the following two archive files:

- anthill-x.xx.tar.gz (normal version)
- anthill-1.x.xx.src.tar.gz. (version with source code)

Installing Anthill from the downloaded file is a simple, 4-step process.

1. Decide where to install Anthill
2. Extract downloaded Anthill file
3. Deploy Anthill WAR file to your servlet container
4. Open Anthill

Decide where to install Anthill.

Anthill should be installed on a central server that everyone working on your projects can access. The centrality of the server is essential for Anthill to manage the build processes for an entire development group.

The libraries for your version control system client should be installed on this server. The version control client executable must be in the path. Apache Tomcat or another Servlet container must also be installed on this machine.

Extract downloaded Anthill file

After choosing what machine to install Anthill on, create an Anthill directory, and extract the downloaded file there.

- For Windows, we recommend creating a "`C:\Programs\anthill-`" directory. The default Windows location: "`C:\Program Files`" contains a space character that may cause unforeseen problems in some applications.
- For Unix users, we recommend creating a '`/usr/local/anthill-`' directory.

Deploy Anthill WAR file to your servlet container

After extracting Anthill, the web application portion of Anthill must be deployed to your servlet container. The `anthill.war` file contains this portion of Anthill. The `anthill.war` file will be in the `\anthill-\dist` folder after extracting Anthill.

If you are using Apache Tomcat, copy the anthill.war file to Tomcat's `webapps` directory, at "`\\webapps`.

> *Note: If you are a Tomcat user upgrading from an older version of Anthill, Tomcat will have created the folder, "\\webapps\anthill" on the earlier install. Delete this folder before copying the new anthill.war file into Tomcat's webapps directory.*

Follow your servlet container's directions for deploying a `.war` file if you're using a servlet container other than Apache Tomcat.

Open Anthill

Anthill's user interface is viewed in a web browser. To open Anthill, start your servlet engine and open a web browser to your Anthill server's URL. Your Anthill server's URL will consist of the location of your server and the anthill directory. (Example: `http://localhost:8080/anthill/`).

The first time you use Anthill, you will be prompted to enter the Anthill home directory. Specify the absolute path to this directory. (Example: "`C:\programs\anthill-`" or "`/usr/local/anthill-`)"

You're now ready to set the Main Antill Properties, and begin using Anthill.

The first time you have installed Anthill, you will be prompted to enter the absolute path to the Anthill installation. This information is stored in the anthill.properties file in the User directory.

Running the “Anthill Example” Project

Anthill ships with an example project called “Anthill-Example”. This project is preconfigured and exists in a CVS repository hosted by Urbancode. This is probably the easiest way to see what Anthill does and to start playing around with it.

Configure Anthill for your Email Server

Before using “Anthill-Example” Project, you’ll need to configure one or two things pertaining to your Email setup to allow full use of its features. For example, you should configure the address of your email server so that Anthill can send out build notification emails. And you may want to configure Anthill's url if it is installed somewhere other than http://localhost:8080/anthill/.

To configure both of these properties, follow the "Anthill Properties" link from Anthill's main page. The email server address (either IP or machine name) goes into the "anthill.mail.host" property. If you want to change the from address on the emails that anthill sends out, then change the "anthill.mail.from" property value. The anthill url, which is used to construct links included in notification emails is configured via the "anthill.server" property. Once you have the properties set, press the "Submit Query" button to save and go back to the main screen.

Make sure you have CVS installed and in your path

The “Anthill-Example” Project uses CVS. Before you can build the Project, you must make sure you have a CVS client installed on your machine and in your

path. To test this, bring up a DOS prompt and type cvs -- if it can't find the command then your cvs client is either not in your path or you do not have a cvs client. To obtain a cvs client, download WinCVS (on windows) from http://www.wincvs.org . If you add cvs to you path after you've started your servlet container (Tomcat), then you must restart the servlet container.

Once you have verified that cvs is installed and in your path, you need to log in to the CVS server used by the "Anthill-Example" project. To do that, type on the command line:

cvs -d

:pserver:anthill-example@cvs2.urbancode.com:/usr/local/anthill-test login

Then press enter. When prompted for a password, type

anthill-example

Then press enter. Now you're all set to build the “Anthill-Example” Project.

Build “Anthill-Example” Project

To tell Anthill to build the "Anthill-Example" project, make sure your servlet container (Tomcat) is started, then go to http://localhost:8080/anthill/ in your browser and press the "build" link next to the project's name. In the build screen, make sure to check the "Force Build" box, then press "Submit Query". Once you do that, you should be able to see quite a bit of output in your Tomcat console. First Anthill is going to checkout the latest version of the project sources from the CVS server, then Anthill will increment the version number, then build the project, and then send out the build notification email.

Once that concludes, press the "refresh" link at the top of Anthill's main page. There should be a green bar in the row with the "Anthill-Example" project now. The green bar indicates that the project built successfully.

View Build Artifacts

Now, lets take a look at the results of the build. Click the "Anthill-Example" link. This link takes you to the project intranet created by Anthill. Any artifacts of the build can be placed in the project intranet. You may notice that the Ant build log is there, a revision log (which is empty since there were no revisions

since the last build), as well as javadocs in the api directory, junit test results, etc. Click the links for the directories and then click the link for the index.html file within each of the directories. Use your browser's back button to get back to the project intranet or to the Anthill main page.

Configure “Anthill-Example” Project to Send Emails

If you would like to configure the "Anthill-Example" project to send a build notification email to you, click the "Edit" link next to the project name on the main screen. Now you should see the project properties screen. This is where all the project properties can be configured. You can read more about this in the manual; all we are going to do is add you to the users list. About half way down the page, there is a property called "anthill.users". It has two text fields next to it. In the first text field enter your name. In the second text field enter you email address. When finished, press the "Update" button on the bottom of the page to save properties and return to the main screen.

To build the project again, hit the "build" link, check the "force build" box, and let it run. You should now get an email with the build results at the completion of the build.

Now start using Anthill for all your projects

The "Anthill-Example" sample project demonstrates the way Anthill works. Of course, the real value from Anthill comes when you set up Anthill to manage builds of projects that are in active development. That's when you can tell Anthill to build a project on a schedule and no longer have to manually force builds via the "build" link.

To set up your own projects and to learn about some of the other features within Anthill, read the rest of the User Manual and/or subscribe and post to the anthill users mailing list (http://www.urbancode.com/projects/anthill/maillist.jsp). Anthill supports all the leading source code control systems, so even if you do not use CVS, you can still set up your own test project within Anthill.

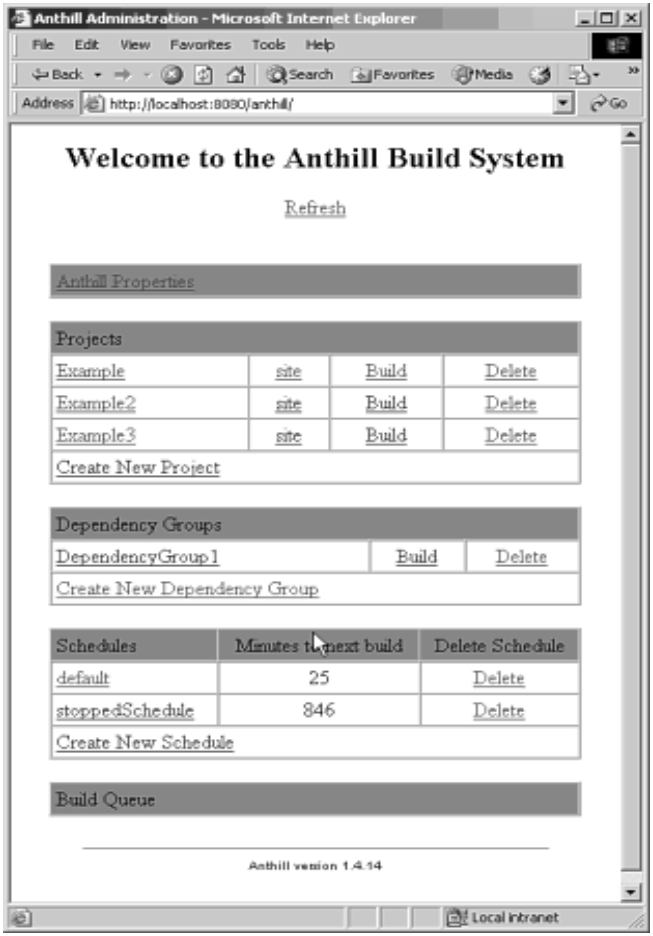
CHAPTER 4:

STARTING ANTHILL

Once Anthill is installed on your computer and it's started up in Tomcat, you can open it from a browser directed to the address you specified as the anthill.server in the main Anthill properties.

footnote: This should actually be pointed to the URL determined in the Tomcat configuration file: “server.xml” Any change in this configuration file has to be duplicated in the Anthill configuration so that the URL links work correctly in Anthill email notification. How this is listed in the configuration file should be duplicated in the anthill.server property in the Anthill properties.

CHAPTER 5:
NAVIGATING ANTHILL



Pull up Anthill in a browser to view the main screen.

Anthill is broken into 5 sections where you can navigate Projects, and the attributes Anthill uses to control your builds:

- Main Anthill Properties
- Projects
- Dependency Groups
- Schedules
- Build Queue

Create a new Project, Dependency Group, or Schedule

To create a new Project, Dependency Group, or Schedule, click on the "Create New Project" link at the bottom of the corresponding list.

This link will open a new properties screen. Enter the requested properties to create the new project. See the section on Projects, Dependency Groups, or Schedules for more information about requested properties.

Delete a Project, Dependency Group, or Schedule

To delete a project, click "Delete" beside the Project, Dependency Group, or Schedule to be deleted.

The registry file associated with Project, Dependency Group, or Schedule will be deleted from the system. You will be asked if you are sure, before Anthill performs the delete.

You will not be able to delete a schedule if a project or group is currently using it.

You will not be able to delete projects if they belong to a dependency group without deleting the association withing the dependency group first.

Warning: Deleted Projects, Dependency Groups, or Schedules will not be retrievable after delete.

Open a Project's website or artifact Directory

Beside each project is a link to a project's website.

This link will be to the address specified in the Project Properties as the `anthill.publish.url`. If this property is left empty, the site link will default to the address specified in main Anthill property: `anthill.publish.dir.default`. The site link will allow the project artifacts to be viewed.

Edit Properties, Projects, Dependency Groups, or Schedules

Click on the name of the Project, Dependency Group, or Schedule to edit its properties. This will bring up the properties page. Most fields may be edited at any time.

The delete checkbox

Some fields have a delete checkbox to remove previously entered data. To use these, click the checkbox and hit the submit button at the bottom of the page. This will delete selected lines.

Do an Unscheduled Build

Click “Build” in the far right column of the project category to force an unscheduled build for a project. This can be done at any time, and will run Ant’s build script.

When the next scheduled build time arrives, Anthill will check to see if modifications have been made since the last build (good or failed.) If changes have not occurred, Anthill will not do another build unless Force Build is checked.

Force Build checkbox

If no changes have been made since the previous build, and you still wish to do another build, you must use this checkbox to do the build.

Build as Version

Use the build as version box to specify a build number.

Note: This should be in the proper format as defined by the `anthill.version.adapter`.

Refresh Main Screen

To refresh the main screen, click the “Refresh” link at the top of the main screen beneath the “Welcome to the Anthill Build System” greeting.

This will refresh the page without reposting form data from the user’s last action, and show projects currently in build queue.

Build Queue

The Build Queue is at the bottom of the main screen. When a project goes into a scheduled build or a forced build, it will appear in the build queue.

When the build is complete, the project will disappear from the build queue. If no projects are currently being built, this list will be empty.

Build Queue
Example2 -- Being built

You will need to hit refresh for page to update and view precisely whether or not a project is currently being built.

CHAPTER 6:

SETTING THE MAIN ANTHILL PROPERTIES

At the top of the main screen is the link to the main Anthill properties. These settings control how the system accesses Anthill.

anthill.ant.home

Enter location of Ant `.jar` file

anthill.version.adapter

Enter class of version adapter.

Currently, the UrbanCode Version Adapter is implemented, and the default value should not be changed. The option to change it is there should someone have a desire to develop their own adapter.

This controls the format of the version number Anthill uses during scheduled builds.

This value defaults to:

- major.minor.build (for example 4.5.234)

Anthill only increases the build number during scheduled builds. To increase major or minor build numbers a build must be forced. Refer to section on unscheduled builds for information on how to force a build.

When building with branches with the default anthill.repository.adapter, the format of the version number will be different. There will be only one number, usually prefaced by the branch name.

Example: `Pluto_Branch-36`

anthill.repository.adapter

Specify the default Anthill repository adapter Anthill should use. The repository adapter is responsible for interaction with the source control repository. The same field is on the Project Properties screen, and allows user to specify individual options for each project.

Anthill has built in repository adapters for CVS, Merant PVCS, Perforce, and Microsoft Visual Source Safe, Starteam, and MKS Source Integrity. The following are the repository adapters included with Anthill:

- `com.urbancode.anthill.adapter.CVSRepositoryAdapter`
- `com.urbancode.anthill.adapter.PVCSRepositoryAdapter`
- `com.urbancode.anthill.adapter.PerforceRepositoryAdapter`

- com.urbancode.anthill.adapter.VSSRepositoryAdapter
- com.urbancode.anthill.adapter.MKSIntegrityRepositoryAdapter
- com.urbancode.anthill.adapter.FileSystemRepositoryAdapter

anthill.publish.dir.default

Specify the location to place build artifacts after a build is completed. Each project starts with a default value of the publishDir, a sub-directory of the main Anthill directory.

anthill.mail.host

Specify the mail server IP address or DNS name.

anthill.mail.from

Specify the name that will show up on automatic emails sent by Anthill after builds.

anthill.mail.from property determines the address that will show up on notification emails Anthill sends out after a build is complete

anthill.server

Specify the URL address where Anthill will be accessed on server.[1]

Note: This should actually be pointed to the URL determined in the Tomcat configuration file: server.xml. Any change in this configuration file has to be duplicated in the Anthill configuration so that the URL links work correctly in Anthill email notification. How this is listed in the configuration file should be duplicated in the anthill.server property in the main Anthill properties

This will be the address you use to access Anthill with your browser.

Note: context name in the URL has to be the same as the name of the .war file.

anthill.dependency.target

Enter the Ant target to execute when building dependencies.

This allows the user to use a different build process for stand-alone projects and projects that are part of a dependency group.

anthill.dependency.artifact.property

Enter the property name to be passed to the Ant script when executing a dependency target whose value will be the directory containing the build artifacts

For example, if you used the default property of `dependency.artifact.property` then you would refer to the build artifact directory from your Ant script as: `${dependency.artifact.dir}`

CHAPTER 7:

SETTING PROJECT PROPERTIES

anthill.project.name

Enter the name of the new Anthill project.

When naming a project, it is advisable not to use spaces. This is known to cause problems on some platforms.

anthill.version.adapter

View `anthill.version.adapter` in the Main Anthill Settings for more information.

The `anthill.version.adapter` in the Project settings is configured slightly differently than the adapter in the Main Anthill properties, to allow the user more control should they wish to use a custom version adapter.

version.file

We recommend that you keep the version number within a file in the repository. Although it is not required, it is a safer practice. The version file is a text file that contains the version of the project. To do this, create a file called <project name>.version in the source repository. (Example: “MyProject.version”). Use this field to specify the path to that file relative to the project root directory.

The general format of the version file is:

<prefix><buildNo><postfix>

where

<prefix>can be any String like "MyProduct" or "1.0."

<buildNo>has to be a number like 1 or 50 or 1150

<postfix>can be any non numeric String like " alpha" or " beta"

So if you have a version such as 1.2.3.4, then 1.2.3. is the <prefix>, 4 is the <buildNo> and the <postfix> is an empty string. The UrbanCodeVersionAdapter looks for the build number starting at the end of the string and working its way forward. The build number is the first numeric portion that it encounters. So you could have a version such as 1.2.3.1150 alpha and then the build number would be 1150.

Using a version file also allows you to configure multiple Anthill projects to point to the same source. For example, you might want to have one Anthill project build target_A every hour, and another Anthill project (pointing to the same source) build target_B every night. Having a separate version file for each target would make it easier to maintain consistent versions for both projects.

anthill.repository.adapter

See anthill.repository.adapter in main Anthill settings for more information.

The following are the repository adapters included with Anthill:

- **com.urbancode.anthill.adapter.CVSRepositoryAdapter**
- **com.urbancode.anthill.adapter.PVCSRepositoryAdapter**
- **com.urbancode.anthill.adapter.PerforceRepositoryAdapter**
- **com.urbancode.anthill.adapter.VSSRepositoryAdapter**
- **com.urbancode.anthill.adapter.MKSIntegrityRepositoryAdapter**
- **com.urbancode.anthill.adapter.FileSystemRepositoryAdapter**

If your repository adapter is not already selected, type the name of the repository adapter exactly as it is named above. Click update at the bottom of the screen.

After refreshing the screen, you will be able to click the link to configure the Repository Adapter. See directions below for your adapter for information on the properties requested.

com.urbancode.anthill.adapter.CVSRepositoryAdapter

If you're using CVS as your version control system, you'll want to choose this repository adapter.

repository.cvs.work.dir

Enter the name of the directory to which CVS will check out the project.

This directory is relative to Anthill's home directory. This is the location Anthill will use to check out files.

repository.cvs.module

Enter this project's CVS module.

The paths of the version file, build script, and publish script, are all relative to the CVS module. For example, if the CVS module is Anthill and the build script is `Anthill/build/build.xml`, the location of the build script is `build/build.xml`.

repository.cvs.root

If using CVS, enter the CVS ROOT used to log in to CVS.

This is the CVS connection information with the following information:

- the authentication method
- user name @ CVS server
- directory of CVS server

Note: Please keep in mind that if you are using pserver authentication, then the user must log into CVS on the machine running Anthill. (Logging in manually will allow the CVS client to store the CVS password for future use.)

repository.cvs.anthill.user

Enter the name of the CVS user account used by Anthill.

repository.cvs.branch

Enter the name of the CVS branch that Anthill is to build.

Most of the time this will be blank.

com.urbancode.anthill.adapter.PVCSRepositoryAdapter

If you're using PVCS as your version control system, you'll want to choose this repository adapter.

repository.pvcs.work.dir

Specify the name of the directory to which PVCS will check out the project. This work directory will override the working directory of the project in PVCS.

repository.pvcs.projectDir

Specify the full path to the directory to the PVCS project database.

repository.pvcs.project

Specify the project name. This is the folder name used in PVCS.

repository.pvcs.archiveDir

Specify the absolute directory to where the PVCS archive for the project is located.

repository.pvcs.branch

Specify the name of the PVCS branch that Anthill is to build. (Most of the time this will be blank.) This can be repetitive if the branch name is also the module name. But this isn't always the case. This parameter will be part of the local project directory on your local project directory on your local system. The module name is solely used for accessing PVCS' depot.

com.urbancode.anthill.adapter.PerforceRepositoryAdapter

If you're using Perforce as your version control system, you'll want to choose this repository adapter.

repository.perforce.P4PORT

Specify the port used by Perforce. This could be something like: `macineName:1666`, or even just `1666`

repository.perforce.P4ROOT

Specify the Perforce root. Do not enter the client's root.

repository.perforce.P4CLIENT

Specify the Perforce client set up for Anthill to use. Anthill only references this client information, the current client is not actually set to this value.

repository.perforce.P4USER

Specify the name of the Perforce user for this client.

repository.perforce.P4PASSWD

Specify the password for the P4USER. If this is left blank, Anthill will assume the user does not require a password.

repository.perforce.P4VIEW

Specify the Perforce View to be used. This field is optional but if it's left blank, Anthill will assume the project is the entire client root. Ex. //Anthill/projects/Anthill/... or //depot/projects/Anthill.

This property will also be appended to the working directory. For example, If the client root is /usr/anthill and you specify a view of //Anthill/projects/Anthill/..., the working directory will be /user/anthill/projects/Anthill.

com.urbancode.anthill.adapter.VSSRepositoryAdapter

If you're using Visual Source Safe as your version control system, you'll want to choose this repository adapter.

repository.vss.work.dir

Specify the name of the directory to which VSS will check out the project. This directory is relative to the Anthill home directory.

repository.vss.module

Specify the name of the VSS module that stores this project. The paths of the version file, build script, and publish script are all relative to the VSS module.

For example, if the VSS module is Anthill and the build script is `Anthill/build/build.xml`, then the location of the build script is `build/build.xml`

repository.vss.root

Specify the VSS Root used to log in to VSS, or the `ssdir` environment variable.

repository.vss.anthill.user

Specify the name of the VSS user account used by Anthill.

repository.vss.user.password

Specify the password for the VSS user account specified by the previous field if required. If no password is entered, Anthill will assume it is not required.

repository.vss.branch (optional)

Specify the name of the VSS branch that Anthill is to build. (Most of the time this will be blank.)

repository.vss.home

Specify the location of the `ss.exe` file.

repository.vss.dateFormat

Specify the date format to be used for VSS history command. The default format is “MM/dd/yyyy;hh:mma”

com.urbancode.anthill.adapter.StarforceRepositoryAdapter

If you’re using Starforce as your version control system, you’ll want to choose this repository adapter.

repository.starteam.work.dir

Specify the name of the directory to which Starteam will check out the project. This must be a directory path relative to the Anthill home directory.

repository.starteam.access

This is the string used to connect to the Starteam database, it is modified as needed internally.

repository.starteam.passwd.file (optional)

Specify the password file that Starteam uses for all commands. If you include your password in the access string above, you do not need to provide a password file.

If you provide a password file, you can omit the password part of the access string. The access string would be `username@hostname:endpoint/projectName/viewName/folderHieracarchy` in that case.

com.urbancode.anthill.adapter.MKSIntegrityRepositoryAdapter

If you're using MKS Source Integrity as your version control system, you'll want to choose this repository adapter.

repository.integrity.work.dir

Specify the absolute path to the project's working directory.

repository.integrity.project.file (optional)

Specify the name of the Source Integrity branch that Anthill is to build. (Most of the time this will be blank.)

com.urbancode.anthill.adapter.FileSystemRepositoryAdapter

Use this repository if you're not using a version control system. Sometimes users who are anxious to just get Anthill up and going will use this to run Anthill "unconnected" while they experiment with the features of Anthill.

repository.fileSystem.work.dir

This is the only attribute you need to specify with the File System Repository. Enter the absolute base directory of the project files.

anthill.build.script

Enter the path to the build script in the project root directory.

anthill.build.tag

Anthill tags the source repository within the build scripts by default.

Previous versions did not have this option, and some users complained that there were times they would only want Anthill to tag the source repository for release builds, or weekly builds.

Under most situations, you would want to keep this tagging there, as it is generally pretty useful.

Click no button for each project you wish to tag in the source repository if you choose to turn off source tagging.

anthill.publish.script (optional)

Enter the path to the publish script in the project root directory. If the user doesn't specify a publish script, they should do the publishing in their build script.

Not specifying this parameter will cause Anthill to skip the whole publishing step.

anthill.publish.dir

Enter the path to which the build artifacts are to be moved.

Most of the time this can be left empty. By default, Anthill will place all publish artifacts in the `publishDir/${project.name}` directory under the Anthill root. Anthill passes the publishDir property to the Antl build script. The Project artifacts are then available via browser at the url:

- `/anthill/projects/${project.name}`

anthill.publish.url

Enter the url where the build artifacts are available.

Most of the time this can be left empty. By default, Anthill will place all publish artifacts in the `publishDir/${project.name}` directory under the Anthill root. Project artifacts will then be available via browser at the url:

- `/anthill/projects/${project.name}`

anthill.users

Enter users that should receive build logs each time this project is built. Enter the user's name in the first box, and the user's email address in the second box.

anthill.mail.policy

The anthill.mail.policy allows you to specify when Anthill send's automatic build notification. You may select to send emails on all builds, or on failed builds only.

anthill.build.ant.params

Enter parameters to be passed to Ant during the build process. These are passed in three types of command line parameters to Ant as in the following example parameters:

- `flag`

Example: `verbose`

- `DPropName=PropValue`

Example: `Dant.home=..\lib\ant1.4`

- `PropName PropValue`

Example: `logfile logFileName`

anthill.publish.ant.params

Enter parameters to be passed to Ant during the publish process. These are passed in three types of command line parameters to Ant as in the following example parameters:

- `flag`

Example: `verbose`

- `DPropName=PropValue`

Example: Dant.home=..\lib\ant1.4

- `PropName PropValue`

Example: `logfile logFileName`

anthill.java.extra.options

Specify any properties to pass to the Java executable. See the documentation for your JVM for more information on what this could include.

anthill.schedule

This dropdown box specifies which build schedule will be used for this project. If a new schedule is needed, refer to the instructions for the Schedules.

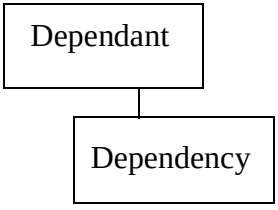
CHAPTER 8:

DEPENDENCY GROUPS

Anthill allows users to specify different projects as dependencies of other projects.

Doing builds of dependent projects often becomes cumbersome because a dependant needs the artifacts from its dependency builds to successfully build. When there is more than a one-to-one relationship, it simply becomes unmanageable to do it manually.

Anthill allows users to set up dependency groups to coordinate the build order of related projects and eliminate the need to manually manipulate the build artifacts of related projects.

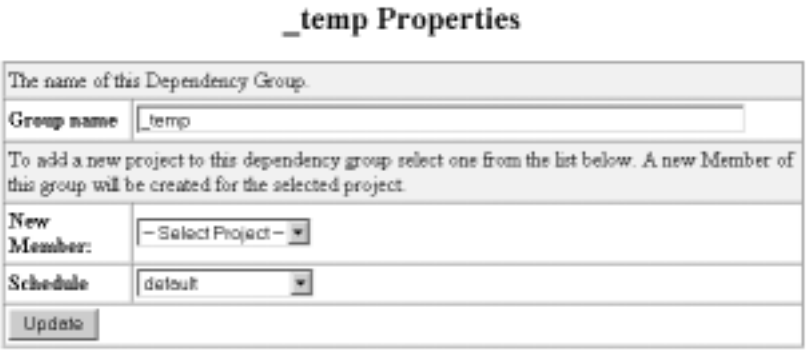


Dependency group establishes a "dependant" (project) and a "dependency" (sub-project) relationship between projects.

Anthill automatically builds all dependencies first and makes the build artifacts available to the dependant.

Create a new dependency group

Click on the "Create New Dependency Group" link at the bottom of the screen. This will open a new properties screen for dependency groups.



Group Name

Name your group in this text box. This is how your group will show up on the dependency group list on the main screen.

New Member

Select the first project to be a member of your dependency group from this dropdown box. All projects currently created will be in the drop-down menu. Click update to add the member.

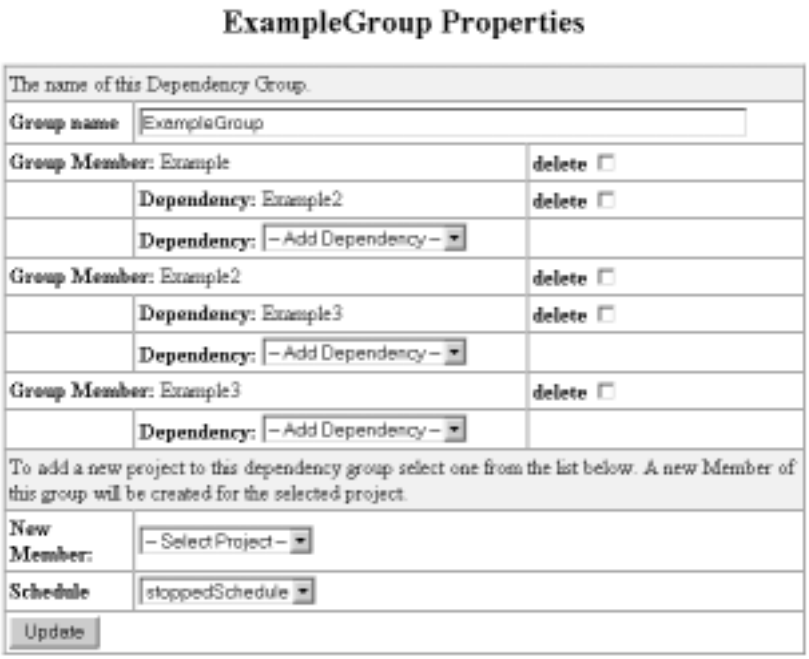
Reopen the group and select the second member from the drop-down list.

Repeat this process for as many members as you need for each group.

Dependency Drop-Down

After the members of a group have been specified, you may then choose the relationships of each member. Once a project is assigned as a dependency of another project, it will disappear from the drop-down list.

Schedule



Select the schedule to use for builds of your dependency group. All schedules currently created will be in the drop-down menu.

Anthill will build each project in the group on this schedule. This will not affect the pre-existing build schedules for each project.

CHAPTER 9:

SCHEDULES

The Schedules box keeps track of the schedules that Anthill uses to automatically conduct builds of Ant projects.

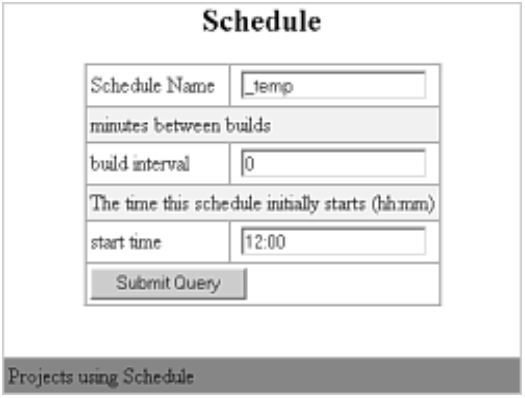
Users may create customized schedules for each project, or use the same schedules for many projects.

Upon install, Anthill will have two schedules created:

- stoppedSchedule,
- default.

Create a new schedule

To create a new schedule, click the "Create New Schedule" link at the bottom of the schedule box. This will open a properties screen that will allow you to name your schedule and specify the start time and the interval at which projects using this build schedule will use.



Schedule Name

Name your schedule in this box. Do not use spaces.

Build Interval

Enter the number of minutes between each build.

Start Time

Enter the time to begin this build schedule.

This will be the start time when Anthill is started, or restarted. After the initial time, Schedules will only run at the specified build interval.

The exact start time will be the time of the computer Tomcat is running on.

stoppedSchedule

Anthill is equipped with a stoppedSchedule to be used if you need to stop builds for a period of time. Because Anthill's very purpose is to perform builds on a regular basis, all projects must select a schedule for builds. This schedule was created as a workaround for situations where it might be wise to delay builds temporarily.

APPENDICES

Appendix A: Troubleshooting

Permission Problems:

Installing Tomcat through RPM

If you are installing Tomcat through via RPM, make sure the owner of Tomcat can write to Tomcat home directory and read/write to the Anthill home directory recursively.

The user running Tomcat needs to have write permission to its own home directory, because it needs to write the anthill.properties file.

View the bug log for further information.

Configuration issues:

More Robust Logging Output

To view more commands in your logging output, you can change the logging level in the `log4j`.properties file in the conf directory.

If you set the log4j.configuration in the Anthill conf directory to `log4j.rootCategory=DEBUG` you will a more robust logging output.

There is one caveat with the `DEBUG` log level -- the work directory to which Anthill checks out the project does not get deleted after the build. So it's probably not a good idea to keep logging at the `DEBUG` level in the long run.

Appendix B: Technical Support

Anthill Newsgroup and Mailing List

Urbancode encourages our users to contribute their thoughts on Anthill with the rest of our community of users with our Anthill newsgroup

For updates on Anthill, you may also join the Anthill mailing list

Contacting Urbancode

Feel free to contact the developers of Anthill at Urbancode Software development for more information about Anthill, or to voice your opinion.

Urbancode, Inc.

2044 Euclid Avenue, Suite 600

Cleveland, OH 44115

voice:(216)858.9000

toll free:(878)858.4599

fax:(216)858.9602

email:cindy@urbancode.com

Appendix C: JavaDoc and HTML

Anthill is available coded in JavaDoc and JavaToHTML on the Urbancode website

Appendix D: Anthill Directories

Anthill creates the following sub-directories after an install:

- build
- doc

- dist
- example
- lib
- projects
- publishDir

build

The build folder contains files necessary to build Anthill itself. These build files are not used for your individual projects.

conf

The conf directory contains the configuration files and registry files, as well as the Repository and Version Adapters that are packaged with Anthill.

dist

The dist directory contains the `anthill.war` file. This file should be copied into a sub-directory of your servlet container. See install instructions for more information.

doc

The doc directory contains the documentation packaged with Anthill.

example

The example directory contains all the Example projects that are included with Anthill.

lib

The lib directory contains `.jar` files that Anthill uses.

The lib directories also contain libraries from Ant 1.3 and Ant 1.4, that include Ant `.jar` used by Anthill. To add anything to the ANT classpath, the `.jar` file must be placed in the appropriate Ant directory.

projects

When you create a project or dependency group using Anthill it writes out a project configuration file in the project directory.

publishDir

The `publishDir` is the default location Anthill sends artifacts to for published project.

source

If you downloaded the Anthill version with the source files, this directory will contain the source code.

This directory is not created with an install of Anthill without the source.

Appendix E: Using Anthill with different SCM Software

This section provides notes that only apply to Anthill users who use a specific SCM Software. Most things mentioned in this section are just little warnings, or hints that will help you use Anthill better with your SCM software.

CVS Notes

Make sure the CVS user whose account Anthill will use has logged in to CVS from the machine where Anthill is installed.

If you're running Tomcat as a Windows service, the windows user account being used must have a Windows home directory. This is because CVS looks for the user's home directory to put the `.cvspass` file into.

You might want to create user account dedicated to Tomcat with limited privileges.

Perforce Notes

Perforce has a view that determines where files are checked out to. Anthill does not override the values in the client specification.

To view the client specification, type `p4 client -o`, the `-o` option prints the information to the screen, rather than bringing up the text editor. The client root is used as the local project directory in this case.

Unix Only You must start up the Perforce server with the `P4ROOT` and `P4PORT`.

Use `p4 set` to view current environment variables.

PVCS Notes

Anthill's work directory overrides the work directory specified by PVCS.

Use `pcli GetArchiveLocation` to view the absolute path to the archive, or if you are using the windows GUI of PVCS Version Manager, you can right click on the database of interest, and select properties.

The database location can be found by using:

- `pcli –ppProjectName ListProjectDB`

Which will list the database location of the project specified with the –pp option. Or if you are using the windows GUI then you can view the database location by clicking on the database of interest, and looking at the path in parenthesis next to the database name.

Unix Only If you are having trouble with Anthill finding the commands it is trying execute, make sure you have sourced vmprofile. See PVCS documentation for more details.

For Linux, it is found under the PVCS home directory `vm/linux/bin`, then run:

- `source vmprofile`

Visual Source Safe Notes

Anthill sets the current project to `repository.vss.module`.

`repository.vss.root` is the directory where `srcsafe.ini` is located. This is also the value you would use as the `ssdir` environment variable.

MKS Source Integrity Notes

Make sure your working directory and archive directory are not the same.

Make sure the mksnt directory is in your classpath.