FINALBUILDER Server

FinalBuilder Server

Table of Contents

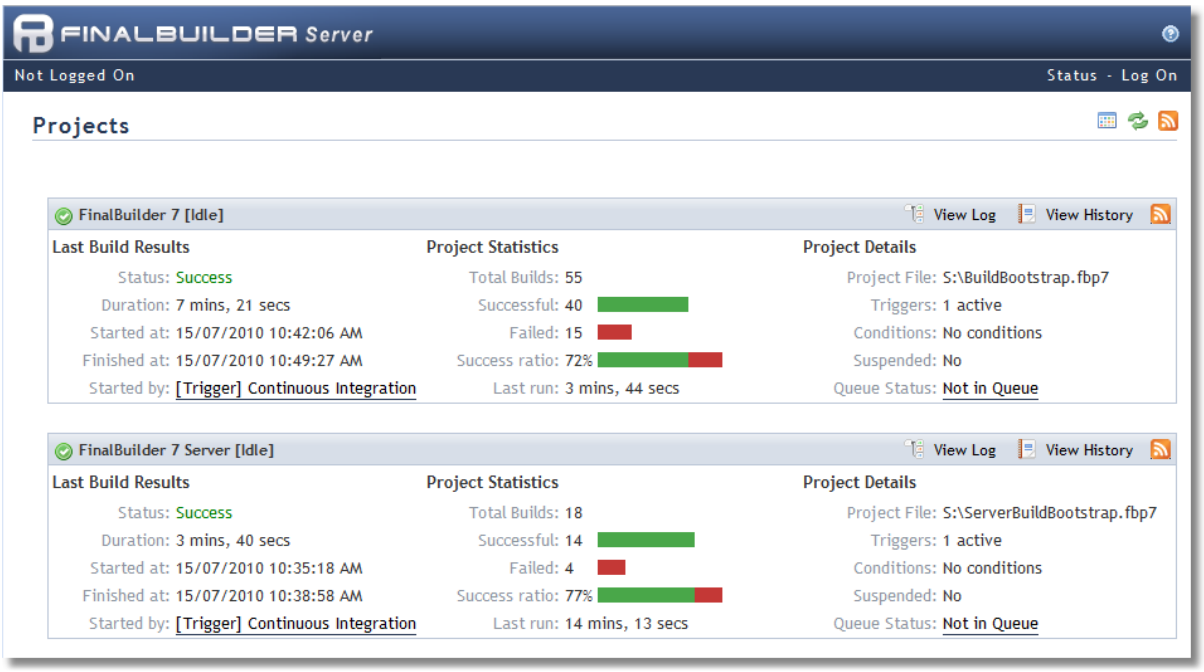
Part I

FinalBuilder Server

1 FinalBuilder Server

1.1 FinalBuilder Server Overview

FinalBuilder Server 7



What is FinalBuilder Server?

FinalBuilder Server centralizes build projects through a web interface, allowing software developers and members of development teams to easily monitor and control all of their builds.

FinalBuilder Server combines with FinalBuilder 7 Professional to allow users to manage multiple build projects through an easy to use interface. Its functionality centralizes build control by allowing authorized users to start, stop and schedule any build, while allowing anyone with a web browser to view build logs and statistics.

1.2 Installation

There are three components which can be installed during the FinalBuilder Server installation.

Build Server

This will install the Build Service, Logging Service and the web based interface to FinalBuilder Server. These components are responsible for controlling the logging and the building of your FinalBuilder build scripts, as well as providing the graphical interface to FinalBuilder Server.

Management Server
This will install the Management Service which will be used by one or more build servers to authenticate and authorize users, provide a central storage location for user profiles and to manage the licenses of each build server.

Help Files
This will install the help files for FinalBuilder Server.

1.2.1 Requirements

Operating Systems Supported

- Windows 2000 SP3+
- Windows 2003 Server
- Windows 2008 Server (Excluding Windows Server 2008 Core)
- Windows Vista (Business, Enterprise and Ultimate only)
- Windows XP Professional SP1+
- Windows XP Professional x64 Edition

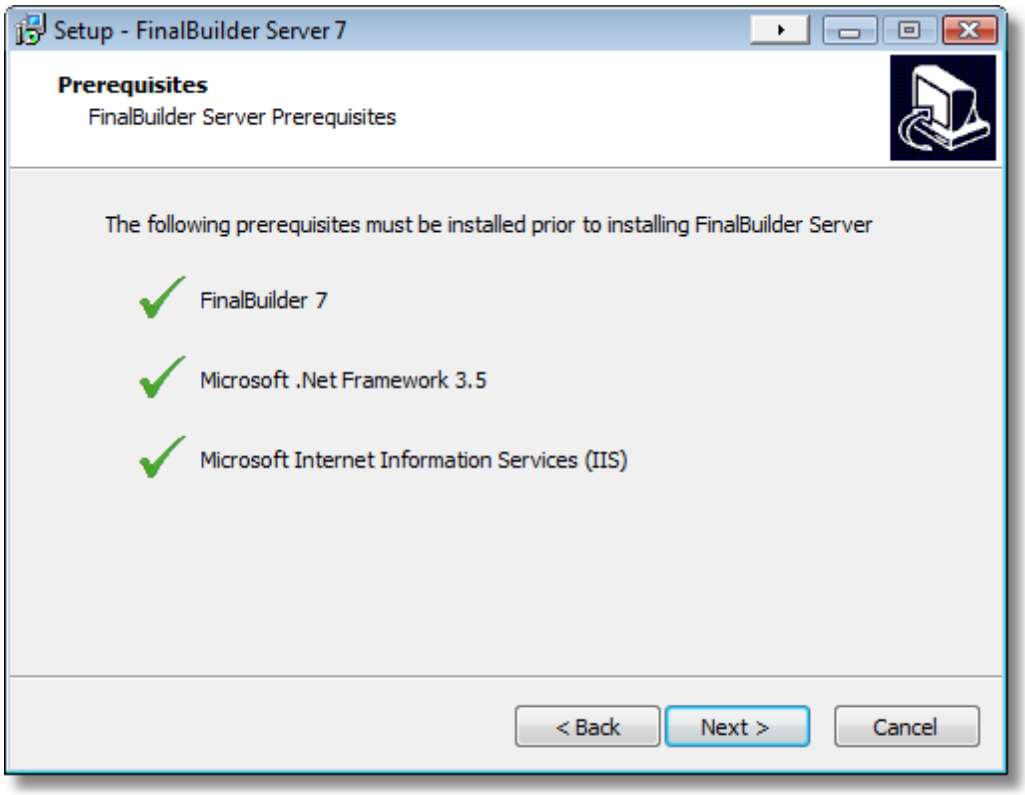
Additional Software Requirements

- FinalBuilder 7 Professional. [*]
- Microsoft Internet Information Services 5.0 or greater. [*]
- Microsoft .Net Framework 3.5

[*] - These components are only required when you are installing the FinalBuilder Server Build Server.

1.2.2 Installing FinalBuilder Server

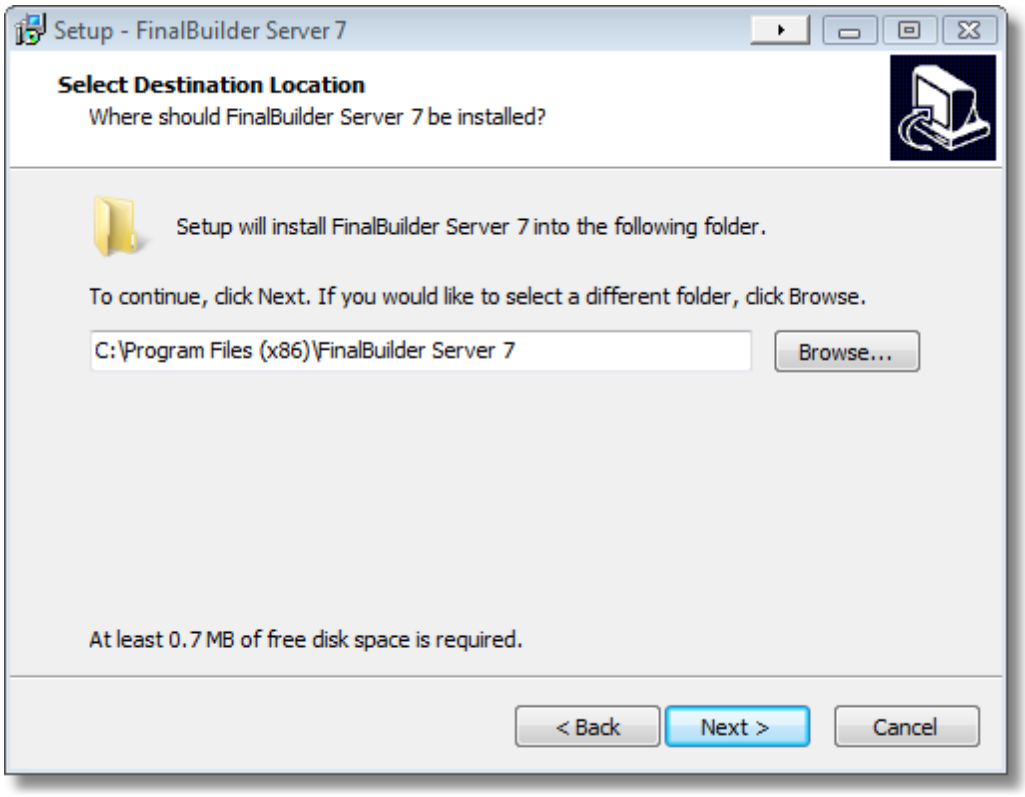
The FinalBuilder Server installer has been simplified by separating the installation of the program files and the configuration of the web interface. For more information on the post-installation configuration see Post Installation Configuration. When you are upgrading from a previous version of FinalBuilder Server and you are using a remote management server, you must upgrade the management server as well.



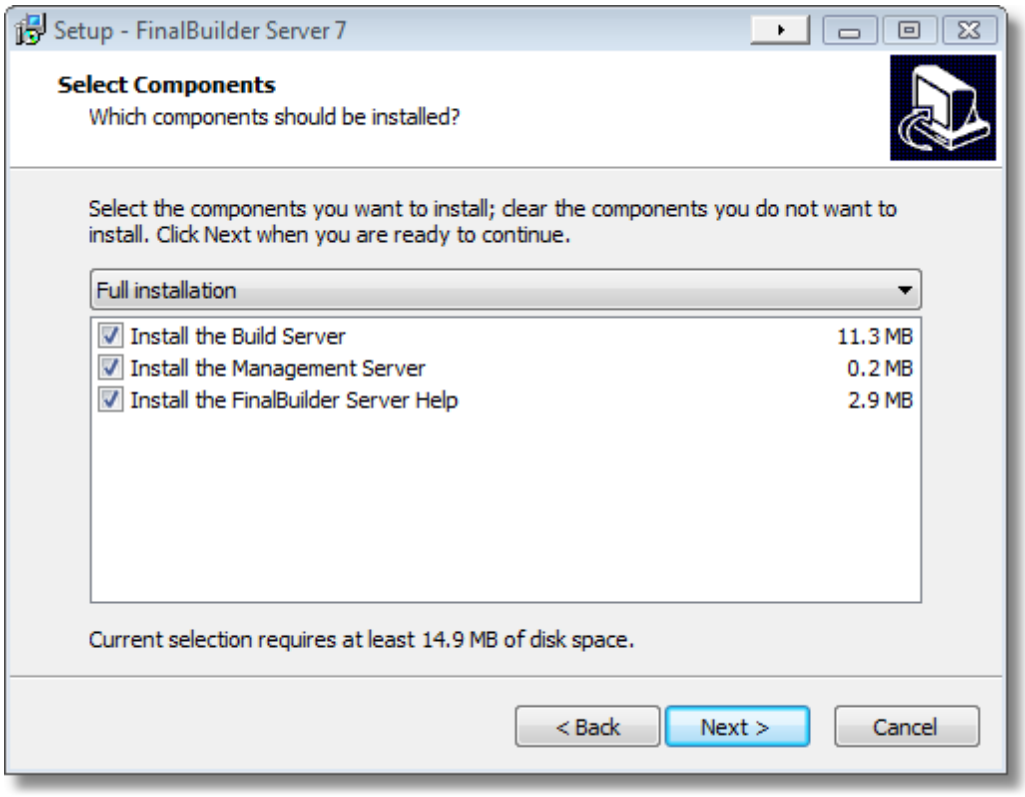
Prerequisites
FinalBuilder Server requires that all the applications that it depends on are installed before installation can commence.

The prerequisites are:

- FinalBuilder 7 - Download from http://www.finalbuilder.com/
- Microsoft .Net Framework 3.5 - Download from http://www.microsoft.com/
- Microsoft Internet Information Services (IIS) - Must be installed as part of the operating system.

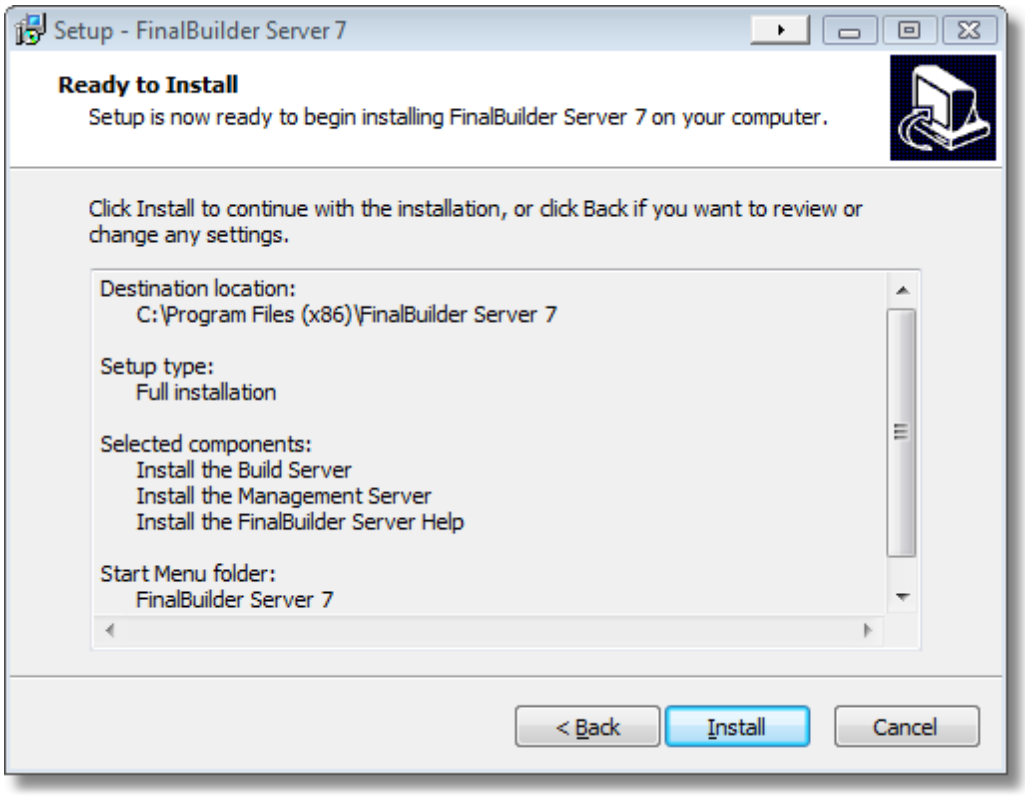


Destination Location
You may choose the location where you wish the FinalBuilder Server program files to be located. It's recommended that you do not install over the top of a previous major version (i.e. Do not use the same directory as FinalBuilder Server 5 for FinalBuilder Server 6).



Components
FinalBuilder Server allows you to choose which components are installed on the current computer. The components are:

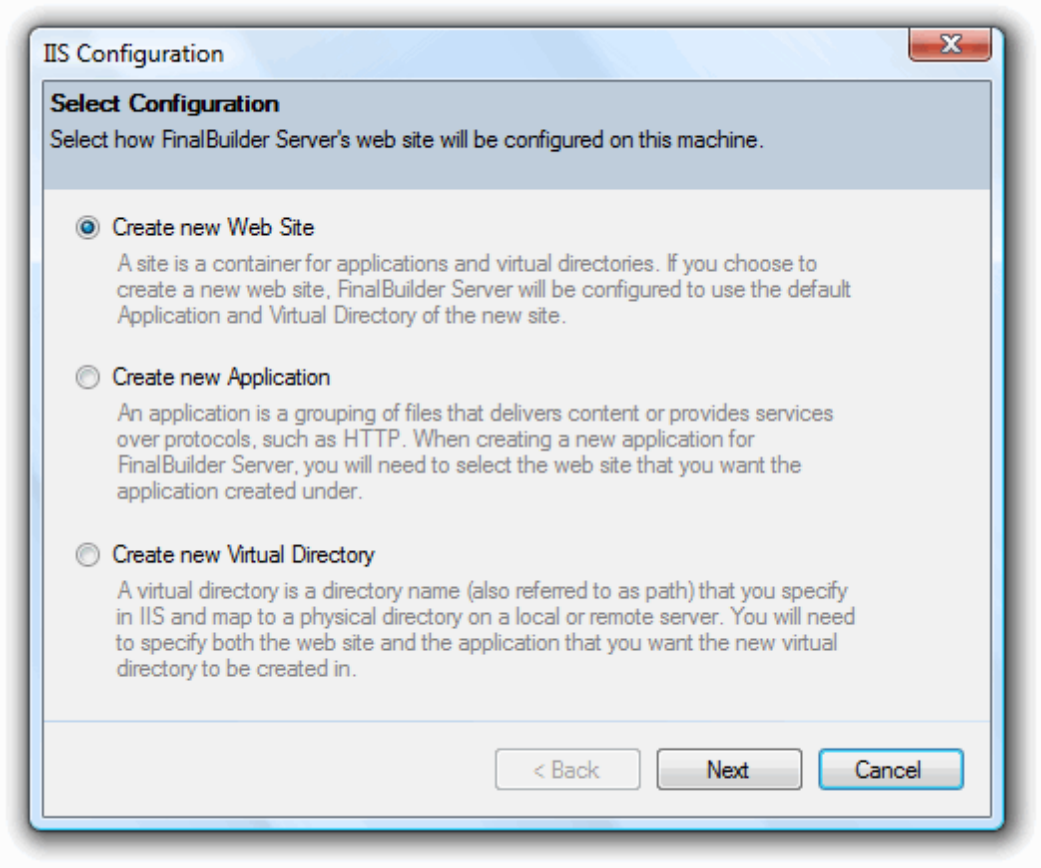
- Build Server - The FinalBuilder Server build server is the central service for managing projects, and controlling your builds. This component also installs the web interface and configuration utility.
- Management Server - The management server controls licensing, user roles and permissions and profiles for all build servers that are connected to it. You do not need to install the management server if you will be using an existing management server located on a different machine.
- Help - Installs the help document that you are currently reading in HTML Help format.



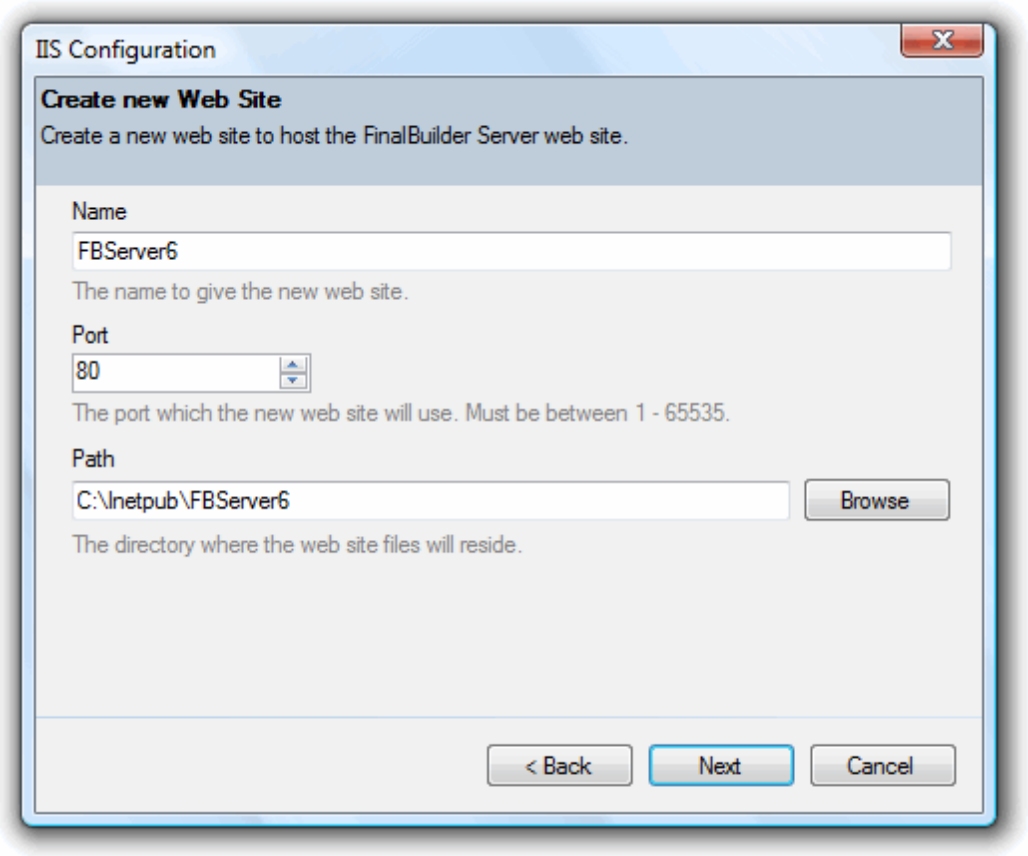
Ready to Install
Clicking 'Install' will copy the files to the specified destination folder and when finished will launch the configuration utility.

1.2.3 Post Installation Configuration

The FinalBuilder Server configuration utility will configure Internet Information Services (IIS) on the current machine to host the web interface. The configuration wizard may appear different depending on which version of Windows your are installing FinalBuilder Server on to (for example, the version of IIS that ships with Windows XP Professional does not include support for multiple web sites).

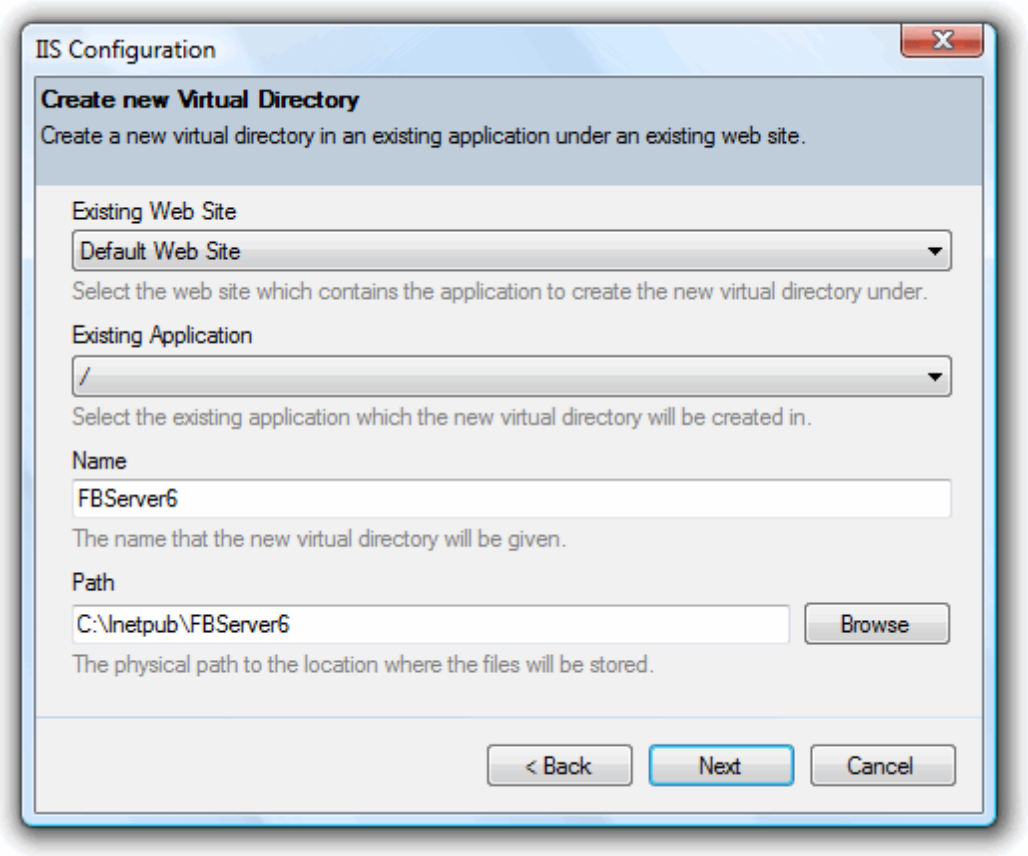


Select Configuration
The configuration that you can choose select depends on which version of Windows you are installing FinalBuilder Server on to and how you would like IIS to be configured. If you are running Windows XP Professional or Windows 2000 Professional then you will not have any other option then to create a new virtual directory (limitation imposed by the operation system).



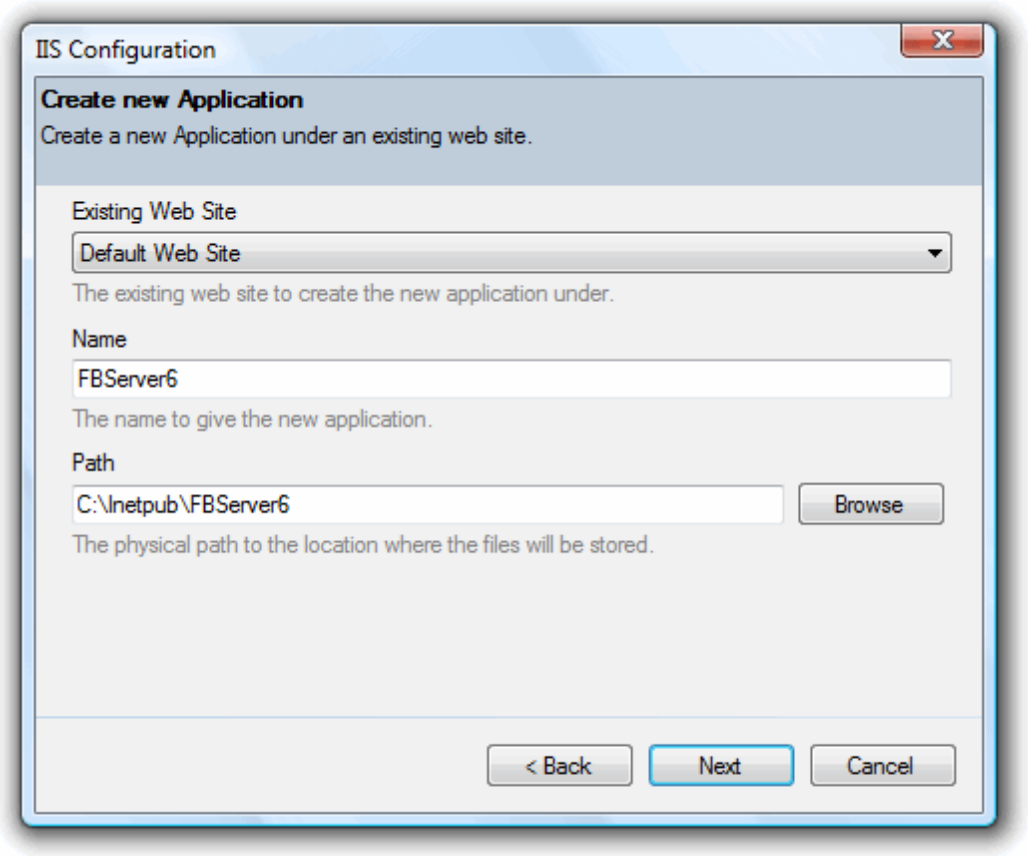
Create a new Web Site

Name - This is the friendly name for the web site.
Port - The port which the FinalBuilder Server web interface will run on. Depending on the configuration of the machine you may need to use a port other then 80 as it may already be in use.
Path - This is the location of the directory where the web site files will be extracted too. You will need to make sure that the IIS Worker Process has the necessary privileges on the directory.



Create a new Virtual Directory

Name - This is the name of the virtual directory, the name specified here will be used to access the web interface (I.E. http://localhost/[Virtual Directory Name]/Default.aspx).
Existing Application (IIS 7 Only) - The application which the virtual directory will be created under.
Path - This is the location of the directory where the web site files will be extracted too. You will need to make sure that the IIS Worker Process has the necessary privileges on the directory.

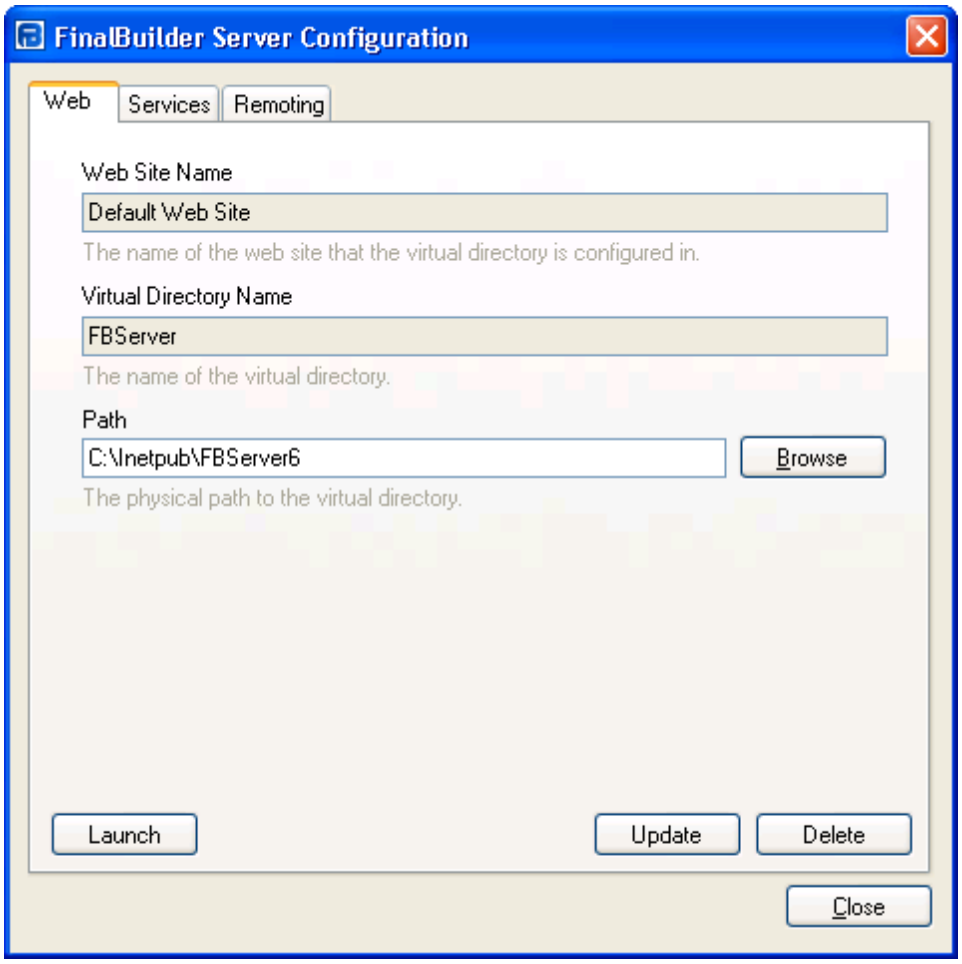


Create a new Application (Internet Information Server 7 only)

Name - The name of the new application.
Path - This is the location of the directory where the web site files will be extracted too. You will need to make sure that the IIS Worker Process has the necessary privileges on the directory.

1.2.4 Configuration and Maintenance

To reconfigure the website, perform maintenance tasks on the FinalBuilder Server services, and change management server location, use the FinalBuilder Server Configuration application accessed via Start | Program Files | FinalBuilder Server 7.

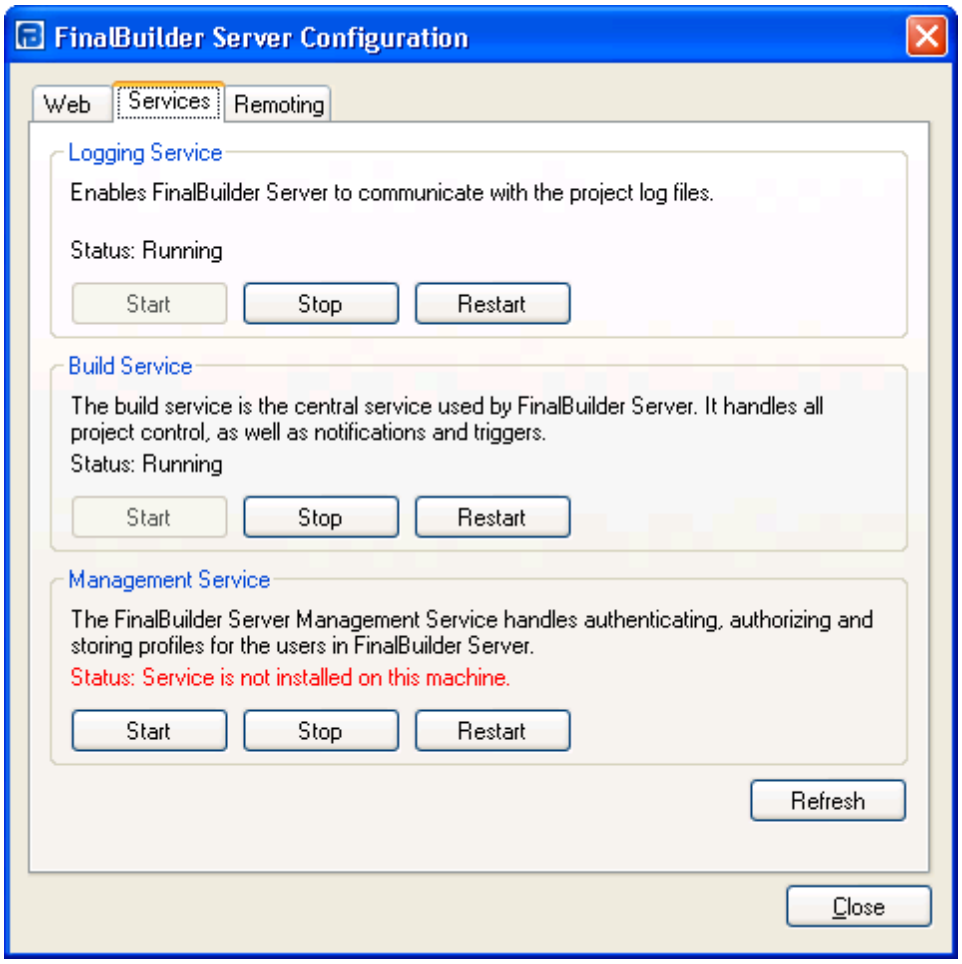


Web Site Name - read only, displays the Web Site Name as configured in IIS

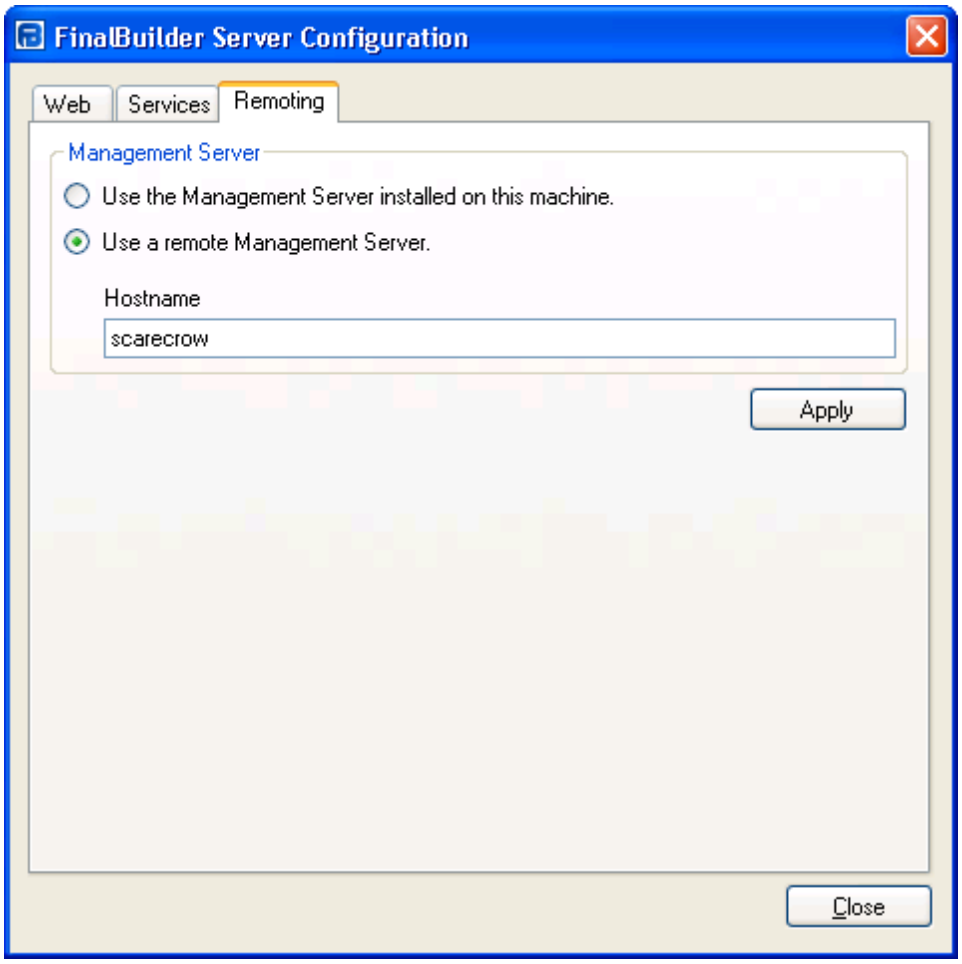
Virtual Directory Name - read only, displays the Virtual Directory as configured in IIS

Path - the physical path to the virtual directory. This property may be changed if required, and then use the Update button to apply the changes.

Launch Button - will start your default browser with your FinalBuilder Server website.



Use the Services tab to view the status of the Logging, Build and Management services, and also stop, start and restart the services if required. Note that if you are using a Management Service on a different machine, then the Management Service section on this machine should be stopped (or not installed).

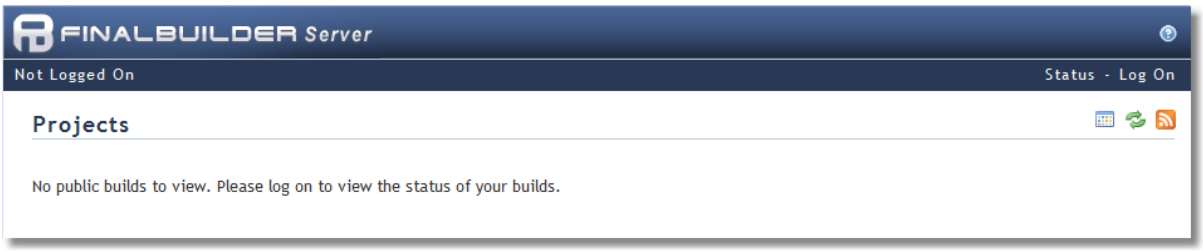


The Remoting tab displays the location of the Management Service and allows you to change the Management Service location. Make sure you Apply the changes before closing the dialog.

1.3 Getting Started

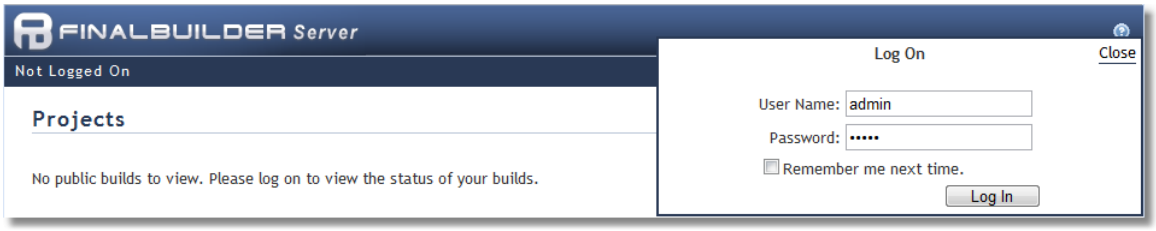
Once FinalBuilder Server has been installed and the post-installation configuration has been completed, you may navigate to the FinalBuilder Server's web interface to begin setting up your projects.

If you are unsure of the address of the web interface, launch the FinalBuilder Server Configuration utility and click the 'Launch' button on the Web tab page. This will launch the FinalBuilder Server web interface in your default browser.



1.3.1 Logging In

After the installation of FinalBuilder Server has been been completed you will need to login to the web interface.

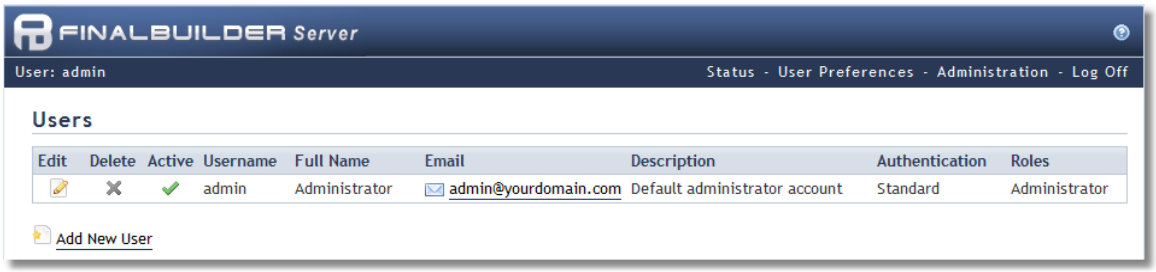


Logging in

1. Click the 'Log On' link positioned at the top-right of the page.
2. If this is the first time the management service has been used with any build servers, the default user will be active. The default username is 'admin' and the default user's password is 'admin'.
3. Once you have logged in successfully you will notice that the 'Administration' menu item is now available, and you can add new projects to the build server. It is recommended that before you begin configuring your build projects, you should create a new user and remove or de-active the default administrator user account.

1.3.2 Creating New Users

Once you have logged into the build server for the first time, it is recommend that you create a new user account and remove or de-active the default administrator user account.



Creating new users

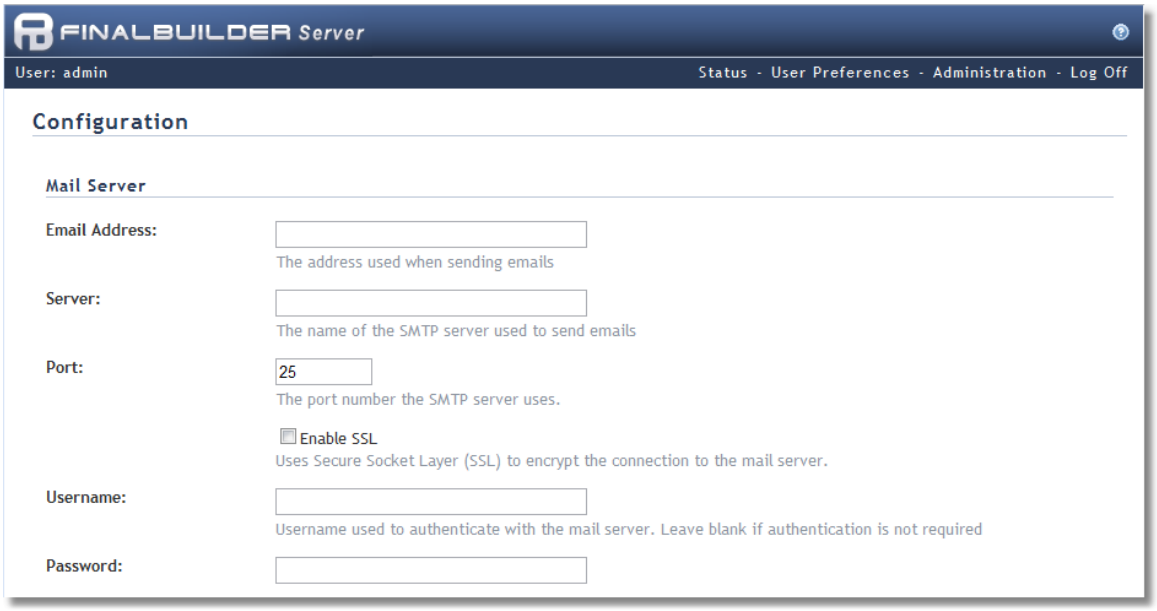
1. To create a new user you will need to be logged on to the build sever.
2. Click the 'Administration' link from the top menu bar.
3. From the 'Administration' page, scroll down to the 'Management Server

Administration' section and click the 'Manage Users' link.
4. From the 'Users' page, click on the 'Add New User' link.
5. Choose the type of **authentication** that will be used to authenticate the new user.
6. Specify the **username** that the new user will use to log into the build server.
7. Specify the user's **name**, **email**, and **password**.
8. Optionally specify a **description** that will be associated with this user account as well as a **phone number** that the user can be contacted on.
9. If you require the new user to be assigned additional roles then click on the 'Assign A New Role To User' link.
10. Once you have finished entering the new user's information, click the 'Save' button to add the user.

See also:
Adding a new Standard User | Adding a new Active Directory User

1.3.3 Configuring the Build Server

Before adding your FinalBuilder projects to the build server you should configure the global options.



Configuring the Mail Server

1. Specify the **email address** that will be used as the sender for all emails sent from FinalBuilder Server.
2. Specify the **address** to your mail server (I.E. 'mail.yourdomain.com').
3. If your mail server requires you to authenticate you will need to specify it in the **username** and **password** fields.

Configuring the Web Site Appearance

If you wish to add a **message** to the status page of FinalBuilder Server, you can specify one here. You may use HTML if you require advanced formatting of the message. A message can be useful to identify the build server if you are using more than one, or as a general notification to users.

FinalBuilder Server also allows you to add a **logo** to the heading bar of each page (next to the help button). You must first copy an image to the logo folder located in the images folder of the FinalBuilder Server web site. It's recommended that the logo be a maximum of 28 pixels high (the width is not so important as there is a lot of horizontal space).

Configuring Global Project Settings

When a project is run under FinalBuilder Server, it is required that it is run under a normal Windows user account (rather then the user which the service is running under). You may use a domain user account by specifying <domain>\<user>.

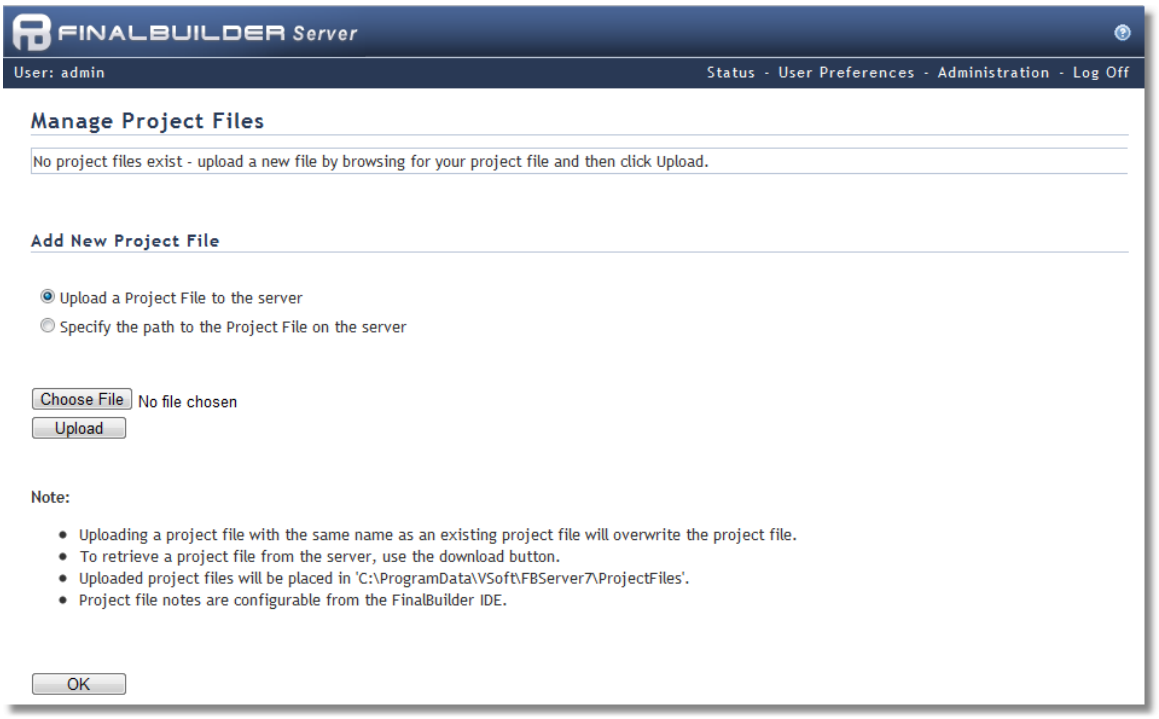
You can choose to specify the user in the global settings which will remove the need to enter it each time a project is configured on the server. If you do decide to globally define a user to impersonate, keep in mind that any user which has permissions to add project files to the server could then run any application as that user. For security reasons it is recommended that the impersonated user does not have administrative privileges on the machine.

You can also select the maximum number of concurrent builds that can be running at any one time by specifying a value in the **'Concurrent Build Limit**' field.

See also:
Configuring a Mail Server | Configuring Web Site Appearance | Configuring Global Project Settings

1.3.4 Uploading a FinalBuilder Project

Now that the build server has been configured you can proceed to add a FinalBuilder project file to the server.



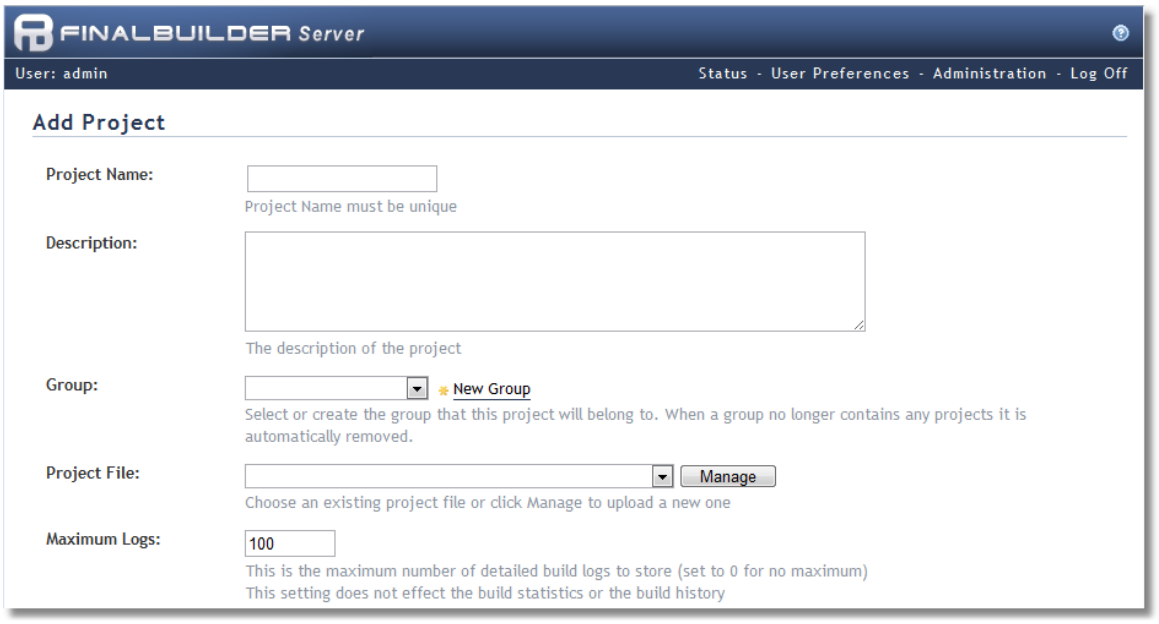
1. From the FinalBuilder Server Status page click on the 'Manage Project Files' link, located towards the bottom of the content.
2. You can choose to either upload the project file to the build server or you can specify a path to a project which already exists on the build machine.

- If you choose to upload the project file, select the path to the file on you local machine by clicking the 'Browse' button and then clicking 'Upload' to upload it to the build machine. The uploaded files are stored in a sub-directory of the FinalBuilder Server data file folder. Note: When a project file is uploaded to the build server, changes made to the original project file are not reflected in the uploaded project file. It is also important that you design your build process so that all file paths will work on both the development machine and the build server. It's useful to use variables for this, and FinalBuilder Server allows you to set the variables when you configure the project, or interactively when the build is started.

- If you would prefer to provide a path to a FinalBuilder project file which is already located on the build machine, click on the 'Specify the path to the Project File on the server' radio button. You can now either browse to the file location, or type it into the 'Selected Path' text field. Once you have chosen the file to use click 'Add File'. Be aware that when you are browsing for the project file, you may not be able to access the entire file system, and in the case where the project file is located in such a location, you will need to specify the path in the 'Selected Path' field. Note: The project file is not copied, so any changes made to the file will be automatically used the next time the build is started.

1.3.5 Adding a New Project

Once the FinalBuilder project file has been uploaded to the build server you can add a new FinalBuilder Server project which uses that project file.

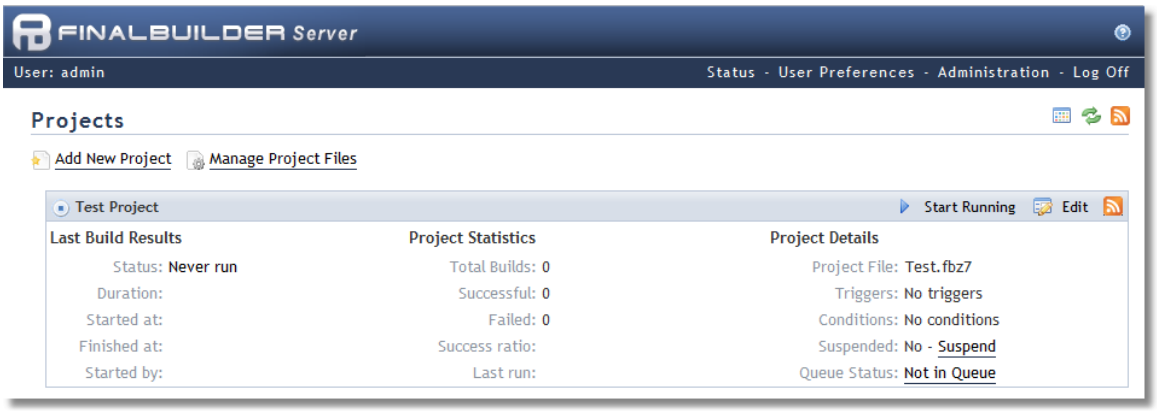


1. From the FinalBuilder Server Status page click on the 'Add New Project' link located towards the bottom of the content.
2. Choose a **project name** which will be used to identify the project throughout the server.
3. Optionally give the project a **description**, the description can be anything (i.e. explains what the project builds, or it might contain the maintainer of the project).
4. If you have not already done so, you will need to upload a FinalBuilder **project file** to the server and then select it from the drop down list.
5. Select whether you want the project to be made **public**. When a project is public, all users including anonymous users can view the project, but only logged in users may interact with the project. A private project is only visible to users who are logged in and have access to it.
6. Depending on whether or not you have a global impersonation user you may need to provide the windows username and password who will be used to execute the build.
7. Clicking 'Add' will add the project to the server and you will be redirected back to the status page after clicking 'Save'.

Note: you can have multiple projects defined which use the same physical project file. Each project can be configured with different variables, different triggers, etc.

1.3.6 Controlling a Project

After you have added the new project to the server you can return to the Status page to start the project.

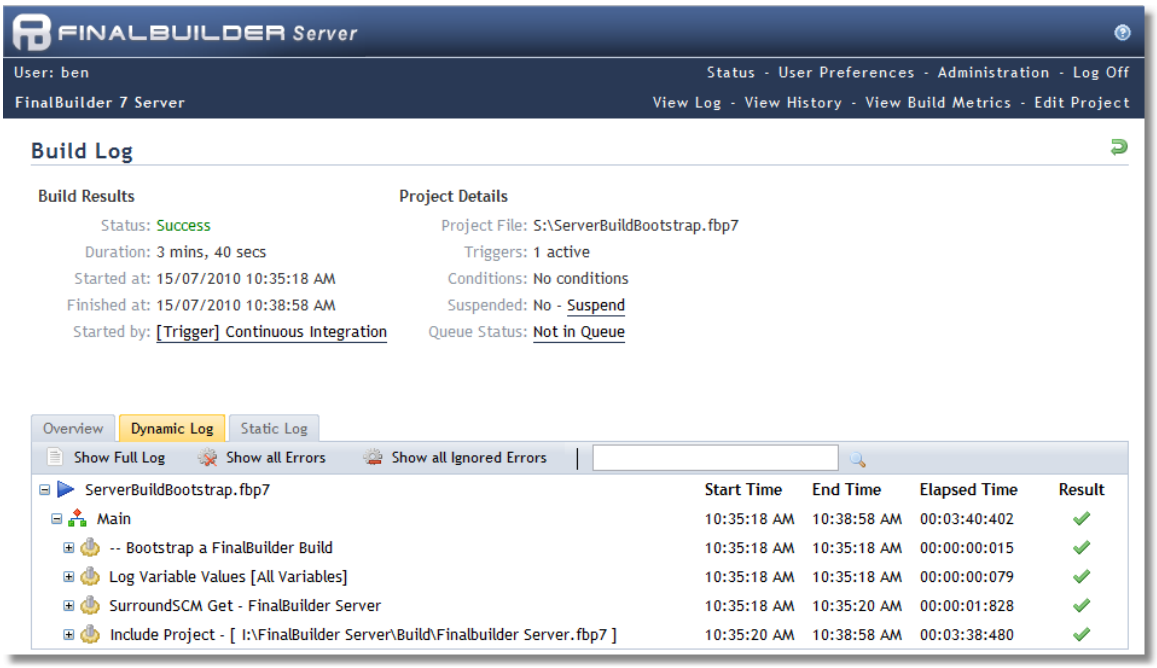


- When a build is started by clicking on the '**Start Running**' button you will be prompted to either start the build normally by placing it on the build queue or by bypassing the build queue and starting it immediately. See Understanding the Build Queue for more information on the build queue.

- When a build is currently running you may can stop it by clicking '**Stop Running**'. If the build has become unresponsive you can force the build to terminate by clicking the same button again, doing so may leave your build logs in an unpredictable state as FinalBuilder cannot cleanly stop the build when it is terminated.

1.3.7 View the Build Logs

FinalBuilder Server allows you to view and search the build logs from any previous that have not already been removed.



Show Full Log - if you have filtered the log, then the log will be reloaded at the root node.

Show all Errors - only show actions that have a status of Error. This is the most useful function to quickly diagnose why a build failed.

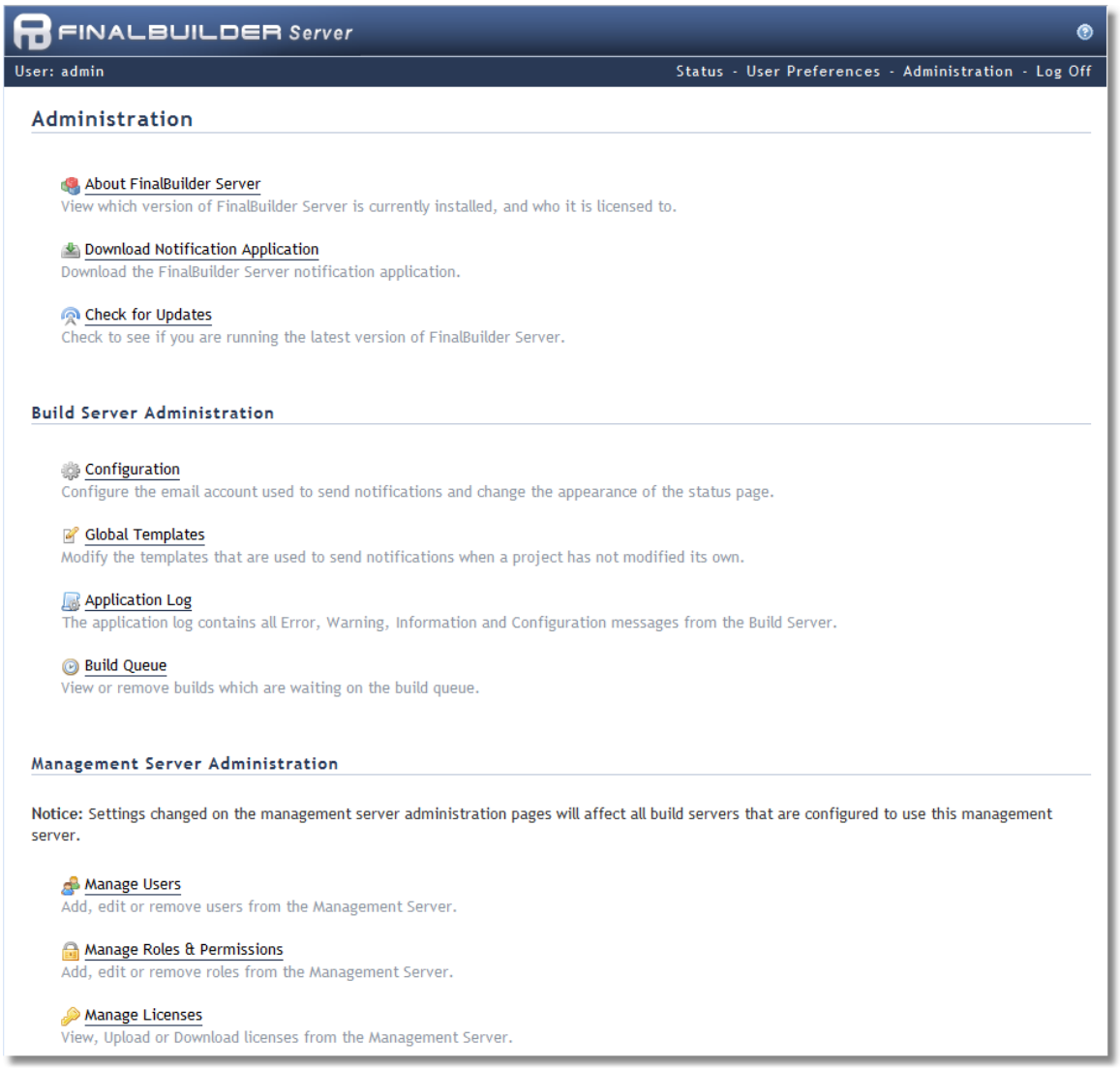
Show all Ignored Errors - if your project includes actions that have "Ignore Failure" turned on, then this filter will allow you to filter by those actions that failed.

Show all Running - not shown on the above screenshot, but if the project is currently running this extra filter will allow you to filter the log by actions that are currently running.

Find - to search for any text in the build log (action description and action log output), enter the text in the text entry field and then click the Find button (looks like a funnel).

1.4 Administration

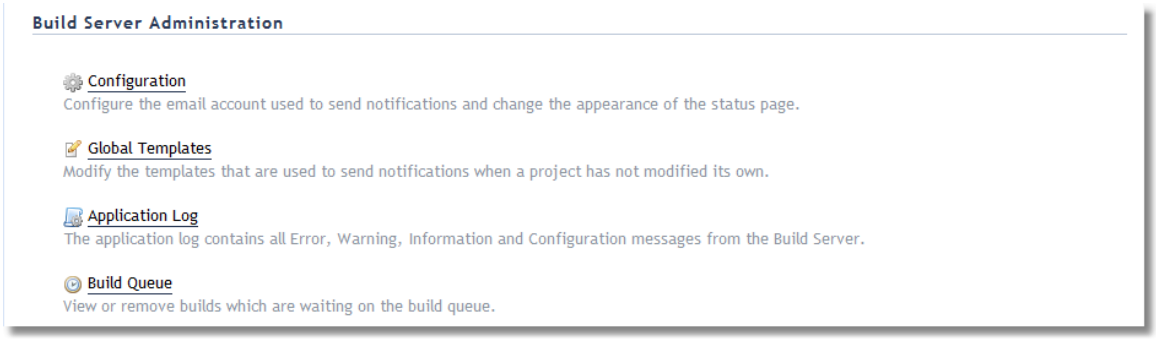
The administration page gives you access to configuring both the current build server, as well as the user and security options for all builds servers connected to the same management service.



Check for Updates
It is recommended that you check for updates frequently to ensure you are running the latest official builds of FinalBuilder Server. This feature reads an XML file from the VSoft Technologies web server and does not transfer any personal information. An internet connection is required on the build server for this to work.

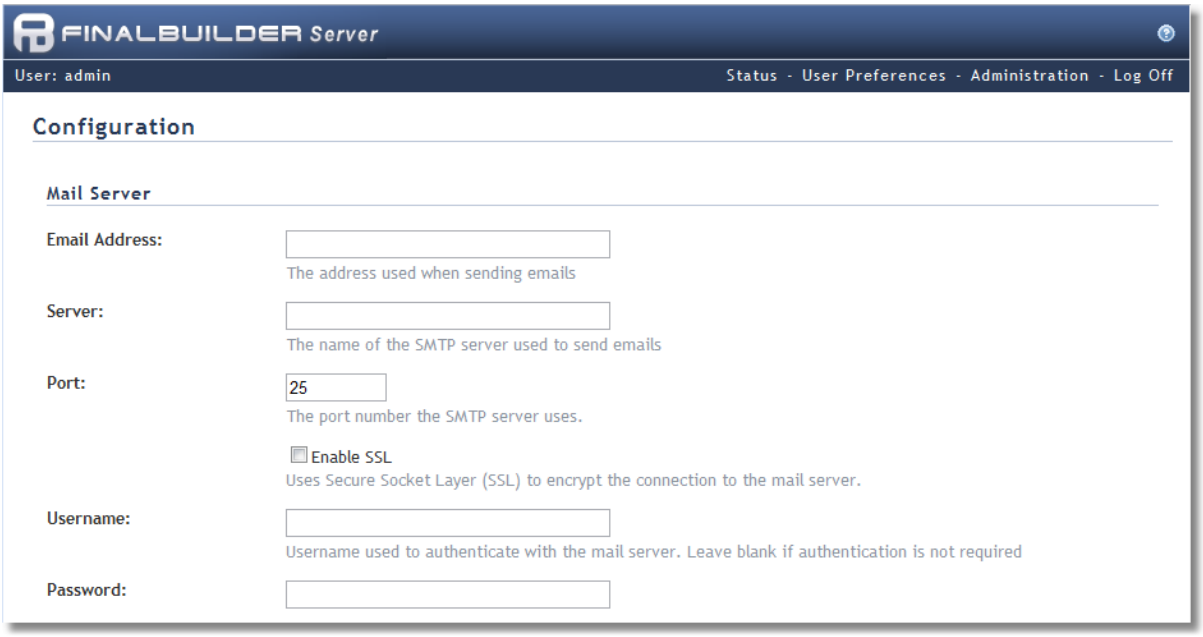
1.4.1 Build Server Configuration

Each FinalBuilder Server build server has a number of settings that apply to only the current build server. Some of these settings include SMTP (for notifications), appearance, global templates and global project settings.



1.4.1.1 Configuring a Mail Server

FinalBuilder Server requires that an outgoing mail server is configured so that build notifications can be sent.



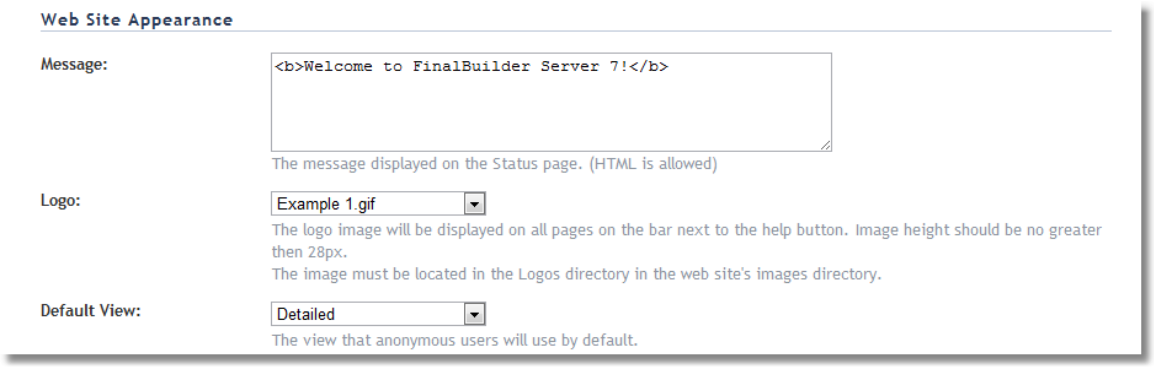
Email Address
This is the email address that FinalBuilder Server uses to send notifications to users. You can specify just an email address or you may wish to include a name as well e.g. 'Build Machine <build@finalbuilder.com>'.

Server
The address of the SMTP server to use to send notifications from FinalBuilder Server.

Username and Password
If the SMTP server requires authentication, then you can provide the credentials here. Talk to your system administrator if you are unsure how to configure the SMTP server settings.

1.4.1.2 Web Site Appearance

FinalBuilder Server allows you to append custom HTML to the top of the status page as well as include a custom logo in the title bar.



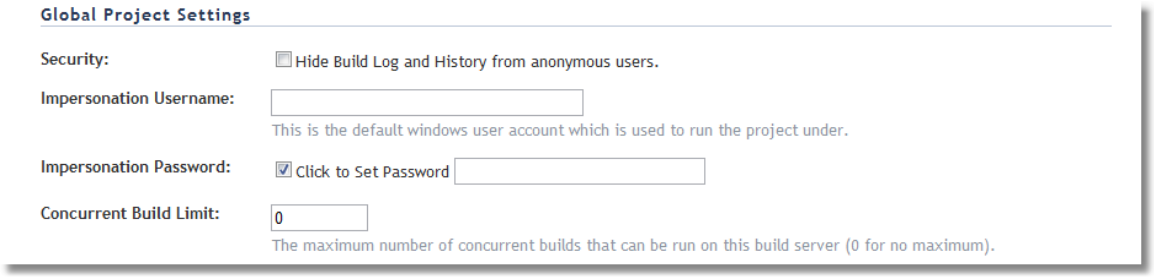
Message
This is the text that will be displayed at the top of the status page. A message can be useful to identify the build server if you are using more than one, or as a general notification to users.

Logo
An image that is displayed at the top of every page next to the help button. You must place the image in the 'Images/Logos' directory, located in the FinalBuilder Server web site installation folder (e.g. C:\InetPub\FBServer6\Images\Logos). It's recommended that the logo be a maximum of 28 pixels high (the width is not so important as there is a lot of horizontal space).

Default View
The default view that users will see. Detailed shows details about each build. List shows a summary.

1.4.1.3 Global Project Settings

Global project settings allow you to configure a default user account which the builds will run under as well as the maximum number of concurrent builds.



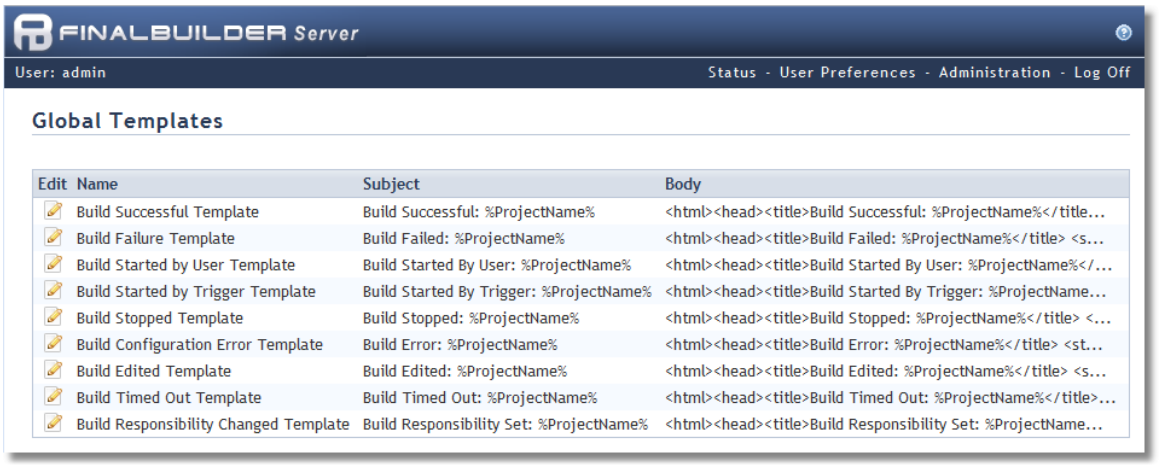
Impersonation Username
The username of the Windows user account that all projects by default will run under. You can specify a domain user as either 'domain\user' or 'user@domain'. You can override this setting on a project by project basis on the Edit Project page.

Concurrent Builds

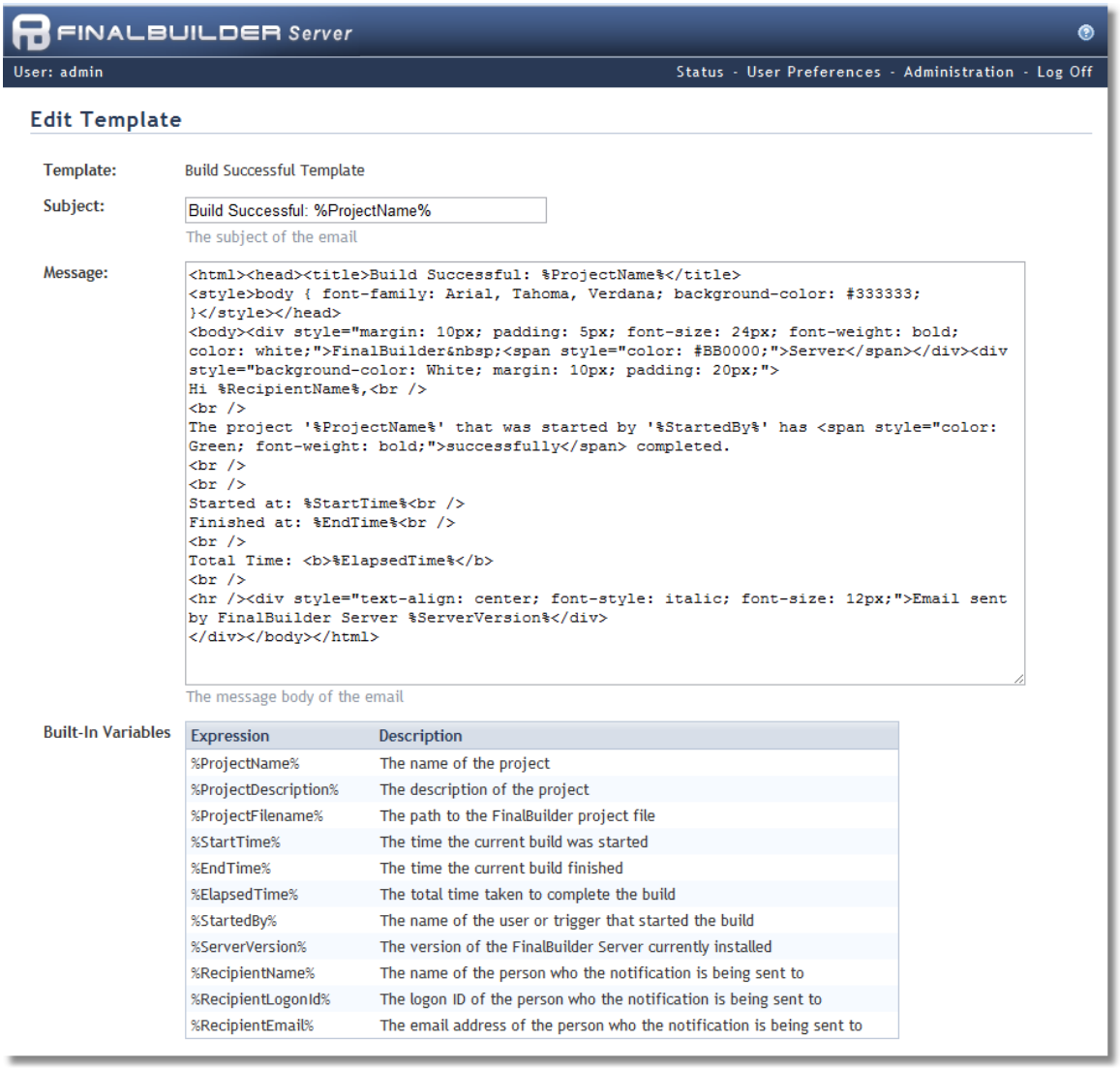
This is the maximum number of builds that will be allowed to run simultaneously on the current build server. If the maximum number of builds has already been reached then the pending builds will continue to wait on the builds queue. Be aware that if a user decides to bypass the build queue when starting a project this option has no effect.

1.4.1.4 Global Templates

FinalBuilder Server allows you to customize the templates that are used when sending build notifications. When you have not changed the default template for a project, FinalBuilder Server will fall back to using the global templates.



Editing a template allows you to change the subject and message. Both the subject and message can contain any of the built in variables that are listed in the Built-In Variables list. Each template can have a different set of variables available.



Subject - the subject of the email that will be sent to subscribers of the notification

Message - enter either an html formatted or plain text message. If the <html> tag is detected, then the mail will be sent formatted as html.

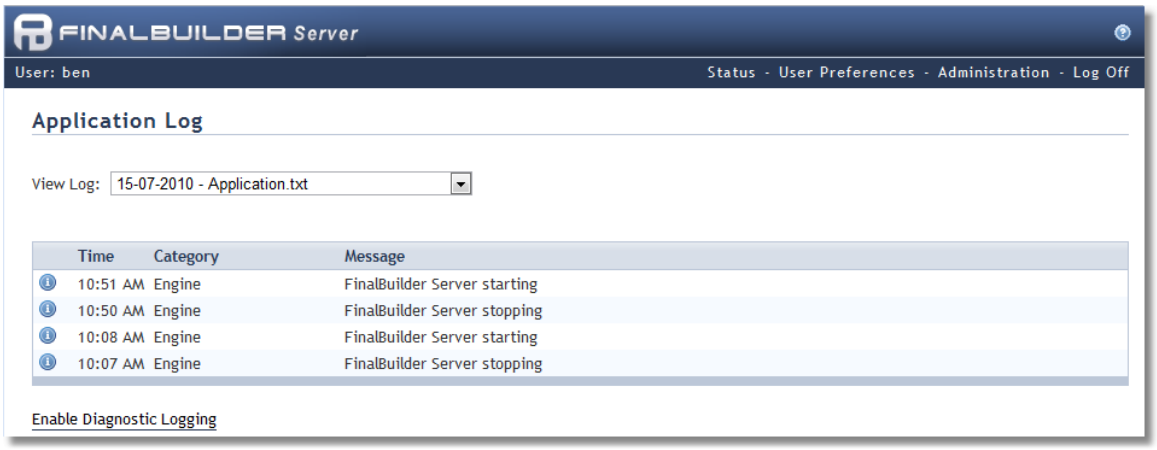
Built-In Variables - a list of the available variables that can be used in both the subject and message. Please note that the variables available vary with each template, for example the **Build Failure Template** defines a %FailureReason% variable.

Reset - the reset button will revert to the default template.

1.4.1.5 Application Log

The application log contains a record of any errors and warnings, as well as other important information that might be useful for tracking down the cause of a problem with the build server. Application logs are persisted to disk and can be found at C:

\ProgramData\VSoft\FBServer7\Logging\Application

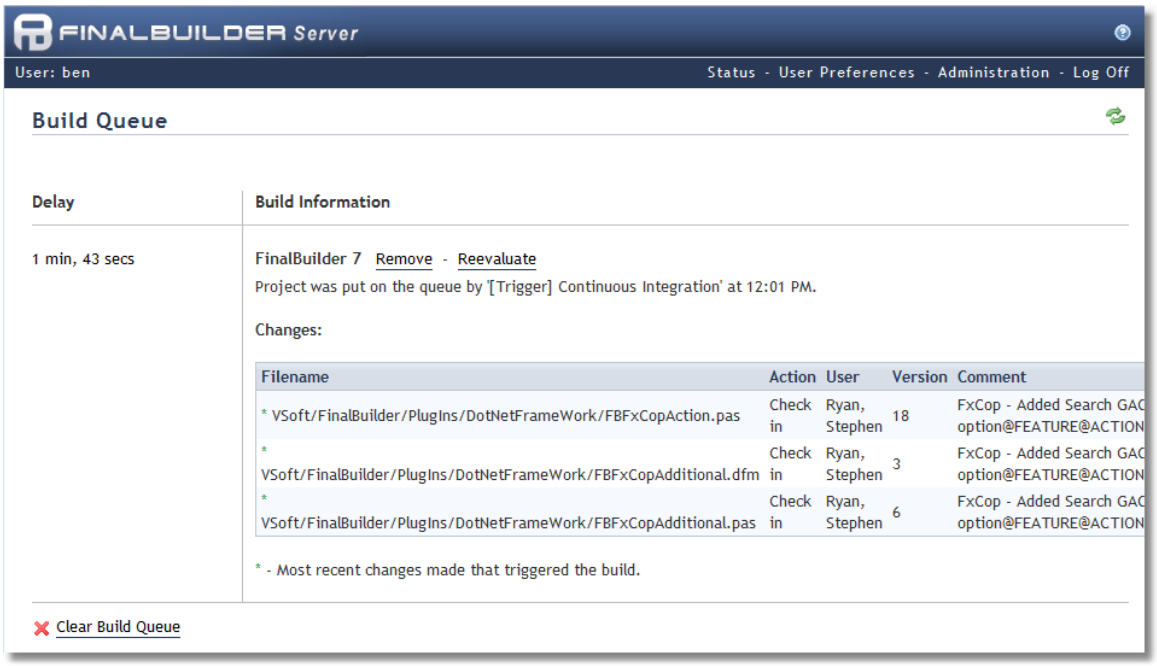


View Log
Choose which log file to view. One log file is created per day.

Enable Diagnostic Logging
Vastly increases the level of logging. Enable this option to help with troubleshooting.

1.4.1.6 Build Queue

FinalBuilder Server runs an internal build queue that is used to control when builds are started. The build queue page allows you to visualize the current state of the queue, as well as removing and re-evaluating build queue items. More in-depth information can be found in the topic Understanding the Build Queue.



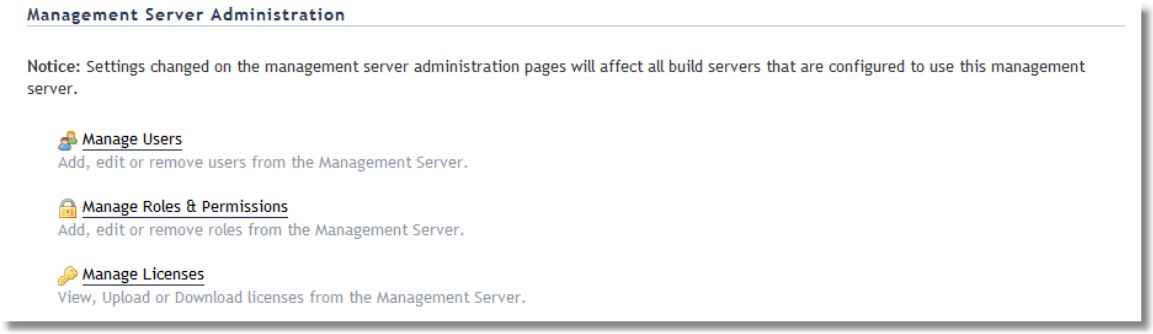
Delay
Each build reports how long it will be delayed until FinalBuilder Server attempts to start the build again. Once the delay has reached zero minutes and zero seconds, FinalBuilder Server will re-evaluate all the build conditions currently active on the project and check to see whether the maximum number of concurrent builds has not been reached.

Reevaluating a Build
When a build is on the build queue and the delay has not reached zero, you may decide to force FinalBuilder Server to reevaluate the build queue item, this has the same affect as the delay reaching zero (conditions are evaluated, and the build will start if all conditions have been met).

To completely and immediately remove this build from the queue, use the **Remove** link.

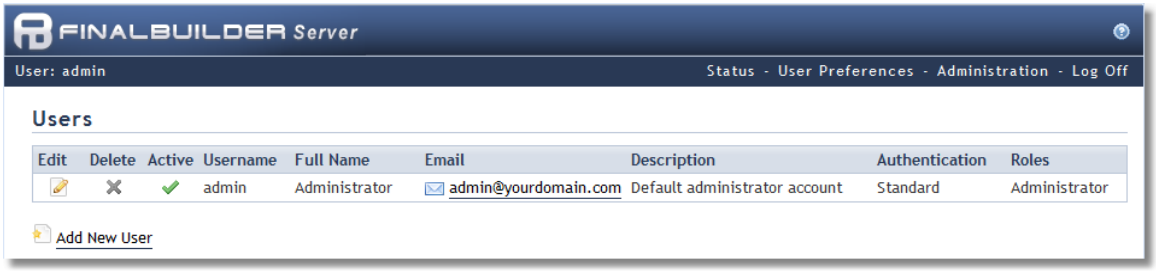
1.4.2 Management Server Configuration

The **Management Server Administration** settings are specific to the management server that the current build server is connected to. When you have multiple build servers connected to a single management service, all builds servers will be affected by any settings that are changed.



1.4.2.1 Managing Users

FinalBuilder Server has support for authenticating users against Active Directory or using its built-in authentication system.



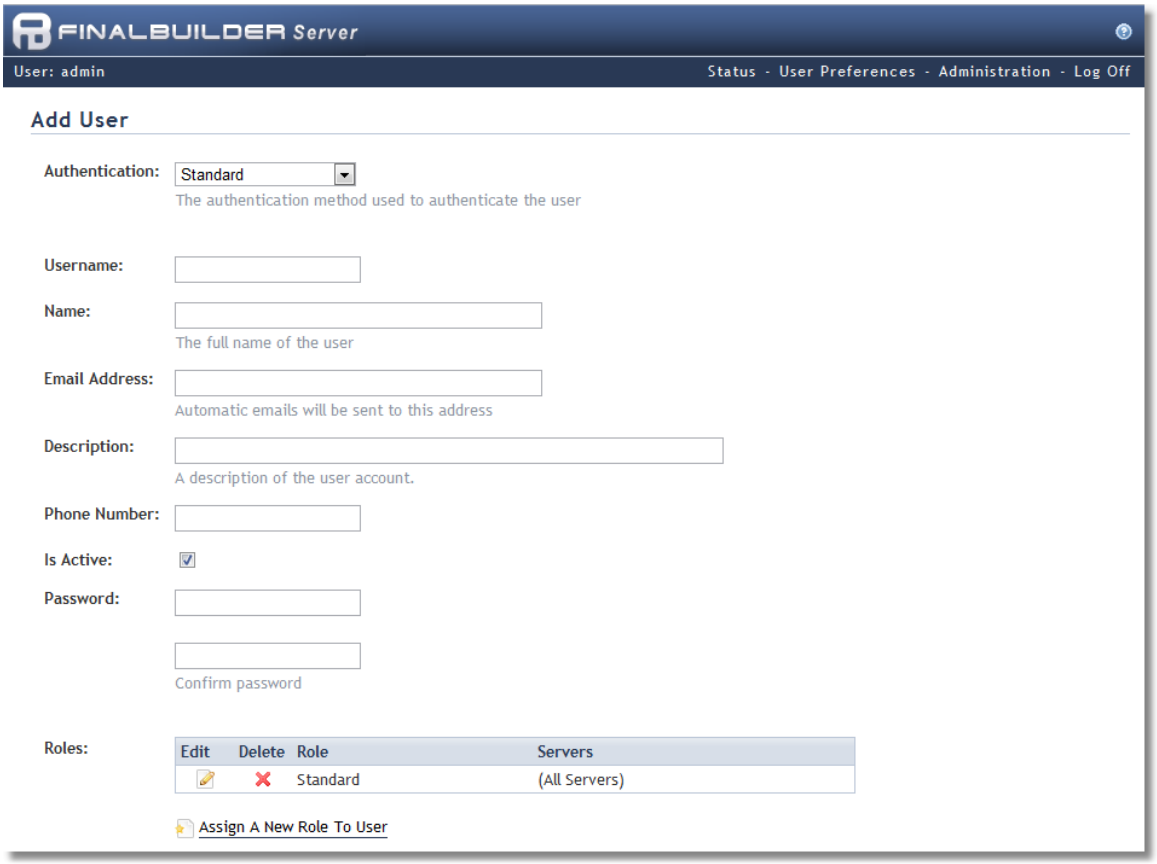
Notes:

- You can only have active the same number of users as you have licenses for.
- Users and user licenses are shared between all build servers that are connected to the same management service. You do not need additional licenses for the same

user on each build server (as long as all build servers are connected to the one management server).
- The email, phone and description is shown when you hover over a users name when it is displayed on the FinalBuilder Server web interface.
- You may mix and match authentication schemes - you can have some users use Active Directory authentication and some users use the built-in authentication system.
- The current usage of user licenses is shown as the bottom note.

1.4.2.1.1 Adding a new Standard User

When adding new standard users you must provide a password which will be used to authenticate the user when they attempt to login.



Authentication
To add a new user using the built-in authentication scheme, choose Standard. To add a user using Active Directory as the authentication scheme, see Adding a new Active Directory User.

Username
The username will be used by the user when they attempt to login.

Name
The full name of user which will be used by the server when sending notifications and listing the user on relevant pages.

Email Address
The user's email address which will used when sending notifications, as well as being displayed on the Manage Users page.

Description
The description can be used to display a note about the user on the Manage Users page.

Phone Number
A phone number that can be used to contact the user. This information is also only shown on the Manage Users page.

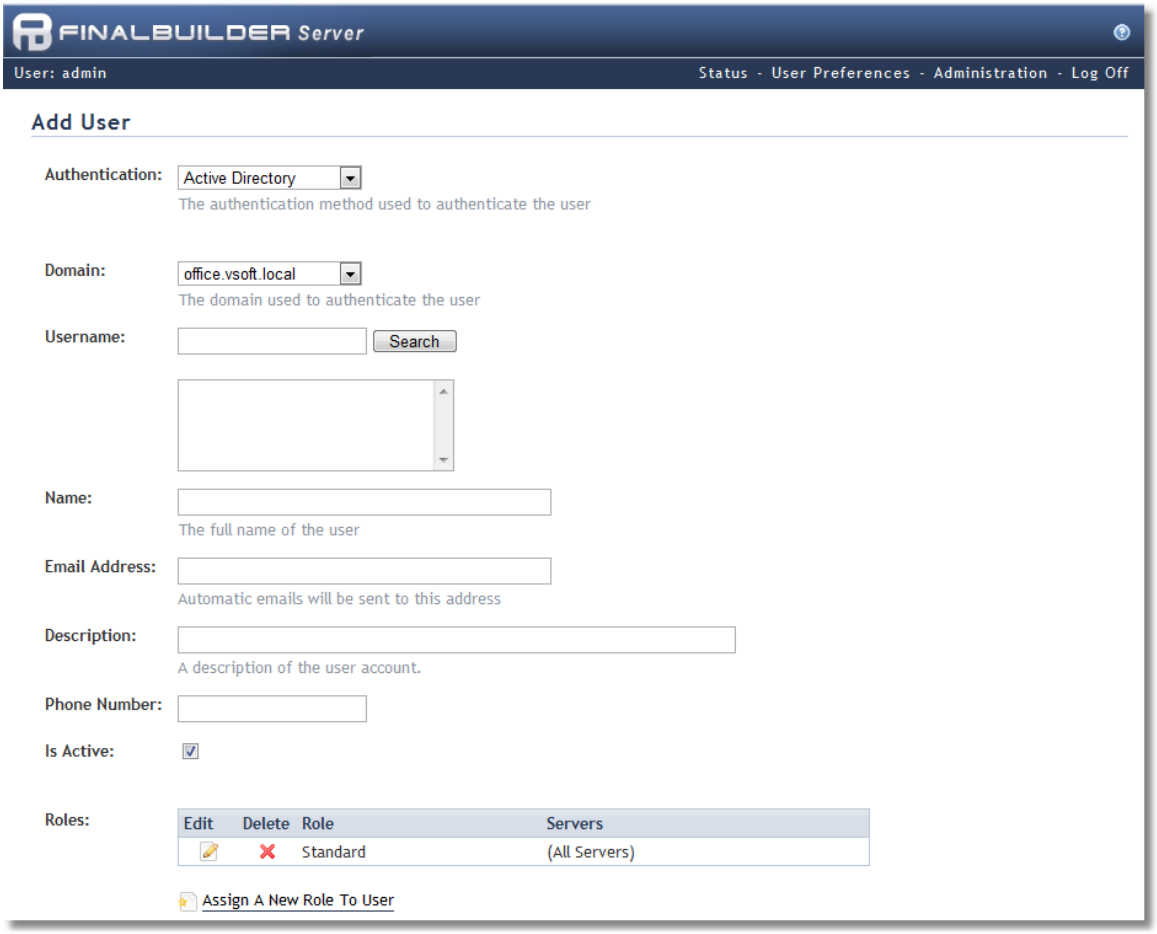
Is Active
When a user is not active they cannot login, or receive notifications. You can still configure the user to receive notifications, they just will not be sent until they are made active. It is advisable to deactivate users and add new users instead of renaming users as this well effect the build history. Only Active users will use up licenses.

Password
The password used to authenticate the user when they attempt to login.

Roles
Lists the roles that the user has on the specified servers. For more information on editing Roles, see Managing Roles & Permissions.

1.4.2.1.2 Adding a new Active Directory User

When adding a new Active Directory user, FinalBuilder Server will attempt to retrieve the Name, Email, Description and Phone Number from Active Directory.



Authentication
To add a new user using Active Directory as the authentication scheme, choose Active Directory. To add a user using the built-in authentication scheme, see Adding a new Standard User.

Domain
This is the F.Q.D.N. of the server which will be used to authenticate the user. This field is automatically populated by FinalBuilder Server.

Username
This is the Active Directory username of the user account to use. This field is automatically populated by FinalBuilder Server. When the user logs in they need to use their Active Directory password in the password field.

Description
The description can be used to display a note about the user on the Manage Users page.

Phone Number
A phone number that can be used to contact the user. This information is also only shown on the Manage Users page.

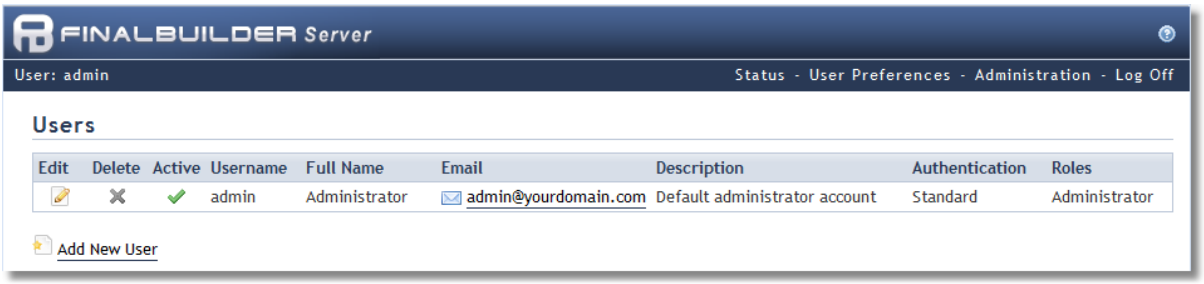
Is Active

When a user is not active they cannot login, or receive notifications. You can still configure the user to receive notifications, they just will not be sent until they are made active. Only Active users will use up licenses.

Roles
Lists the roles that the user has on the specified servers. For more information on editing Roles, see Managing Roles & Permissions.

1.4.2.1.3 Editing Existing Users

You can change a users account information by clicking the edit icon for the relevant user on the manage users page.

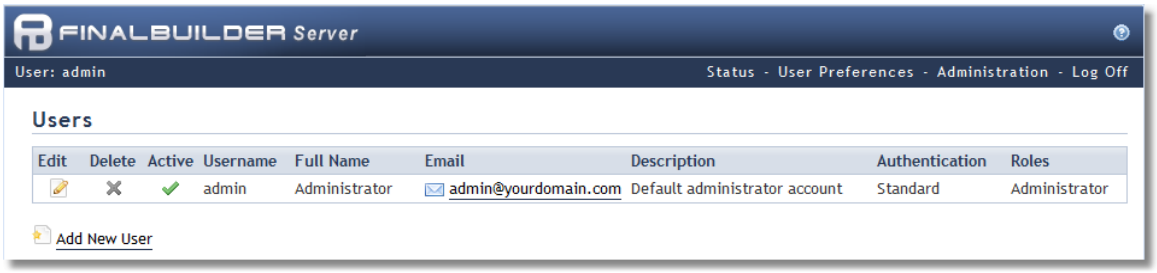


Note:

- You can change the authentication method used when you edit the user.
- When using the standard authentication method you can reset the user's password by click Set Password.

1.4.2.1.4 Deleting Existing Users

If you no longer require a user to be configured in the server then they can be removed from server by clicking on the delete icon for the relevant user. If its possible that this user will be added again in the future, consider marking the user as inactive as this will preserve the record of the user in the Build History. Only Active users will use up licenses.



Note:

- At least one user must exist with full administrator rights on the current build server.
- You can not delete the user that you are currently logged in as.

1.4.2.2 Managing Roles & Permissions

FinalBuilder Server allows you to restrict a users access or functionality to certain areas by the means of roles and permissions.

The **Administrator** role is read-only. It cannot be renamed, modified or deleted. There must be one user assigned to the Administrator role at all times.

The **Standard** role is added by default during installation, but it may be deleted, renamed or modified.

A user should be assigned at least one role and may have multiple roles. If a user doesn't have any role assigned, they won't have permission to perform the vast majority of functions.

If a role is marked as "Default", then it will be added automatically to any new users defined (it can be easily removed if not required though).

The permissions that roles grant are **additive** - that is, if a user has more than one role assigned, then they have permission to perform any action that is allowed by either or both of the roles. You cannot "ungrant" or "dissallow" a permission from a role, you'll need to create an alternative role that doesn't grant that permission and make sure that the other roles the user is assigned also don't grant that permission.

1.4.2.2.1 Permissions Overview

User access to parts of FinalBuilder Server can be restricted by the use of permissions. Each user is assigned one or more roles which define the permissions that the user will have. Below is a list of all permissions, grouped by subject.

Interact with Projects

- Starting projects
- Stopping projects
- Setting a user to be responsible for a failed project
- Taking responsibility for a failed project
- Clearing the entire build queue
- Removing single items from the build queue
- Suspending projects
- Resuming suspended projects

Manage Projects & Project Files

- Creating new projects
- Removing existing projects
- Modifying existing projects

- Uploading new project files to the build server
- Downloading existing project files from the build server
- Removing existing project files from the build server
- Configure the project level access for users on a project

Manage Variables

- Define which variables are to set when a project starts
- Remove a variable from being set when a project starts
- Modify previously defined variables.

Manage Notifications

- Set which users are to be notified and change what users are to be notified on
- Remove users from being notified
- Change which users are to be notified or change what users are to be notified on
- Change the notification templates that are sent to users

Manage Project Triggers and Conditions

- Create new project triggers
- Remove existing project triggers
- Modify existing project triggers
- Create new build conditions
- Remove existing conditions applied to a project
- Modify existing conditions

Manage Users

- Create new users on the management server
- Delete users from the management server
- Modify existing users on the management server

Manage Roles

- Create new roles that can assigned to users
- Delete existing roles
- Modify existing roles, including the permissions which are allowed with the role
- Assign roles to users

Manage Licenses

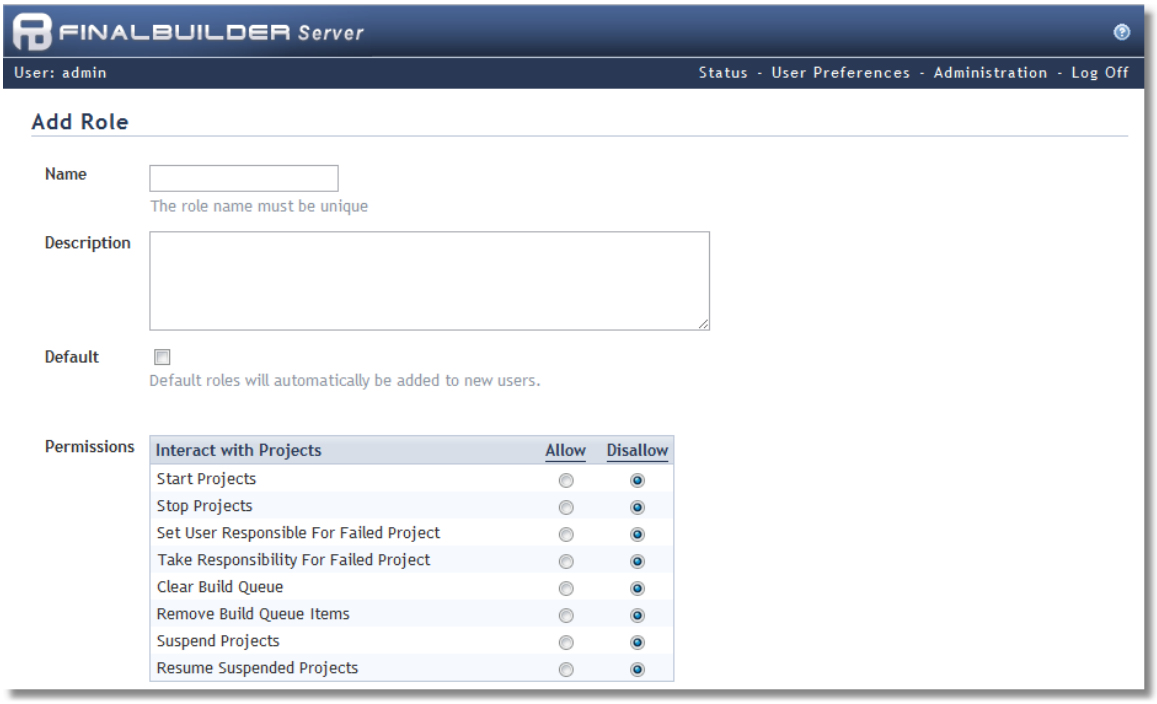
- View the licenses that have been uploaded to the management server
- Remove previously uploaded licenses from the management server
- Download licenses from management server
- Remove a build server from being licensed on the management server

Manage Configuration

- Configure the Smtp server that the build server will use to send notifications
- Change the appearance of the build server's status page, including the custom message text and the custom logo
- Change the default project settings that all projects can use, such as the default impersonation username and password and the maximum number of concurrent builds
- Alter the global notification templates, which are used when a project has not yet customized its own templates
- Viewing or clearing the application log

1.4.2.2.2 Adding a new Role

When creating new security roles you need to consider which permissions to allow for users who have this role. When a user has more then one role, each role will be queried when the user attempts to access a secured page, and the user will only be allowed access if at least one of the roles has the required permission enabled.



Name
The name of the role, which must be unique to all other roles defined on the management server.

Description
A brief note which can be used to describe who the role should apply to or what access a user will have when they have this role.

Default
When a role is set to be a default role, it will automatically be selected when you create

new users. Setting this value does not have any affect on existing users.

Permissions
These are the permissions that the user will be granted if they have the role is applied to them. You can toggle all radio buttons in a group by clicking on the 'Allow' or 'Disallow' headings. Please note that permissions are additive, so that if a user has multiple roles then they will be allowed to perform any action that is allowed in any of the roles.

1.4.2.2.3 Assigning Roles to Users

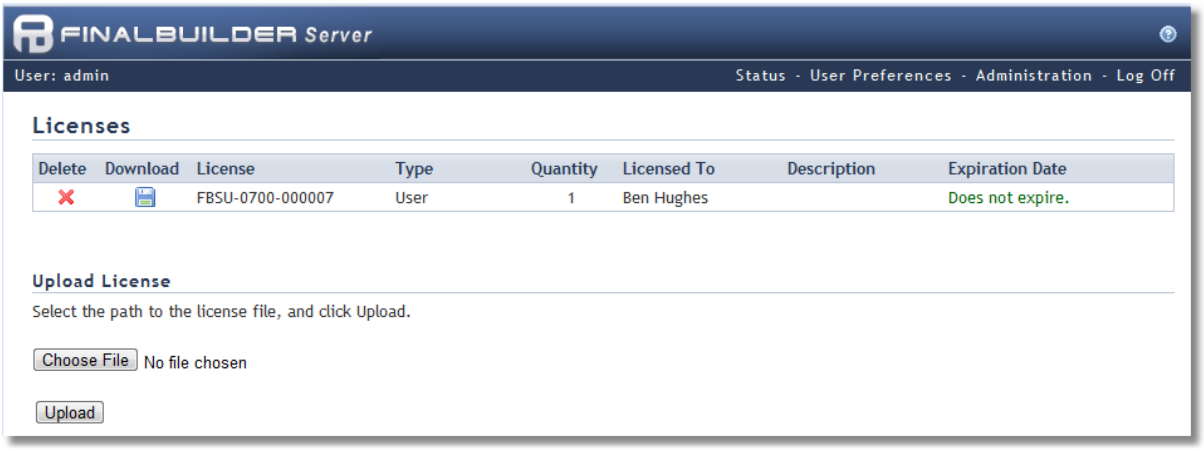
You can assign roles to a user when they are created or by editing an existing user. When assigning role to a user you must decide whether you want the user to have this role on all build servers connected to the current management server, or just on the selected build servers.

Role
Select the role which you wish to assign to the current user. You can only select roles which have not already been assigned to this user.

Apply to Servers
Choose whether this role should be assigned to this user on all builds servers or just on the ones selected. The build servers which have already been connected to the current management server will be listed, if the server is not listed then you can type the host name of machine which is hosting the build service.

1.4.2.3 Managing Licenses

FinalBuilder Server will allow a single user (including the the default admin user) to log in until one or more valid user licenses have been uploaded to the management server.



Uploading a License
You can upload a license file to the management server by browsing to the file and clicking upload. This will add all the licenses contained in the license file to the management server.

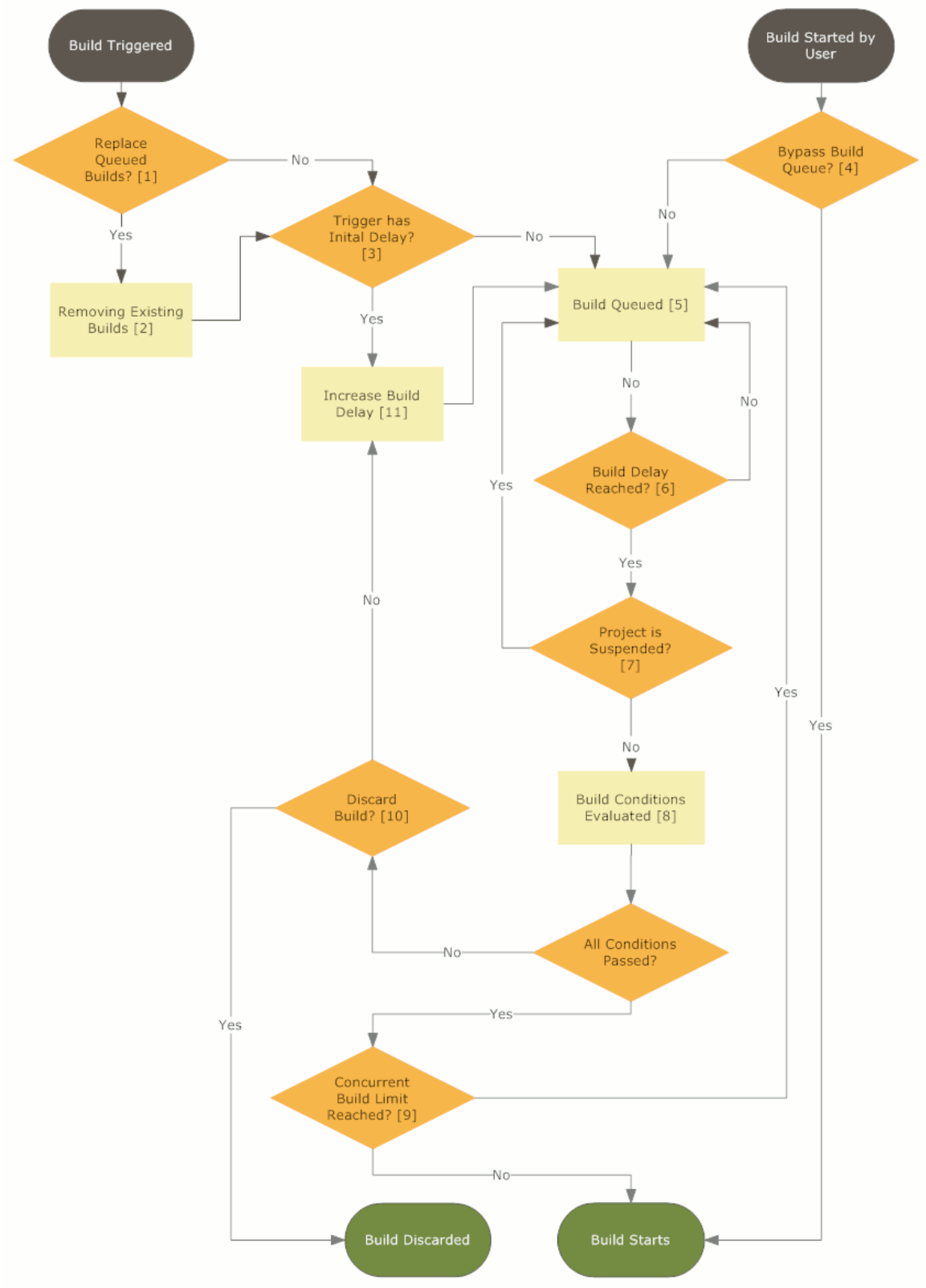
To view the current usage of user licences, you can view this information on the Users page.

1.5 Advanced Topics

1.5.1 Understanding the Build Queue

FinalBuilder Server runs an internal build queue to control when a build starts running, when a build should be re-queued and when a build should not be started at all.

The following flowchart illustrates the logic used to determine this process:



1. Replace Queued Builds

When configuring a trigger, you can decide whether you want to replace any existing builds of the same project that are currently on the build queue. This feature allows you to make sure that a trigger does not queue multiple builds up for the same project.

2. Remove Existing Builds
All builds that are for the same project as the one being started are removed from the build queue.

3. Trigger has Initial Delay
When configuring a source control trigger, you can specify a 'Quiet Period' that will force the build to start with an initial delay. This has the benefit of allowing you to check-in multiple files within in the specific time span without triggering a build for each check-in. If the build is already on the queue and the trigger triggers the build again, the delay will be reset to the value of the initial delay.

4. Bypass Build Queue
Occasionally a user will need to start a build instantly, without requiring that all conditions are met and without checking to make sure the maximum number of concurrent builds are running. When this happens the entire build queuing process is avoided and the build is started immediately (as long as its not already running). The user is prompted for this option whenever a build is started manually.

5. Build Queued
At this stage, the build is placed on the queue and will continue to wait there until the build delay has been reached.

6. Build Delay Reached
Each build on the queue has a value to indicate how long it should wait on the build queue before FinalBuilder Server attempts to start it. When the delay has been reached, the project is checked to see if it currently suspended then the build conditions are evaluated.

7. Project is Suspended
Users may decide that a project should not be started regardless of any builds conditions, this may be because they are working on the FinalBuilder project file or they are working on the machine hosting the build server. While a project is suspended it will continue to wait on the build queue until the suspension is removed or expires.

8. Build Conditions Evaluated
Each project can define a set of conditions which are tested each time the build queue attempts to start the project. Each of the conditions defined specifies whether the build should be discarded or whether it should be re-queued and how long it should be re-queued for. The build queue will test all conditions currently active on the project and for each condition that has not been met, will choose the condition that has the most impact on the build. Conditions that force the build to be discarded are considered to have the most impact on the build, and the rest are sorted by the length of the time the build delay will be increased.

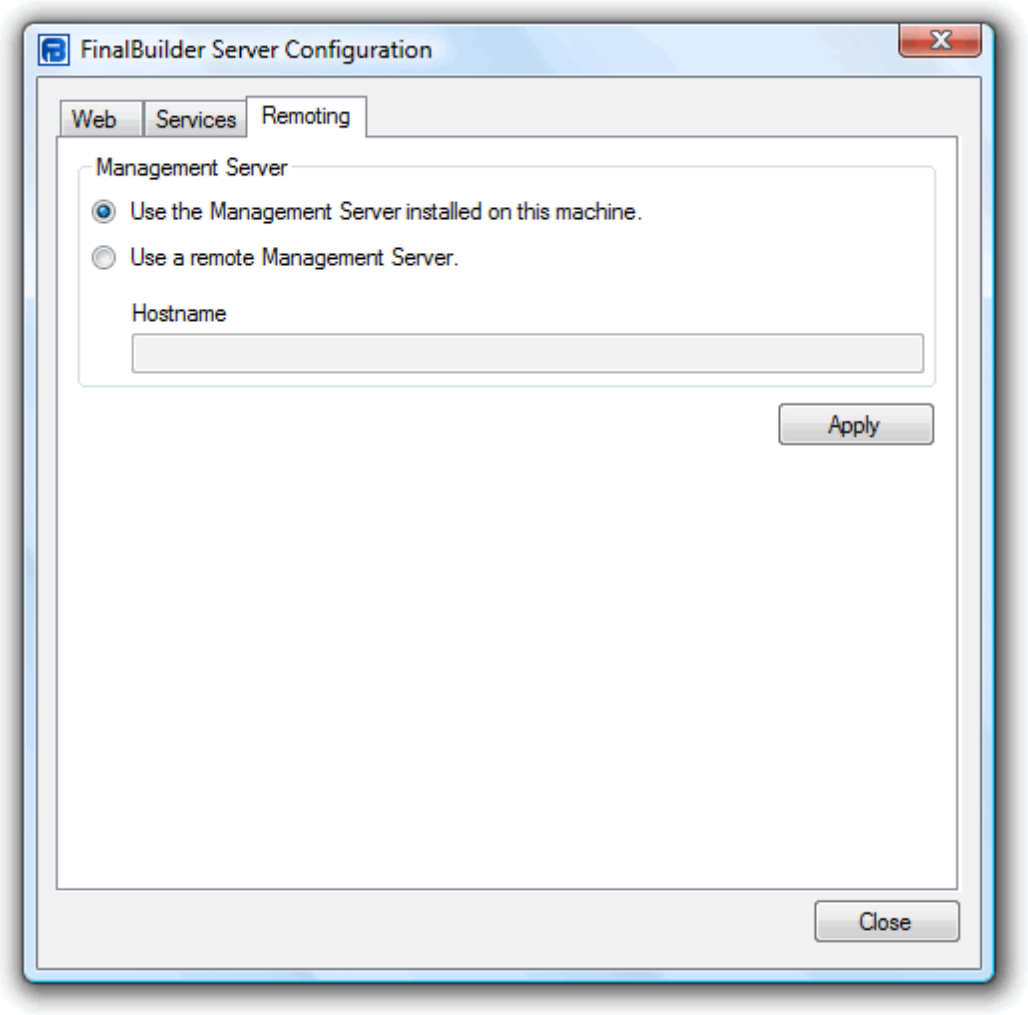
9. Concurrent Build Limit Reached
A build server can specify the maximum number of concurrent builds that can be running at any one time on the configuration page. The maximum number of concurrent builds is not checked until after the conditions have been evaluated so that builds may still be discarded from the queue, which can help to free up the build server in the future.

10. Discard Builds
When a build condition is configured to discard the build and the condition is not met, the build will be removed from the queue.

11. Increase Build Delay
When a build condition is configured to queue a build when the condition is not met it will be re-queued onto the build queue and the delay adjusted according to what has been set on the condition.

1.5.2 Configuring Multiple Build Servers

When you have more then a single build machine you may wish to use a central management server to manage the users, security, licensing and user profiles between all the build servers.



Notes

- The machine which hosts the management service will need to be online and must

be able to accept incoming connections from the machine which hosts the build service for the build server to be operable.
- When multiple build servers share a single management service, only a single user license is required for each user across all builds servers.

1.5.3 Web Service API

FinalBuilder Server exposes a web service which can be used to programmatically control builds.

The URL to the web service is *[build_machine]*/Services/FinalBuilderServer.asmx, where *[build_machine]* is the URL to machine hosting FinalBuilder Server, including the virtual directory if applicable.

Their are currently 8 operations which are supported:

Authenticate
Authenticates the specified username and password and returns an authentication token (GUID String) which can be passed into other web service operations.

Parameters:
Username - *string*
Password - *string*

Returns:
Authentication Token - *string*

Create User
Creates a new standard user on the build machine with the specified parameters.

Parameters:
Authentication Token - *string*
Username - *string*
Password - *string*
Name - *string*
Email - *string*
Roles - *array of string*

Get Project Names
Gets the names of all projects on the server which the authenticated user has access to.

Parameters:
Authentication Token - *string*

Returns:
Project Names - *array of string*

Get Project Status
Gets the current status of the specified project.

Parameters:
Authentication Token - *string*
Project Name - *string*

Returns:
Project Status - *ProjectStatus [NeverRun, Running, Failure, Success, ConfigurationError, Stopping, Terminating, Terminated, Suspended]*

Get Project Trigger Log
Gets the trigger log of the specified project.

Parameters:
Authentication Token - *string*
Project Name - *string*

Returns:
Trigger Log - array of *string*

Start Project
Adds the specified project to the build queue.

Parameters:
Authentication Token - *string*
Project Name - *string*

Start Project With Variables
Adds the project to the build queue with the variable values specified. Variables to be specified in format '*VariableName=VariableValue*'

Parameters:
Authentication Token - *string*
Project Name - *string*
Variables - array of String

Stop Project
Stops the specified project if it is currently running.

Parameters:
Authentication Token - *string*
Project Name - *string*

User Exists
Checks whether the specified user exists.

Parameters:
Username - *string*

Returns:
Result - *boolean*

1.5.3.1 C# Example

Generating the Proxy Class

Included as part of the .Net Framework tools is the Web Services Description Language Tool (Wsdl.exe). Using Wsdl.exe, you can create a C# proxy class which can be used to communicate with the FinalBuilder Server web services.

Usage:

wsdl.exe /n:*[namespace]* /o:*[output_file]* *[build_machine]*/Services/FinalBuilderServer.asmx?wsdl

Substituting the following:

[namespace] - The namespace that the generated source file should have.
[output_file] - The location where you want the source file to be generated.
[build_machine] - The URL to machine hosting FinalBuilder Server, including the virtual directory if applicable.

For example:
wsdl.exe /n:"VSoft" /o:"C:\Proxy.cs" http://MyBuildVM/Services/FinalBuilderServer.asmx?wsdl

Using the Generated Proxy Class

You will need to add the proxy class to your C# project and include a reference to 'System.Web.Services'.

```
using System;
using VSoft;

namespace WebService.Sample
{
   public class Program
   {
      public static void Main(string[] args)
      {
         // Instantiate the FinalBuilder Server Proxy Class.
         FinalBuilderServer server = new FinalBuilderServer();

         // Authenticate the user.
         string authToken = server.Authenticate("paul", "my_password");

         // Get the list of all projects configured on this build server.
         string[] projectNames = server.GetProjectNames(authToken);

         foreach (string projectName in projectNames)
         {
            // Get the status of the project
            ProjectStatus status = server.GetProjectStatus(authToken, projectName);
            Console.WriteLine("Project '{0}' state is currently '{1}'.", projectName, status);

            // Get the trigger log of the project
            string[] log = server.GetProjectTriggerLog(authToken, projectName);
            Console.WriteLine("Project '{0}' Trigger Log:", projectName);
```

```
            foreach (string line in log)
            {
               Console.WriteLine(line);
            }
         }

         Console.ReadLine();
      }
   }
}
```

1.5.3.2 VB.NET Example

Generating the Proxy Class

Included as part of the .Net Framework tools is the Web Services Description Language Tool (Wsdl.exe). Using Wsdl.exe, you can create a VB.NET proxy class which can be used to communicate with the FinalBuilder Server web services.

Usage:

wsdl.exe /n:*[namespace]* /o:*[output_file]* */l:VB [build_machine]*/Services/FinalBuilderServer.asmx?wsdl

Substituting the following:

[namespace] - The namespace that the generated source file should have.
[output_file] - The location where you want the source file to be generated.
[build_machine] - The URL to machine hosting FinalBuilder Server, including the virtual directory if applicable.

For example:
wsdl.exe /n:"VSoft" /o:"C:\Proxy.vb" /l:VB http://MyBuildVM/Services/FinalBuilderServer.asmx?wsdl

Using the Generated Proxy Class

You will need to add the proxy class to your VB.NET project and include a reference to 'System.Web.Services'.

```
Imports System
Imports WebService.Sample.VBNet.VSoft

Namespace WebService.Sample
   Public Class Program
      Public Shared Sub Main(ByVal args As String())
         ' Instantiate the FinalBuilder Server Proxy Class.
         Dim server As New FinalBuilderServer

         ' Authenticate the user.
         Dim authToken As String = server.Authenticate("paul", "my_password")

         ' Get the list of all projects configured on this build server.
         Dim projectNames As String() = server.GetProjectNames(authToken)

         For Each projectName As String In projectNames
            ' Get the status of the project
```

```
            Dim status As ProjectStatus = server.GetProjectStatus(authToken, projectName)
            Console.WriteLine("Project '{0}' state is currently '{1}'.", projectName, status)

            ' Get the trigger log of the project
            Dim log As String() = server.GetProjectTriggerLog(authToken, projectName)
            Console.WriteLine("Project '{0}' Trigger Log:", projectName)

            For Each line As String In log
                Console.WriteLine(line)
            Next
        Next

        Console.ReadLine()
    End Sub
End Class
End Namespace
```

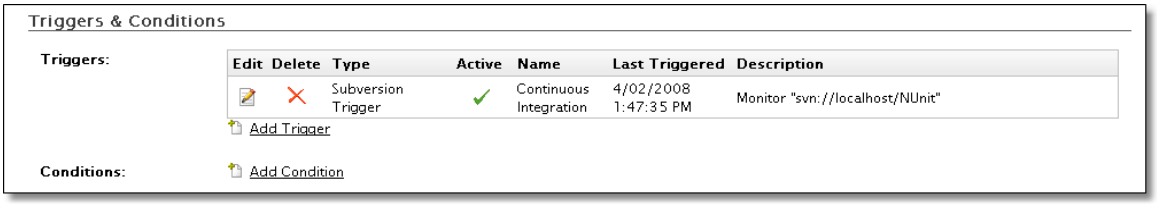
Part II

Triggers Reference

2 Triggers Reference

What are Triggers?
Triggers can be used to start a project based upon an event. For example, the Time Trigger will start a project when the current time passes the specified time.

How to add a Trigger to a Project?
A trigger can be added to a project navigating to the 'Edit Project' page. The 'Edit Project' page is accessible by clicking 'Edit' on the project, on the 'Server Overview' page.



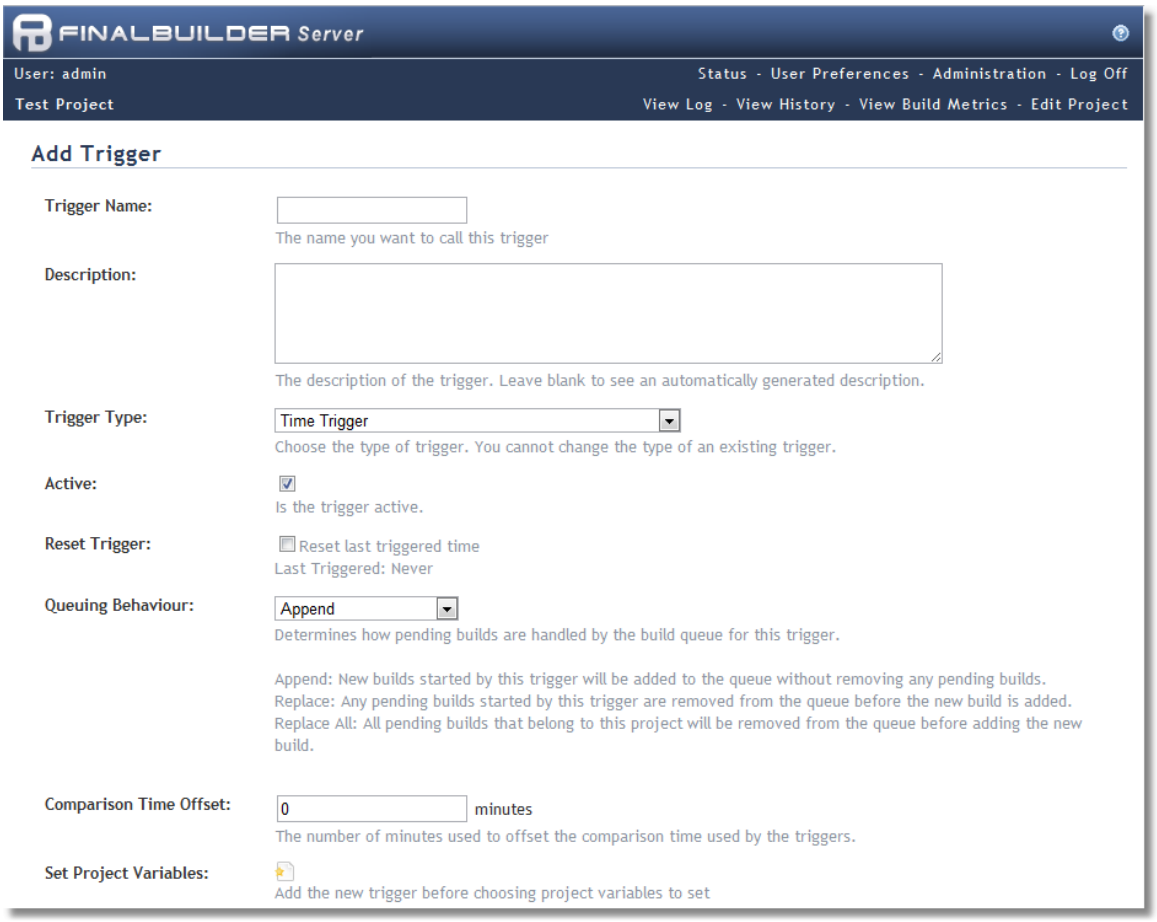
Clicking on the 'Add Trigger' link will navigate to the 'Add Trigger' page where the new trigger can be configured.

Select the Type of the new trigger by using the "Trigger Type" combo box.

When the Trigger Type changes, the bottom part of the Add Trigger page will reload and show the options for the new trigger type. Please note that once a trigger is defined (saved), then you can't change the trigger type.

2.1 Trigger Types

FinalBuilder Server supports a variety of ways to trigger a build.



Version Control Triggers
Triggers that interface with a version control system to trigger builds when a source control item is checked in. These triggers are typically used to setup FinalBuilder Server to perform continuous integration.

File-Based Triggers
Triggers that watch or interact with a file on the local file system and trigger a build based on some pre-determined conditions.

Time Triggers
Time triggers allow you to schedule a build at certain dates or times.

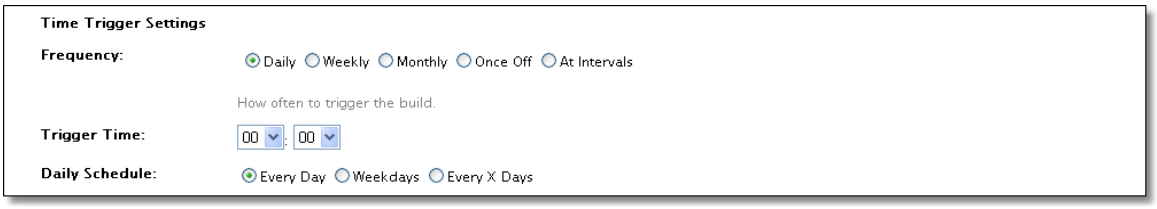
Run Process
The Run Process trigger allows you to run a process and then trigger a build based on the result of the process.

Time Trigger | File Trigger | Run Process Trigger | Version Control Triggers

2.1.1 Time Trigger

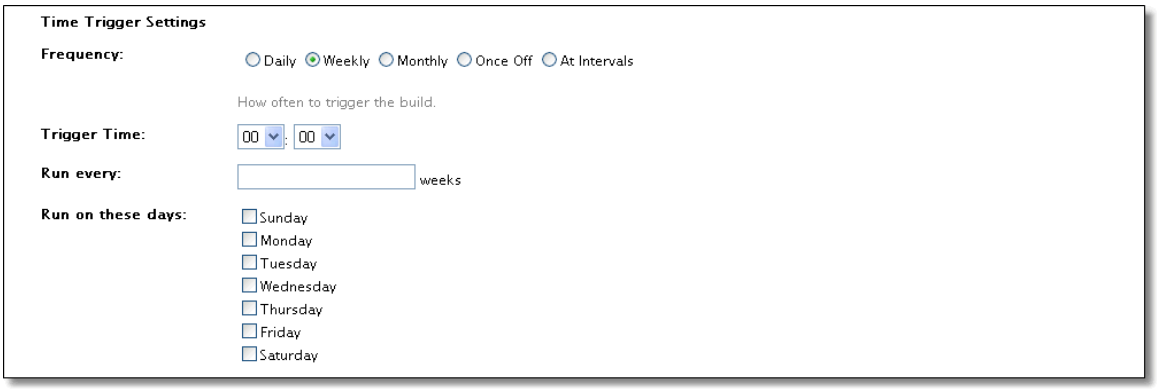
The **Time Trigger** can be configured to trigger a build Daily, Weekly, Monthly, Once Off or at Intervals.

Daily



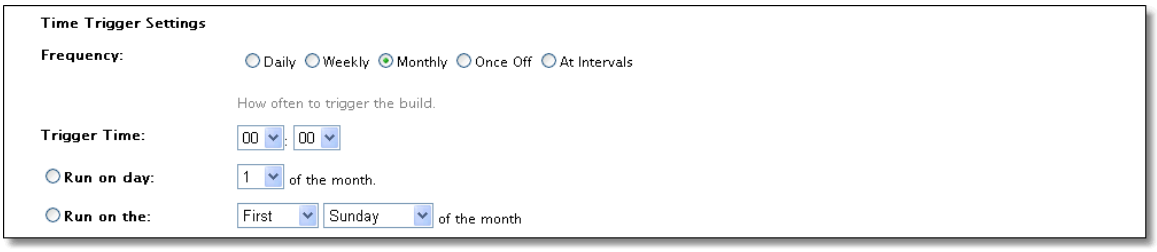
Every Day - The project will be triggered everyday.
Weekdays - The project will be triggered only on week days (Monday - Friday).
Every X Days - The project will be triggered every X days.

Weekly



The project will be triggered every X weeks, on the selected days.

Monthly



The trigger can be configured so that the project will run either on a set day of the month, or on a set position in the month (for example, 'Second Sunday of the month').

Once Off

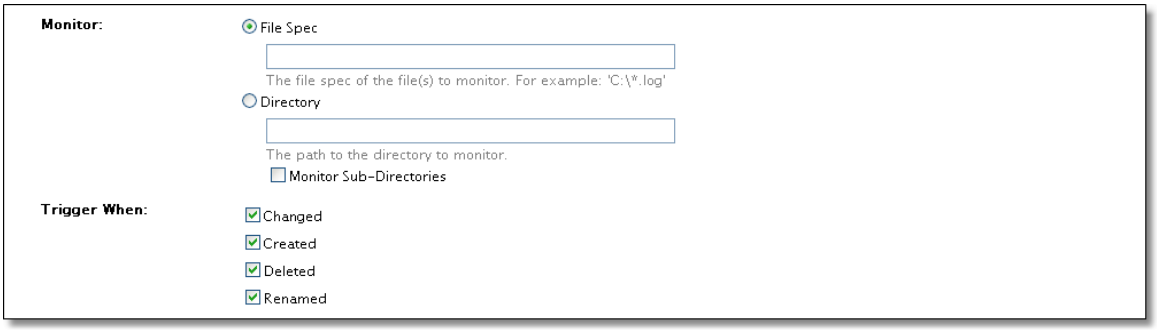
The trigger can be configured so that it will only run once, on the specified date.

At Intervals

The trigger can be configured to run every X minutes, hours, days or weeks. The trigger can also be configured not run between certain times and on certain days.

2.1.2 File Trigger

The **File Trigger** allows you to start a build when a file or directory is Changed, Created, Deleted or Renamed.



Monitor File Spec
Configures the File Trigger to monitor the specified file(s) and trigger when they have changed.

Monitor Directory
Configures the File Trigger to monitor a directory (and optionally its subdirectories) and start a build when a change occurs.

Trigger When Changed
Triggers a build when changes are made to the size, system attributes, last write time, last access time or security permissions of a file or directory.

Trigger When Created
Triggers a build when a file or directory is created.

Trigger When Deleted
Triggers a build when a file or directory is deleted.

Trigger When Renamed
Triggers a build when a file or directory is renamed.

2.1.3 Run Process Trigger

The Run Process Trigger allows you to launch any Windows process and trigger a build based on what happens.

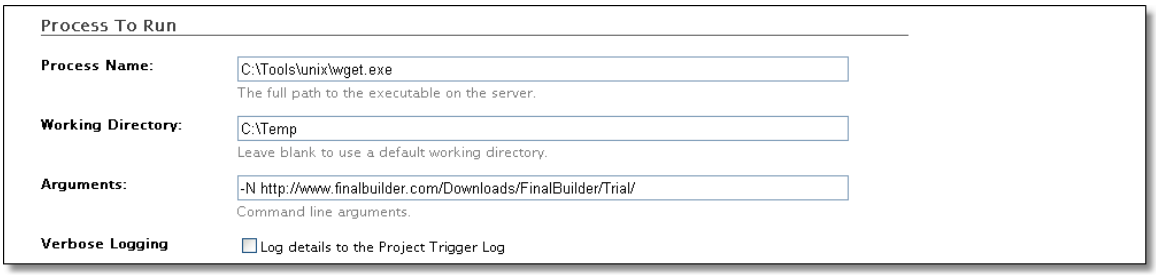
Triggering can be based on any of three things:

- Process output
- The process exit code
- Files modified while the process is running

Example

The example given below uses GNU WGet (for Windows) to monitor a remote URL and trigger the build if the file changes.

The URL we will monitor is the trial download of FinalBuilder. Any time it changes, we want to trigger a build.

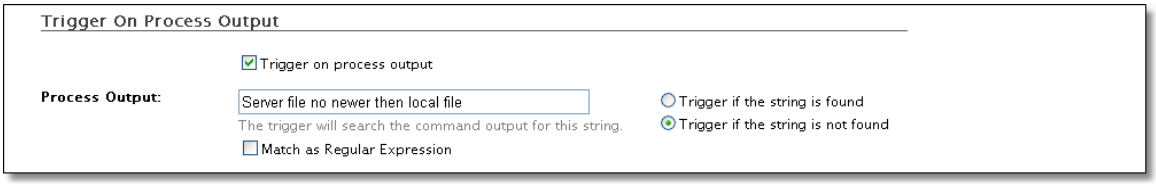


Process Name
The name of the process to run. This time we didn't actually need to put the full path here, because 'wget.exe' is on the system PATH of the server computer.

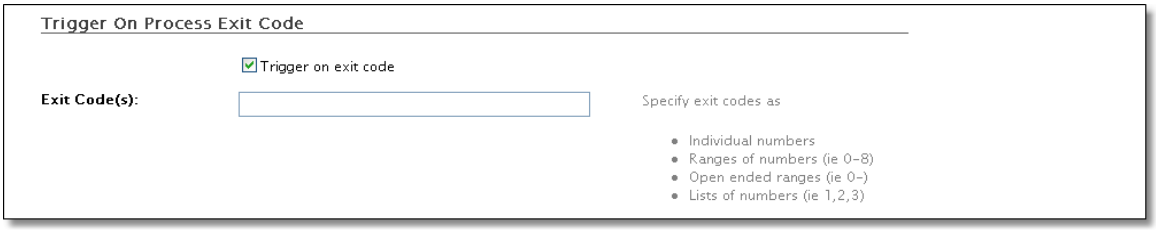
Working Directory
The directory to run the process in. In this case, this will be the directory that the remote URL is download to.

Arguments
The command line argument, '-N', tells WGet not to download the file unless it is newer than the version on disk.

Verbose Logging
Specifies that the Project Trigger Log will be filled with all the output from the executable each time it runs, as well as some extra information on trigger matching. This option produces a lot of log output, but it can be useful when setting up the trigger.



Process Output
Searches the output of the program that is executed and matches the text specified. You can choose whether to trigger the build when the text is found or when the text is not found. In the example, we search the output of WGet to find the text 'Server file no newer then local file' and trigger the build if the string is not found.



Exit Code
Compares the exit code returned by the process run against the specified value(s). In this example, we do not require that the exit code be tested.

Tip: Process exit codes can be quite unpredictable. It's recommended that you use one of the other trigger methods when possible.

It is not necessary to have 'Trigger On Files Changes' *as well as* 'Trigger on Process Output'. They're both enabled just for the sake of the example.

Monitor File Path
FinalBuilder Server will monitor the downloaded file and trigger whenever it changes. Alternatively, you could specify the entire C:\Temp directory in order to trigger whenever any file in the directory changes.

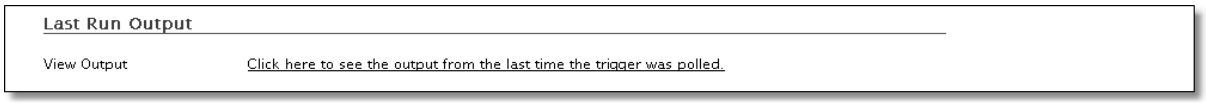
Note: The trigger can't actually tell whether the files change because of the process running, or for some other reason. It's best if you use a 'sandbox' directory where you can control what processes make changes.

Feedback

There are two main ways to get feedback from the Run Process Trigger.

The first is to use the 'Verbose Logging' option and look at the process output in the Project Trigger Log.

The second is to go to the Edit Trigger page after the trigger has run, and click on the link to Show Last Output:



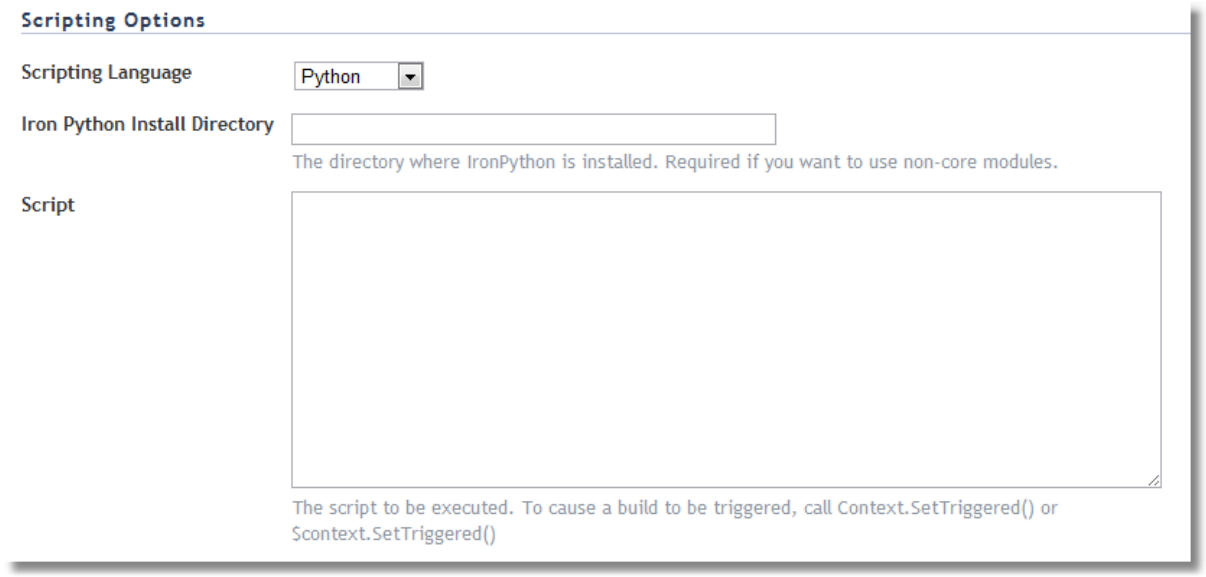
Once clicked, the section expands to look like this:

Note: This output does not refresh until you reload the entire page in your browser.

If the process has been running for more than one minute, you will be given the option to Terminate it. See the Trigger Errors topic for details.

2.1.4 Script Based Triggers

Script Based Triggers allow you to use a Python or PowerShell script to determine if a build will be triggered.



Language
The language you wish to use for the script. Currently Python and PowerShell are supported.

Iron Python Install Directory
Optional: the directory where Iron Python is installed. Specifying this location allows you to import any of the modules shipped with IronPython.

Script
The script to execute. In order for a build to be triggered, either Context.SetTriggered() or $context.SetTriggered() must be called. See below.

The Context object
An object is made available to both languages that allows you to interact with the build server. In Python, it is exposed as a module called Context, which you do **not** need to import. In PowerShell, it is exposed as a variable called $context.

Context methods
- SetTriggered(): This is the most important method of the Context object. Calling this method means that the build will be queued
- AddTriggerOutputMessage(message): The string parameter passed to this method is logged in the build queue. Use this to provide information about the circumstances triggered the build
- AddFileModificationItem(name, action, user, comment, type, version): Adds information about a file that was involved in the build triggering. This is logged and also passed to FinalBuilder, for use in the Trigger Files Iterator action
- ExecuteProgram(filename, workingDirectory, arguments): This method runs an executable *under the context of user set to run the build.* It returns an ExecutionResult object
- ComparisonDateTime: a DateTime property used to indicate when the trigger was last triggered, or when the last build started (which ever is more recent). It is useful when, for example, you are checking against a version control system and only want to find changes since the last build.

Context.py stub
A stub IronPython module has been provided to allow you to test scripts without needing to run them on the build server. This can be found in the <FinalBuilder Server Install Dir>\Examples.

Example.py
A simple IronPython script demonstrating how to use the Context module. This can be found in the <FinalBuilder Server Install Dir>\Examples.

2.1.4.1 Python Example

This is an example IronPython script that can be tested outside of FinalBuilder Server using the Context.py stub (see Script Based Triggers).

```
import Context

#copy from here

from System import DateTime

def TriggerTest():
    app = 'cmd.exe'
    wd = r'C:\Windows\System32'

    if Context.ComparisonDateTime <  DateTime.Now.AddMinutes(-30):
           result = Context.ExecuteProgram(app, wd, '/c "echo hi"')
           if result.ExitCode == 0:
                  Context.SetTriggered()
                  Context.AddTriggerOutputMessage('%s successful, build will be
triggered' % app)
           else:
```

```
                print 'Build will not be triggered'
                print '%s failed with return codee %d: %s ' % (app, result.
ExitCode, result.StdError)
    else:
            print 'Build has been run in the last 30 minutes - not running'

TriggerTest()

#copy to here

print Context.Output()
```

Things to note:

- Only the code between the copy from/copy to comments would be copied to the Trigger script
- You can import modules, such as time. To do this you will need to specify the install directory of Iron Python
- you can use sys.path.append() to add any other directories from which you would like to import modules
- 'print' statements from the last run will appear under the Script area of the trigger

2.1.4.2 PowerShell Example

Triggering a build with PowerShell is as simple as:

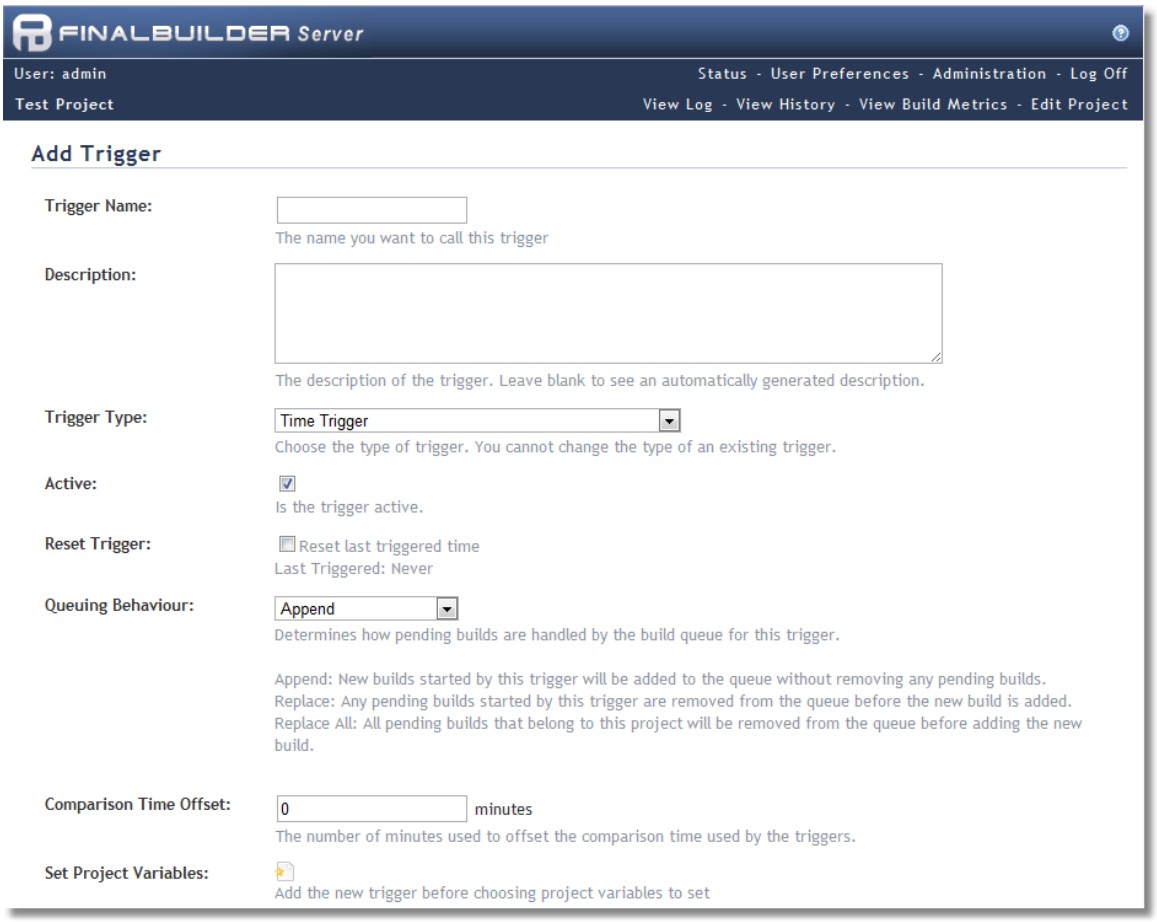
```
$context.SetTriggered()
$context.AddTriggerOutputMessage("Triggered by PowerShell!")
```

To test your script before adding it to the trigger, you can load the Trigger and TriggerExecutionContext objects from FinalBuilderServer.Build.dll:

```
[reflection.assembly]::LoadFile("FinalBuilder Server Install Dir>\FinalBuilderServer.Build.dll
$trigger = new-object VSoftTechnologies.FinalBuilderServer.Triggers.ScriptBasedTrigger
$context = new-object VSoftTechnologies.FinalBuilderServer.Triggers.TriggerExecutionContext($t
```

2.1.5 Version Control Triggers

Version control triggers allow a project to be started whenever a check-in is performed on a source control repository (otherwise known as continuous integration).



Queuing Behaviour
Changes how the build queue handles multiple pending builds of the same project on the build queue.

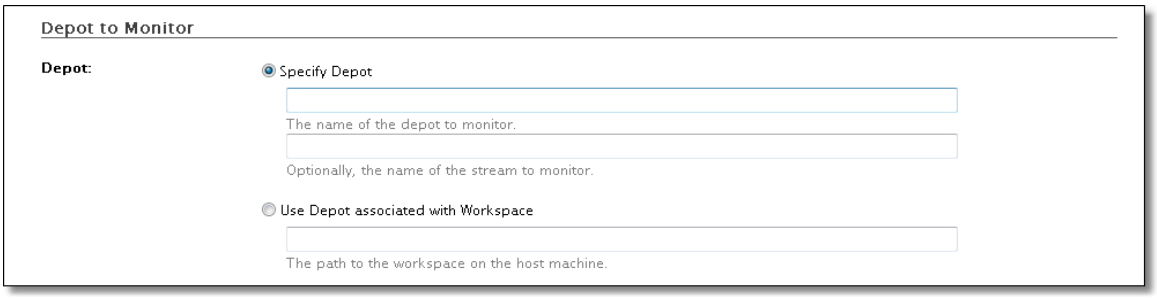
Quiet Period
The quiet period is the number of minutes a build will wait on the build queue before it starts.

Tips
Using version control triggers to trigger builds when files are checked into source control can cause the build server to have a project continuously running, as each time a file is checked into source control, the build is added to the build queue. FinalBuilder Server allows you to prevent this by setting the Queuing Behaviour to Replace or Replace All and setting a 'Quiet Period'.

2.1.5.1 Accurev Trigger

The Accurev Trigger allows you to monitor a Accurev Depot for changes, and trigger builds whenever new changes are checked in.

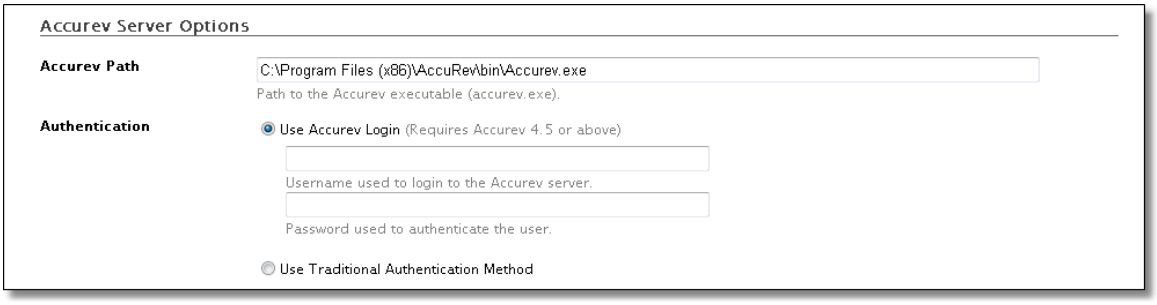
General information on creating triggers can be found under the Triggers Reference topic.



You can choose to have the Accurev trigger monitor a specific depot and optionally a stream in that depot, or you can configure the trigger to monitor the depot associated with a local workspace.

The Accurev trigger allows you to automatically update the local workspace when the build is triggered, you can only use this option when you have specified to use the depot associated with a depot.

You can also force the client and server times to be kept in sync by enabling 'Synchronise time between client and server', when this option is enabled the Accurev trigger will execute the 'synctime' Accurev command before checking for changes.



The Accurev trigger will attempt to locate the accurev.exe executable by looking through the windows registry. If the path is not found, you will need to provide the path to the accurev.exe executable.

Depending on which version of Accurev that you are using, you can either choose to explicitly login prior to checking for updates by using the 'Use Accurev Login', otherwise the Accurev trigger will use the credentials associated with the user who the FinalBuilder Server project is set to run under.

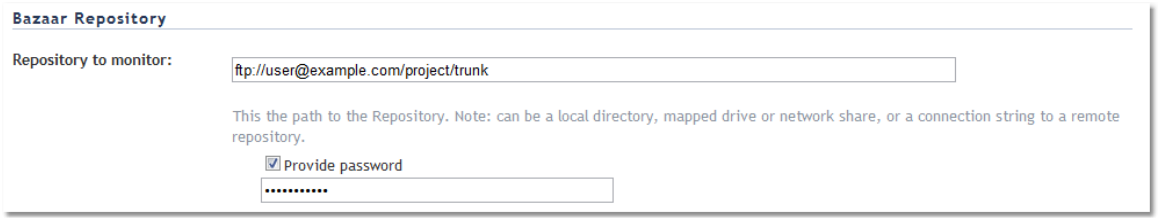
2.1.5.2 Bazaar Trigger

The Bazaar Trigger allows you to monitor a Bazaar Repository for changes. A build is triggered when a commit occurs on the repository being monitored or when changes are pushed and updated into the repository being monitored. The monitored repository may be either a local repository or a remote repository. Using the Update on Trigger option, a local branch or checkout of a remote repository can be updated when the trigger is triggered.

General information on creating triggers can be found under the Triggers Reference topic.

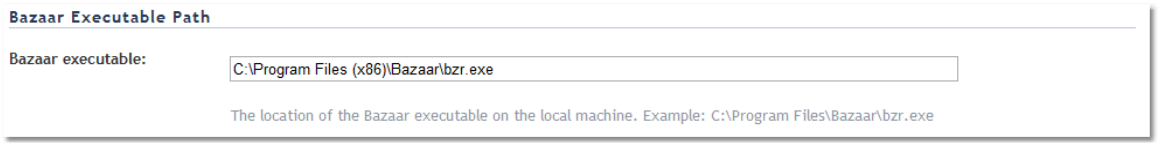
When adding a Mercurial trigger you are required to supply the following details:

Repository to monitor - This is the location of the Bazaar repository that you want to monitor for changes. This can be a local repository (local directory, mapped drive or network share) or a remote repository (using a URL as specified in this reference).

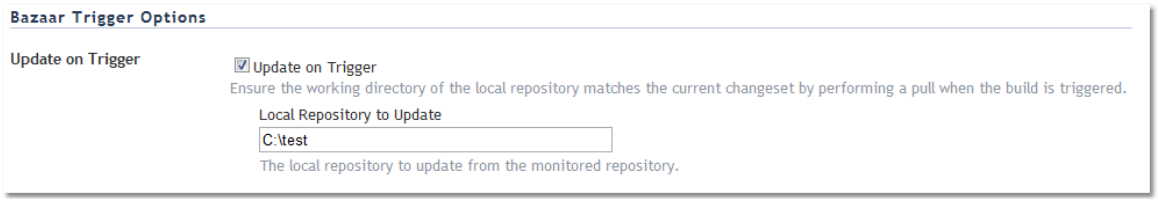


- **Provide Password** - If using a remote connection string, use this option to provide a password, as opposed to having it stored as plain text as part of the string.

Bazaar executable - This is the location of the Bazaar executable file on the local machine.



Bazaar Trigger Options



- **Update on Trigger** - Perform an update on a local repository when the build is triggered to ensure that the working directory is up to date. If checked, the trigger will perform a "pull" action going from the monitored repository to the specified local repository.

Exclude Files From Monitoring

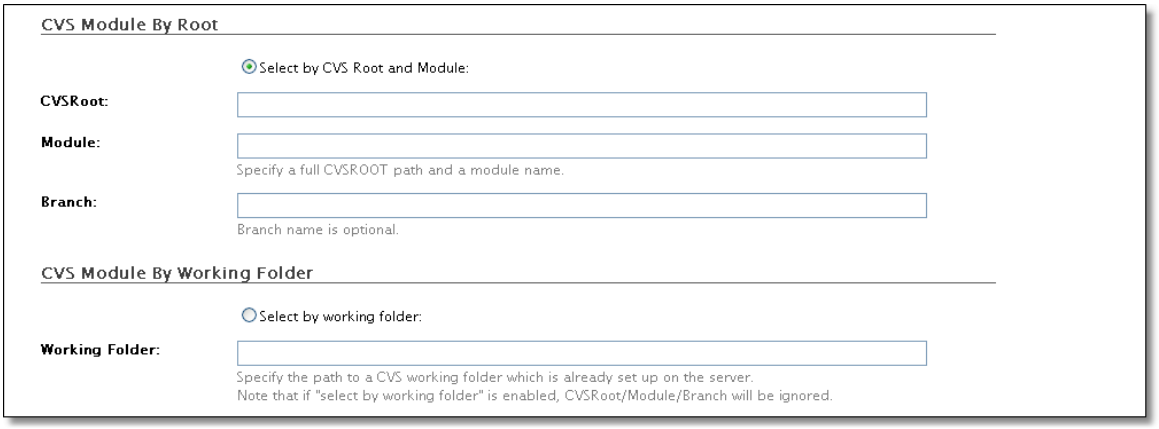
See Excluding Files From Source Control Monitoring

2.1.5.3 CVS Trigger

The CVS Trigger allows you to monitor a CVS source control repository for changes, and trigger builds whenever new changes are checked in. The CVS Trigger is designed to be used with the CVSNT version of CVS, although it should work with other CVS command line clients.

General information on creating triggers can be found under the Triggers Reference topic.

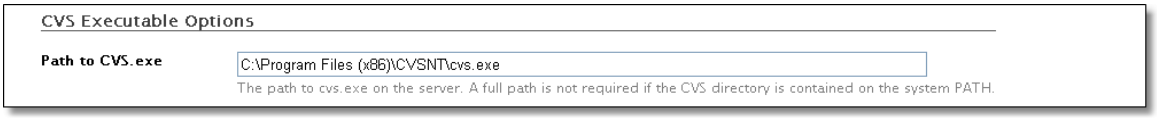
The CVS trigger supports the following options:



You can choose to select the CVS Module to monitor, *either* by the CVSRoot and Module path, or via the Working Folder. Only one option needs to be filled in.

To select via CVS Root, specify the CVSRoot (including the protocol) and the module name, and optionally a branch. If the connection requires a login, then it must be specified in the CVSRoot string. Alternatively, log in to CVS via the CVSNT client and then map a local working folder, and then use the CVS Module by Working Folder option.

To select via working folder, map a working folder on the server and then enter the full path to the folder. If this option is selected, the CVSRoot/Module/Branch settings are ignored.



Path to the CVS.exe executable. This path should be automatically detected if CVSNT is installed.

Exclude Files From Monitoring

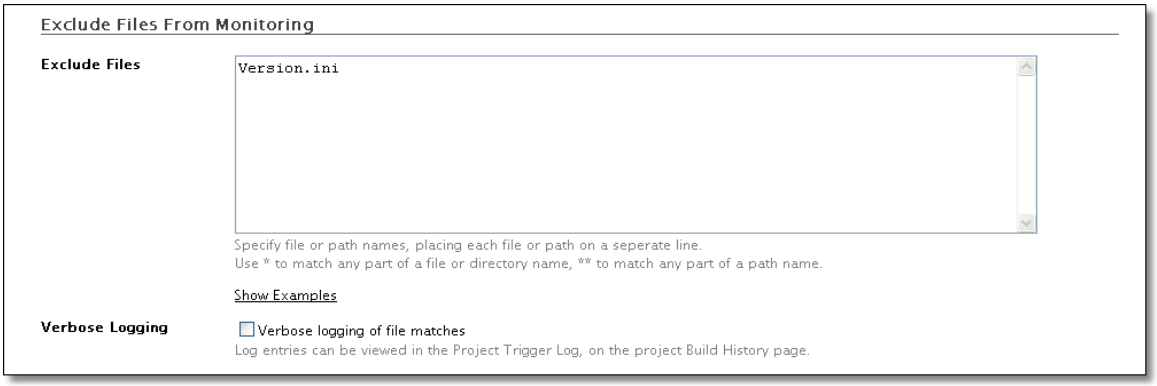
See Excluding Files From Source Control Monitoring.

2.1.5.4 Excluding Files from Source Control Monitoring

All of the version control trigger types include an option to exclude files from triggering the build.

This option can be useful when there are files which are changed in source control as part of the build process (eg. version information), or when there are files in the source control repository which should not trigger the build process (eg. internal documentation files.)

The Exclusion options are available at the bottom of the Add Trigger/Edit Trigger page for each Version Control trigger type:



In the above example, the trigger is set to exclude the 'Version.ini' file. Adding 'Build/Installer/' as an extra line would also exclude all files in the Build/Installer repository from triggering builds.

Click the 'Show Examples' link on the page to see a quick list of example patterns.

Special Characters

The following special characters can be used to match files:

Character	Meaning
*	Matches any part of a file name, but not path delimiters / or \.
**	Matches anything, including path delimiters.
?	Matches any single character, apart from path delimiters / or \.
/ and \	Path delimiters, are considered identical and can be used interchangeably.
;	Separates multiple Exclude Files patterns.

Tips

- Exclude Files entries are matched on either the file name, or the full path name.
- Use ** at the beginning or end of the path name to match any parent or subdirectory combinations.
- The files which are matched are returned as source control repository paths, with the particular format being determined by the source control server.
- Path delimiters / and \ are considered identical (ie forward slashes will match back slashes, and vice versa.)

Verbose Logging

Enable this option in order to make it easier to debug and monitor File Exclude matches.

When this option is enabled, the Project Trigger Log will record the names of any modified files, and whether or not any were excluded from triggering the build.

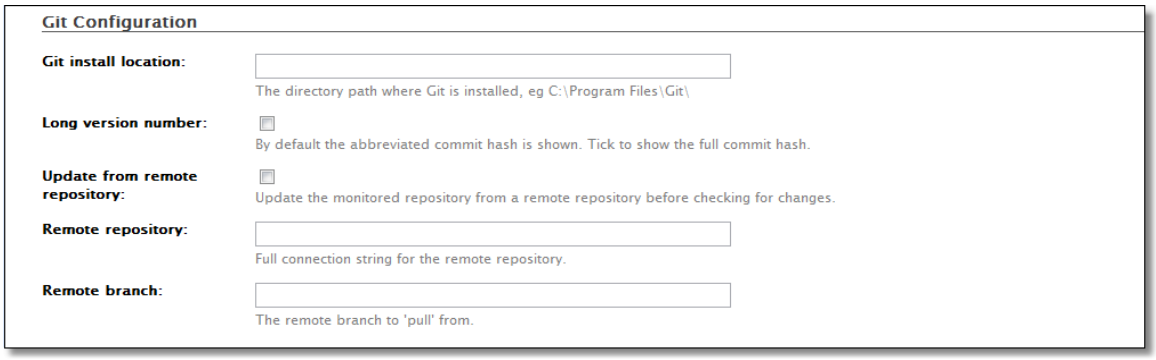
2.1.5.5 Git Trigger

The Git Trigger type allows you to monitor a Git source control repository and automatically start builds when changes are committed.

General information on creating triggers can be found under the Triggers Reference topic.

The Git Trigger Type contains the following options:

Git only allows actions to be performed on a local repository, so you must specify a repository that is in a local or mapped network drive.



The Git executable location is the directory where Git is installed. Note, this is the directory that **contains** the bin directory, not the bin directory itself.

The Git trigger uses the commit hash as the version number. By default, the abbreviated commit hash is used. The Long version number option allows you to instead use the full commit hash.

Monitoring a remote repository
Git does not allow the 'log' command to be run on remote repositories, so they cannot be monitored directly. The easiest solution is to map a network drive to the server that contains the repository. If that is not possible you can create a local clone of the remote repository on the server where FinalBuilder Server resides.

The Git trigger can update this local clone from a remote repository before checking for changes, ensuring the trigger is working on the most recent version of the repository.

To do this, tick the "Update from remote repository" checkbox, specify the remote repository location and the branch to pull from. The repository location can be any valid Git URL, for example:

- i:\data\git\TestProj
- git@github.com:<username>/TestProj.git
- git://github.com/<username>/TestProj.git
- http://github.com/<username>/TestProj.git

All that is required is that the user FinalBuilder Server is running as has read access to the repository.

Using SSH
If you have a private Git repository you can authenticate via SSH. To set this up:

- log into the FinalBuilder Server host server as the user that FinalBuilder Server runs as
- Use ssh-keygen to create an SSH key pair **without** a password. The keys will be created in %USERPROFILE%/.ssh/
- Add the generated public key to the remote Git server
- Test that you can pull from the remote repository via the git command line tools

NB Because FinalBuilder Server is non-interactive, you **cannot** password protect the private key.

Exclude Files From Monitoring

See Excluding Files From Source Control Monitoring.

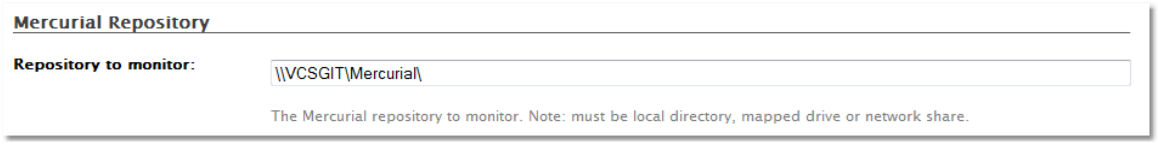
2.1.5.6 Mercurial Trigger

The Mercurial Trigger allows you to monitor a Mercurial Repository for changes. A build is triggered when a commit occurs on the repository being monitored or when changes are pushed and updated into the repository being monitored. By making use of the Remote Repository Monitoring options you can specify a remote repository to monitor, if a change is detected, the local repository will be updated to match before checking the local repository for changes.

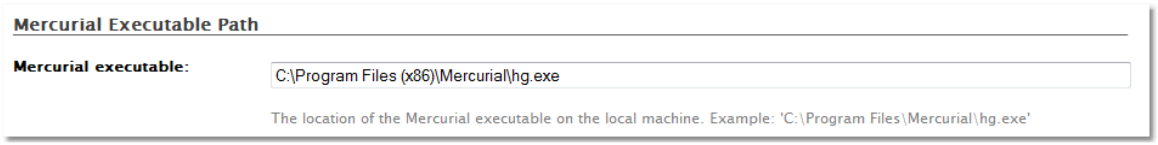
General information on creating triggers can be found under the Triggers Reference topic.

When adding a Mercurial trigger you are required to supply the following details:

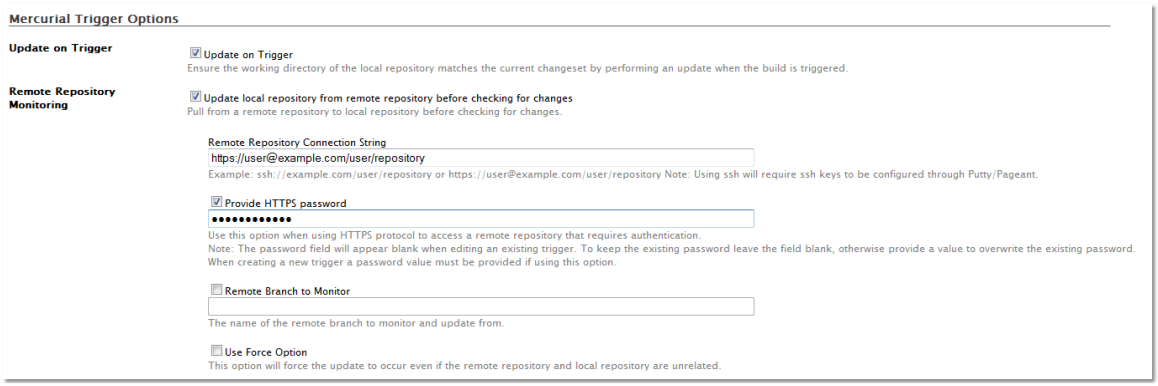
Repository to monitor - This is the location of the Mercurial repository that you want to monitor for changes. This must be a local repository (local directory, mapped drive or network share).



Mercurial executable - This is the location of the Mercurial executable file on the local machine.



Mercurial Trigger Options



Update on Trigger

- **Update on Trigger** - This perform an update on the local repository when the build is triggered to ensure that the working directory is up to date.

Remote Repository Monitoring

- **Update local repository from remote repository before checking for changes** - Enabling this option, will enable the fields below so that you can specify the details of the repository you wish to monitor.
- **Remote Repository Connection String** - Provide the connection string of the remote repository that you want to monitor. This can be in any of the following formats:
  - http://example.com/user/repository - this is read only and requires no authentication.
  - https://user@example.com/user/repository - will generally require authentication either via *Provide HTTPS password* option (see below) or by configuring the [auth] section in the Mercurial.ini file to store the password for URL being used.
  - ssh://example.com/user/repository - using SSH will require public/private key authentication, this will need to be configured using Putty/Pageant and also require the *ssh* node to be configured under the [ui] section of the Mercurial.ini file.

  Click here for more information about configuring the Mercurial.ini file.
- **Provide HTTPS password** - This option enables you to specify the HTTPS password to be used with a HTTPS connection string. If you specify a HTTPS connection string that requires authentication without providing a password the trigger will throw an error. An alternative to using this option is to configure the [auth] section of the Mercurial.ini file which will allow you to use a HTTPS connection string without being prompted for a password. *Note: As a security measure, the password field will be blank when editing an existing trigger that already has a password defined. In this case leaving the option enabled and the field blank will retain the existing password, entering a new value will override the existing password. Enabling the option and leaving the field blank when creating a new trigger will result in a validation exception.*
- **Remote Branch to Monitor** - Provide the name of a specific branch to monitor and pull changes from. Changes occurring on other branches will be ignored.
- **Use Force Option** - By default the Mercurial will not allow to pull changes from a source repository that is unrelated to the destination repository. Enabling the Force option overrides this behaviour allowing changes to be pulled from an unrelated source.

Exclude Files From Monitoring

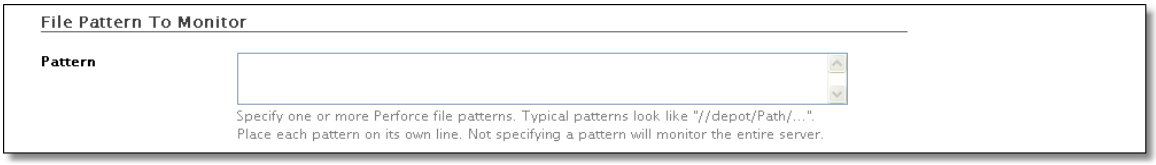
See Excluding Files From Source Control Monitoring

2.1.5.7 Perforce Trigger

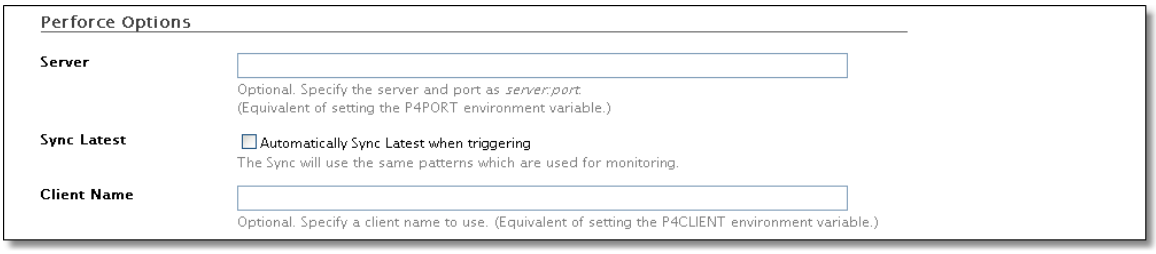
The Perforce Trigger allows you to monitor a Perforce source control repository for changes, and trigger builds whenever new changes are checked in.

General information on creating triggers can be found under the Triggers Reference topic.

The Perforce trigger supports the following options:



This is the Perforce file pattern(s) to monitor. Leave this field blank to monitor the entire Perforce server.



Specify a Perforce "Server" name to override the default Perforce server name.

Enable "Sync Latest" if you would like to get the latest source to a working directory before building. The working directory to use will be determined based on the client.

Specify a "Client Name" to override the default Perforce client name.

Optionally, specify credentials for the Perforce server.

Specify the path to the p4.exe executable on the server. This path should be autodetected if Perforce is installed on the server.

Exclude Files From Monitoring

See Excluding Files From Source Control Monitoring.

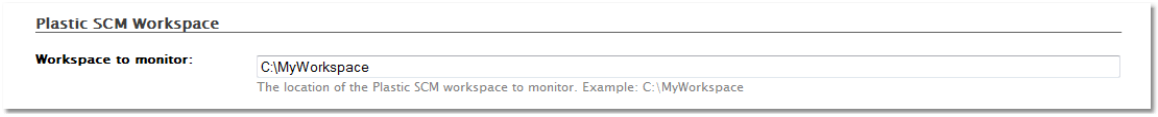
2.1.5.8 Plastic SCM Trigger

The Plastic SCM Trigger allows you to monitor a workspace for changes. A build is triggered when a file is checked in to the repository that the monitored workspace is pointing to.

General information on creating triggers can be found under the Triggers Reference topic.

When adding a Plastic SCM Trigger the following details are required:

Plastic SCM Workspace

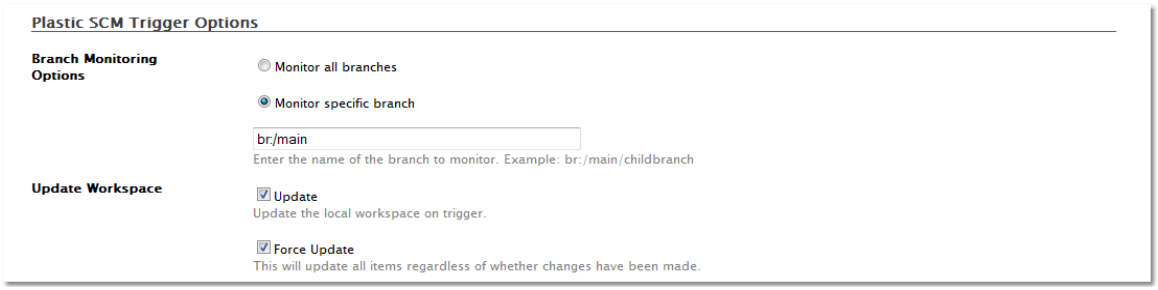


Workspace to monitor - Provide the location of Plastic SCM workspace to monitor. This workspace will need to be configured to point to a repository server where changes are being made in order to trigger a build.

Plastic SCM Client Executable

Path to client executable - Specify the location of the Plastic SCM Client Executable (cm.exe) on the local machine.

Plastic SCM Trigger Options



Branch Monitoring Options

- **Monitor all branches** - Monitor all the branches under the workspace for changes.
- **Monitor specific branch** - Only changes that occur within the specified branch will trigger a build.

Update Workspace

- **Update** - Update the local workspace from the repository if changes have been detected.
- **Force** - By default the update command will only update items that have been modified. The forced operation will create a new revision of all items regardless of whether a change has occurred or not.

Exclude Files From Monitoring

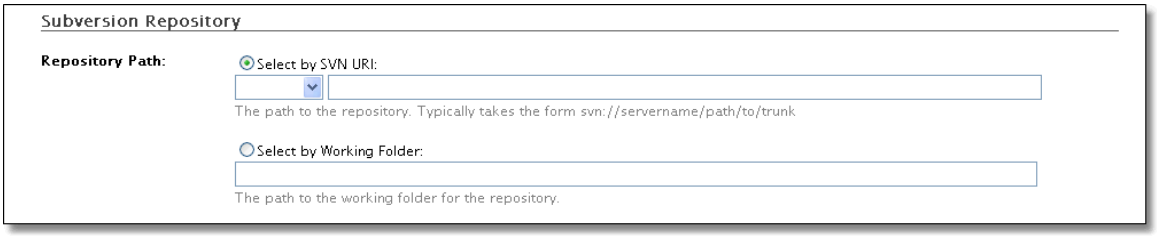
See Excluding Files From Source Control Monitoring

2.1.5.9 Subversion Trigger

The Subversion Trigger type allows you to monitor a Subversion source control repository and automatically start builds when changes are checked in.

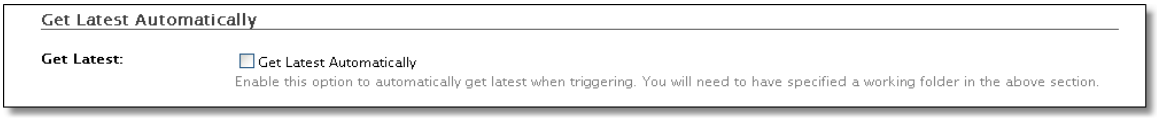
General information on creating triggers can be found under the Triggers Reference topic.

The Subversion Trigger Type contains the following options:



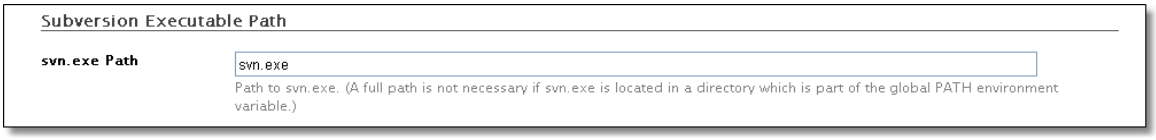
You can choose to select a repository either from it's Subversion URI path, or based on the current working directory for the repository.

You must specify at least one of these two items.



Enable this checkbox if you want to check out a current copy of the source to the specified working directory. Even if you chose "Select by SVN URI", above, then you will still need to put in a working directory if you want to use this option.

Credentials for the Subversion repository. Check "Update Password" to save the contents of the Password field.



Specify the path to svn.exe. This path cannot be autodetected, so you will need to know the path to the subversion installation on your computer.

If svn.exe is located in the system path (ie you can type "svn" at a command prompt to run svn.exe) then you can just enter "svn.exe" as the path.

Exclude Files From Monitoring

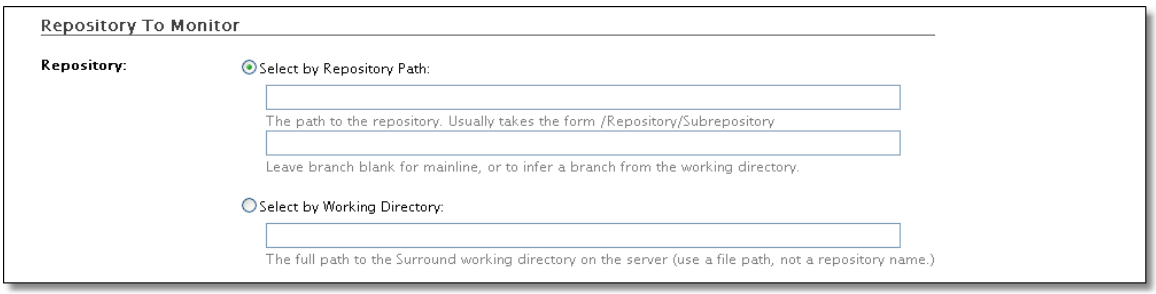
See Excluding Files From Source Control Monitoring.

2.1.5.10 Surround SCM Trigger

The Surround SCM Trigger type allows you to monitor a Seapine Surround SCM source control repository and automatically start builds when changes are checked in.

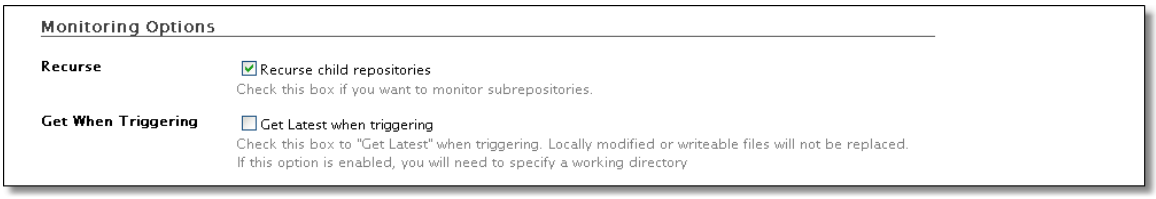
General information on creating triggers can be found under the Triggers Reference topic.

The Surround SCM Trigger type supports the following options:



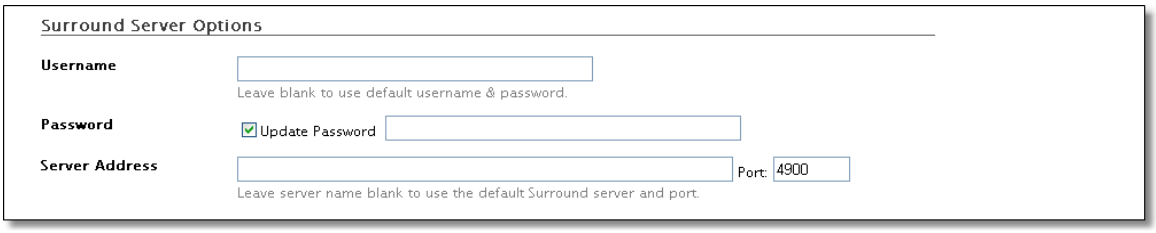
You can choose to select the repository to monitor either based either on the repository path and branch name, or based on the path to a working directory of that repository on the server.

You must provide at least one of these two sets of values.



Select "Recurse" in order to monitor all repositories and directories under the target. If this box is not checked, only the top-level repository will be monitored.

Select "Get Latest when triggering" in order to get a copy of the latest source to the working directory when the trigger runs. You will need to have specified a working directory (see above) for this to work.



The "Server Options" let you override the defaults for the installed version of Surround. Optionally specify a username, password, and server name.

To save the password, the "Update password" checkbox must be selected.

The path to the Surround SCM installation directory. This path should be automatically filled in if Surround is installed.

Exclude Files From Monitoring

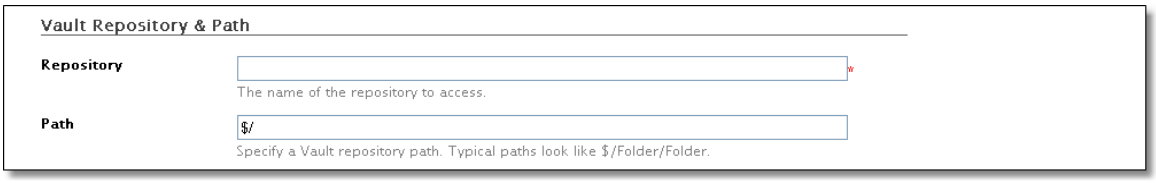
See Excluding Files From Source Control Monitoring.

2.1.5.11 Vault Trigger

The Vault Trigger allows you to monitor a Sourcegear Vault source control repository for changes, and trigger builds whenever new changes are checked in. The Vault Trigger requires Sourcegear Vault 4, or newer.

General information on creating triggers can be found under the Triggers Reference topic.

The Vault trigger supports the following options:



Specify the name of a Vault Repository on the server (ie "Default Repository"), and a Vault repository path to monitor.

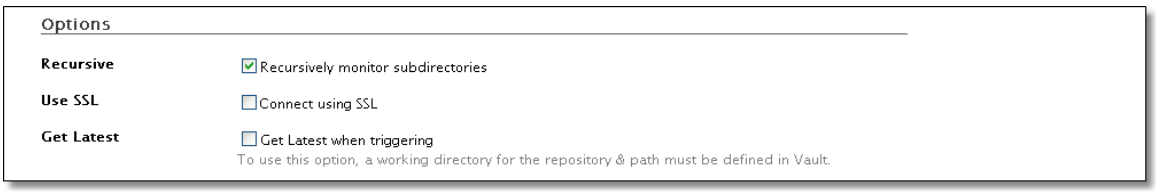
The Repository name is required, unless it has already been saved outside of FBServer by using the "Vault Rememberlogin" command.

The Path is required.

Specify the name of a Vault server to connect to, and a username and password for the connection.

The Server, Username & Password are required, unless they have already been saved outside of FBServer by using the "Vault Rememberlogin" command.

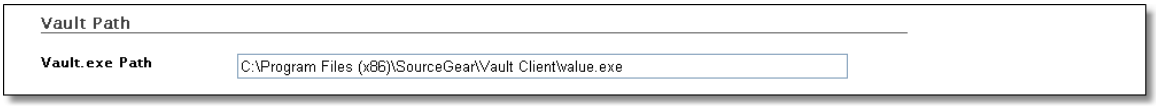
The password will not be saved unless the "Update password" checkbox is checked.



Select "Recursively monitor subdirectories" to monitor all directories under the target repository path.

Select "Connect using SSL" to connect to the Vault server using SSL.

Select "Get Latest when triggering" to get the latest copy of the source to the working directory defined in the Vault Client on the server.



Specify the full path to vault.exe on the server. This path should be automatically

detected if Vault is installed on the server.

Exclude Files From Monitoring

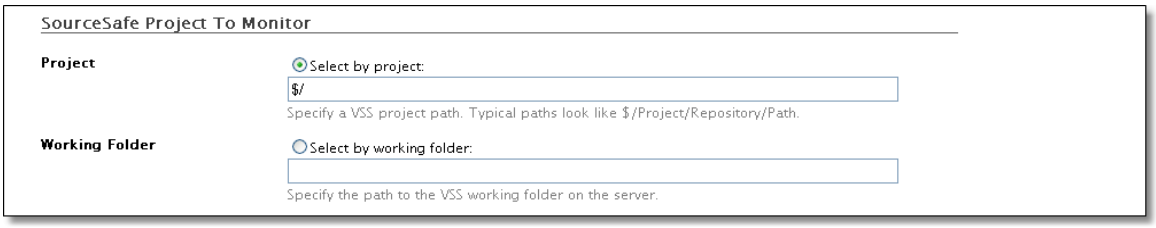
See Excluding Files From Source Control Monitoring.

2.1.5.12 Visual SourceSafe Trigger

The Visual SourceSafe Trigger allows you to monitor a Microsoft Visual SourceSafe repository and trigger builds when changes are checked in. FinalBuilder Server supports SourceSafe version 6.0 and SourceSafe 2005.

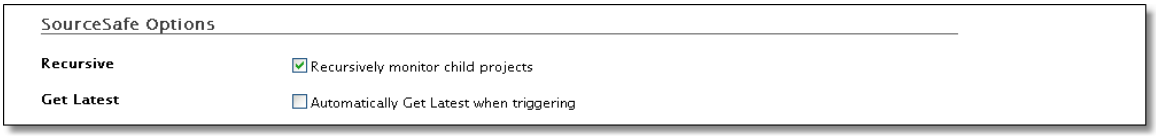
General information on creating triggers can be found under the Triggers Reference topic.

The Visual SourceSafe Trigger supports the following options:



The project to monitor can be selected in one of two ways. Selecting by project works by specifying a VSS project path.

Alternatively, specify a Working Folder which has already been defined in VSS on the server.

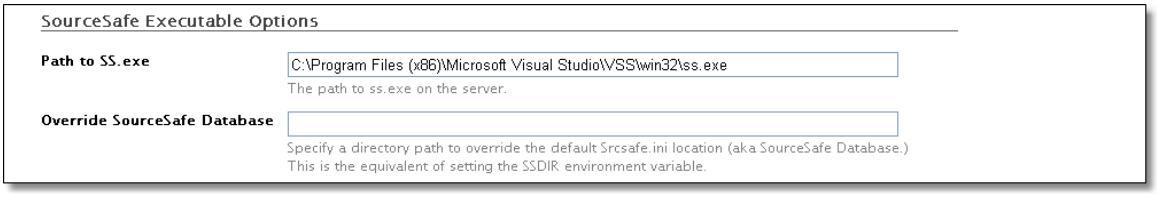


Select "Recursive" to monitor all child projects/directories under the target project.

Select "Get Latest" to automatically get the latest copy of the source to the working folder when triggering.

Specify credentials for the SourceSafe project. These parameters are required.

Changes made in the Password field will not be saved unless "Update password" is checked.



Specify the path to the ss.exe executable on the server. This path should be automatically detected if SourceSafe is installed.

"Override SourceSafe Database" allows you to use a non-standard Srcsafe.ini file for the SourceSafe Database. Specify the directory in which the file is stored. This is the equivalent of setting the SSDIR environment variable.

Exclude Files From Monitoring

See Excluding Files From Source Control Monitoring.

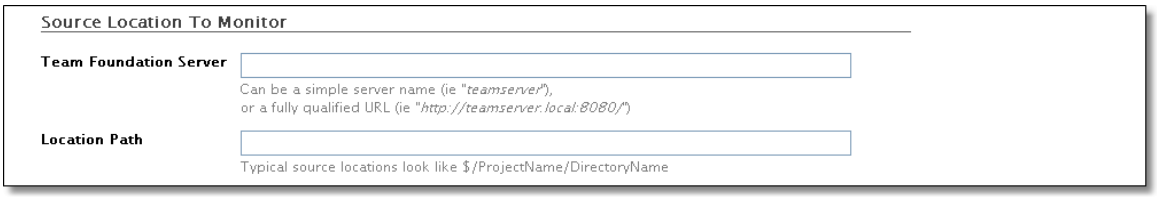
Note: With VSS and Exclude Files monitoring, full repository paths are only returned when a file is checked in. For adds and deletes, only the file name is provided. This is a limitation of VSS.

2.1.5.13 Visual Studio Team System Trigger

The Visual Studio Team System Trigger allows you to monitor a Visual Studio 2005 Team Foundation Server source control repository for changes, and trigger builds whenever new changes are checked in.

General information on creating triggers can be found under the Triggers Reference topic.

The Team System Trigger supports the following options:

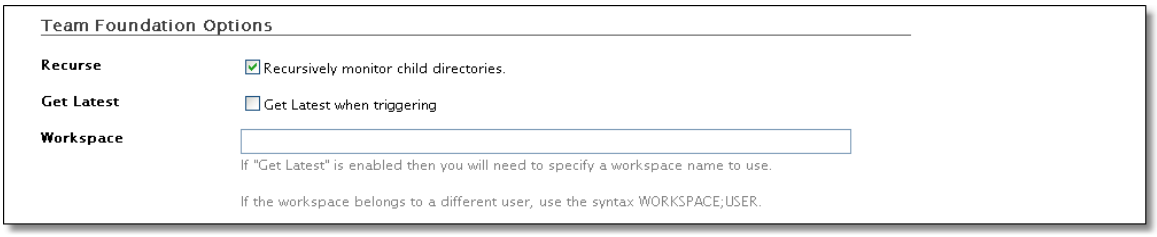


Specify the Team Foundation Server to monitor, and the project path to monitor. You can obtain the Location Path by looking at the status bar in Visual Studio Source Explorer.

Both of these parameters are required.

Specify the username and password for the Team Foundation connection. These parameters are required.

Changes to the password will not be saved unless the "Update password" checkbox is checked.



Enable "Recursively monitor child directories" to monitor project directories underneath the target path.

Enable "Get Latest" when triggering to get the latest version of the source to the working directory of a specified Workspace. You will need to specify the name of a Workspace.

You do not need to specify a Workspace name if you are not using "Get Latest."

Note that if you want to create a temporary Workspace for the build, get the source to the new Workspace, and then remove the workspace, then you can do so from inside the FinalBuilder project, by using the Team Foundation Create/Delete Workspace actions and the Team Foundation Get action.

Exclude Files From Monitoring

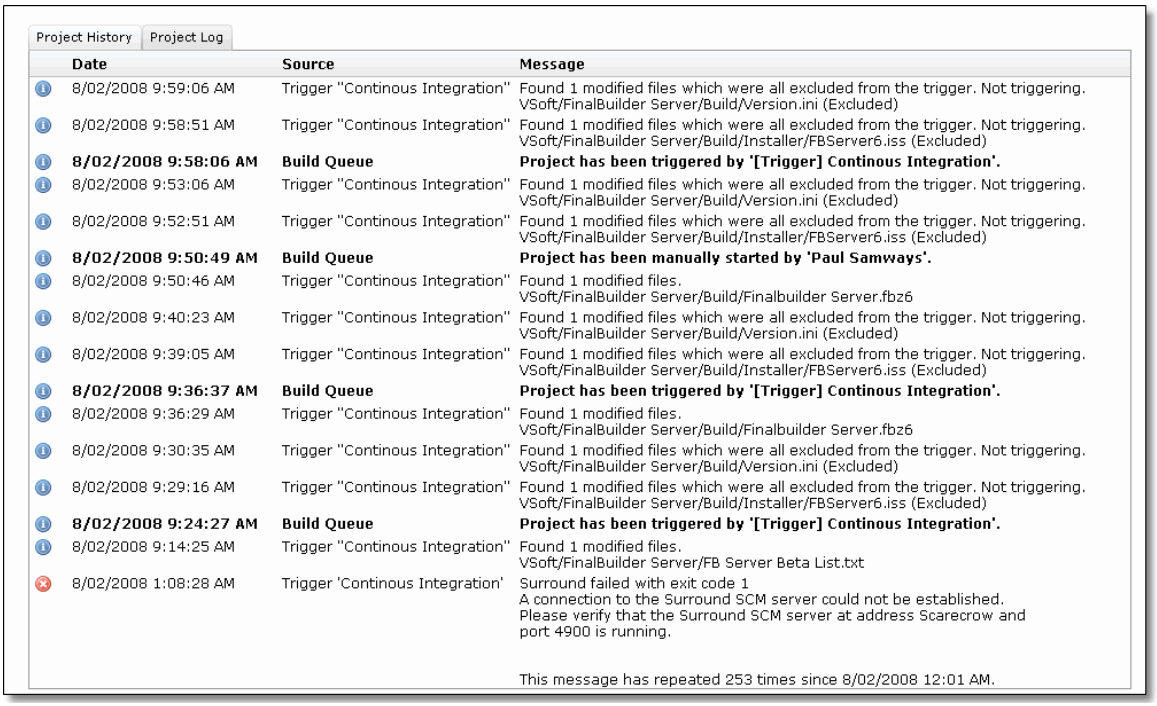
See Excluding Files From Source Control Monitoring.

2.2 Project Trigger Log

Each Build Project contains a Project Trigger Log which provides a history of trigger activity. Log entries are added when builds are triggered, or when triggers encounter errors.

To view the Project Trigger Log, navigate to the Build History page for the project, and then click on the Project Trigger Log tab.

Example Project Trigger Log



Project logs are persisted to disk and can be found in C:\ProgramData\VSoft\FBServer7\Logging\Projects

2.3 Trigger Errors

Sometimes, Triggers can fail due to error conditions.

When this happens, three things happen:

1. The error details are logged in the Project Trigger Log for the project.
2. The error details are shown on the Edit Trigger page for the action.
3. The build status and Project Edit pages are updated to notify the user that an error occurred in the trigger.

If you see that a trigger has encountered errors, edit the Trigger to see the full error message, then take action to correct the problem.

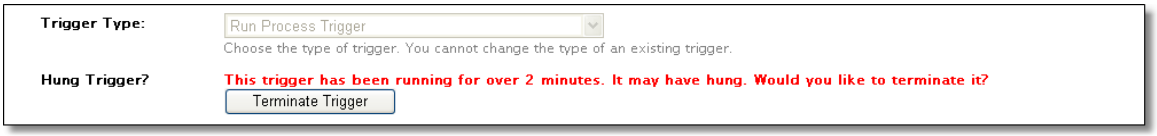
If the trigger needs to be taken offline, uncheck the "Active" box on the Trigger edit page and click Save.

If a trigger has an error, it will keep running. If the error goes away, the error details will disappear from the Edit Trigger and Build Status pages, but the error remains in the Project Trigger Log.

Hung Triggers

If a trigger runs a child process and the process runs for more than one minute, FinalBuilder Server will give you the option to terminate it.

When you load the Edit Trigger page for a long-running trigger, you will see an option like this:

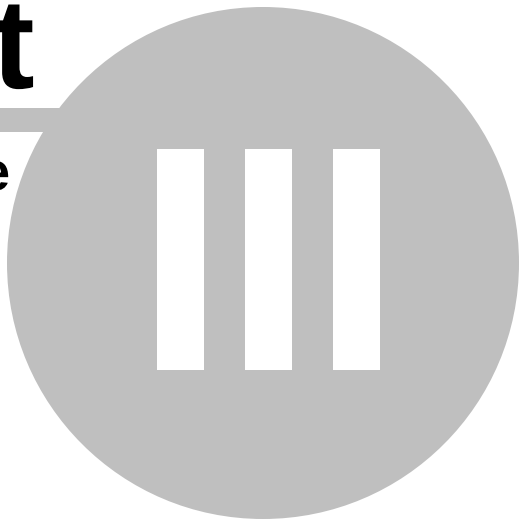


Click the "Terminate Trigger" button to terminate the trigger process if you think it may have hung.

The terminated trigger will be automatically de-activated (you will need to activate it again to continue.)

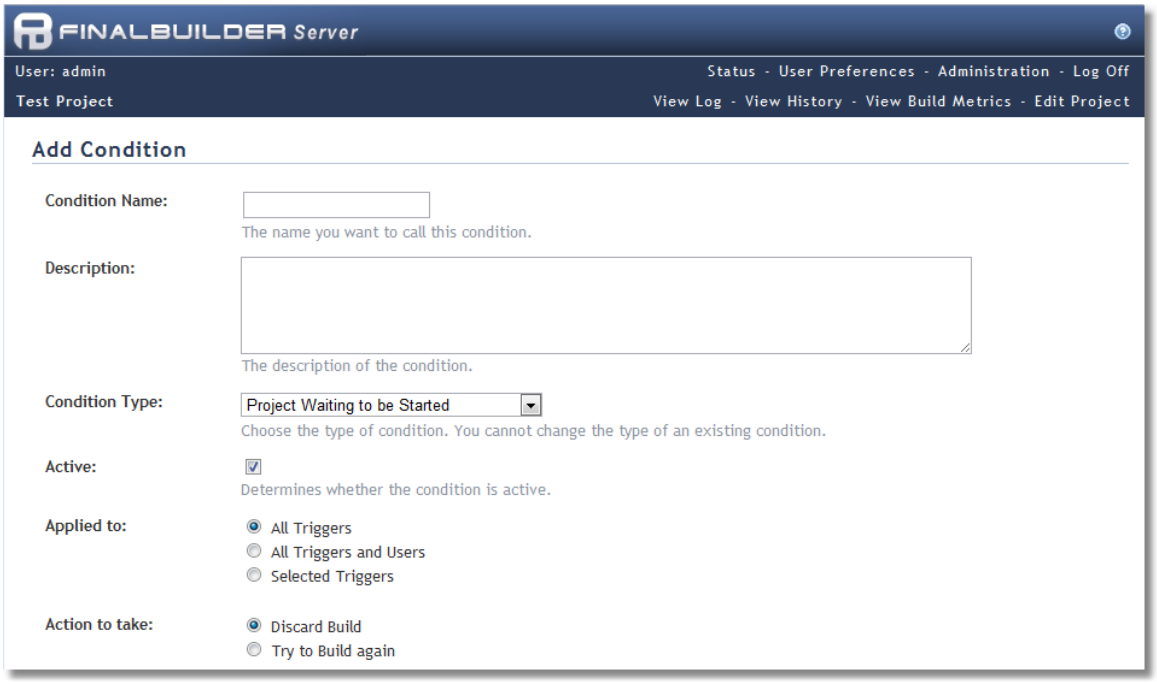
Part

Trigger Condition Reference



3 Trigger Condition Reference

Trigger conditions are used to determine whether a trigger should start the build as soon as the project is triggered or whether it should then have to pass a set of "conditions" before starting.



Condition Name
The name you wish to call this condition.

Description
The description you want to give this condition. Can be used to make it clear why you have this condition set.

Condition Type
The type of condition to add to the project. Once you have added the condition, you cannot change this value.

Active
Determines whether this condition is currently active.

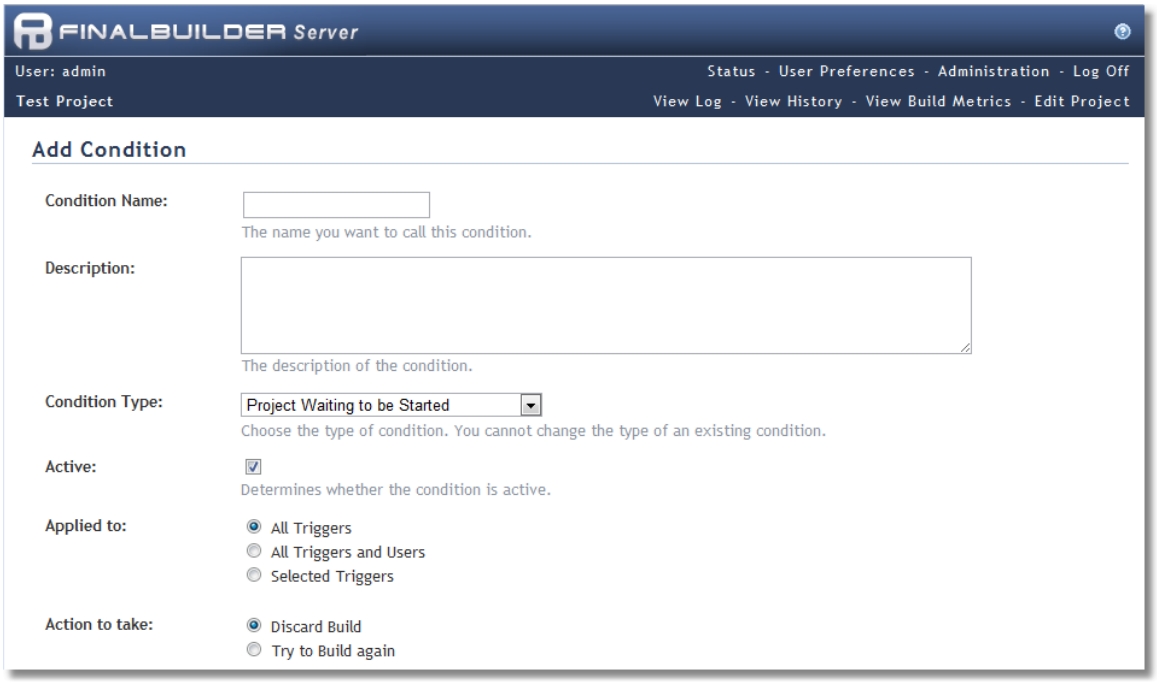
Applied To
A trigger condition can be applied to all triggers, meaning that when any trigger tries to start a build this condition will be evaluated. They can be applied to all triggers and users, which means that whenever a build is started this condition will be evaluated (with the exception of a user by-passing the build queue when starting the build). You can also choose to only evaluate this condition when the build was started by a specific trigger.

If Condition not met
This is the action the build server will take when a build is triggered and at least one of

the conditions was not met. Discarding the build will cause the build server to forget that the build was ever triggered, where as setting the server to try again will cause the build to be started at a later time. When the build is set to start at a later time, the trigger conditions are re-evaluated before the build is started.

3.1 Project Waiting to be Started

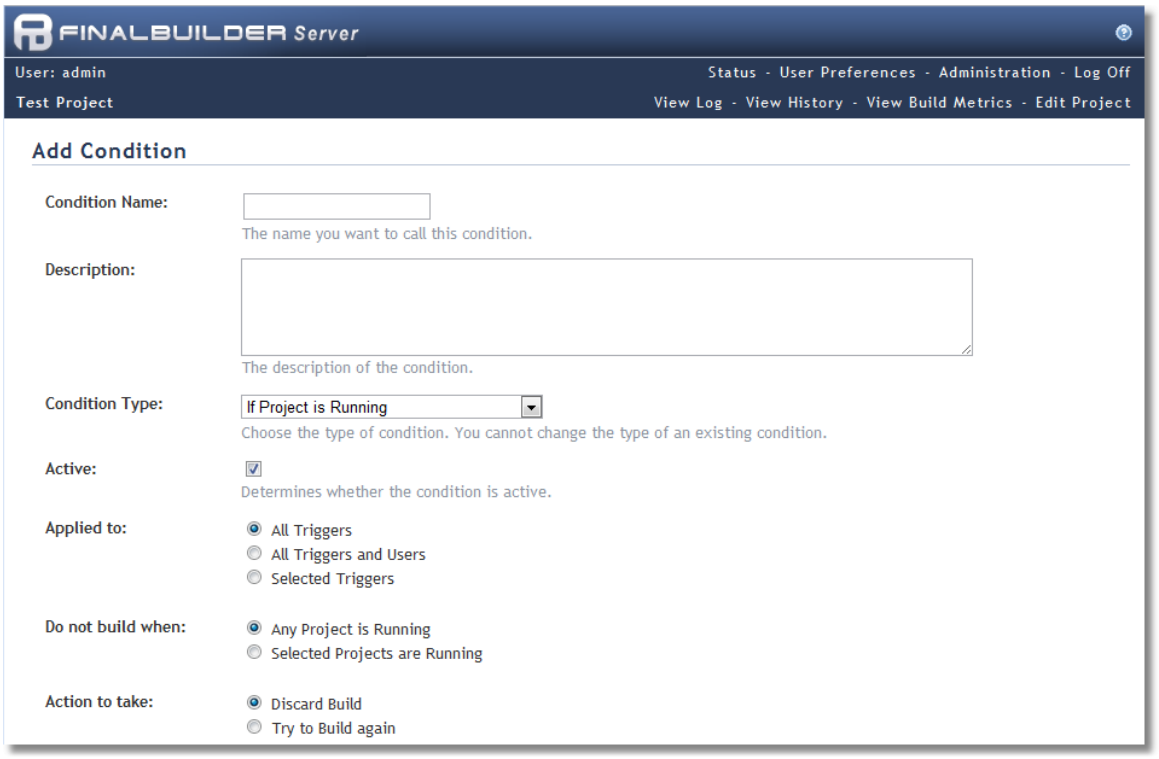
The 'Project Waiting to be Started' trigger condition will prevent a build from starting if it has previously been triggered, but has not yet started because one of its trigger conditions has not been met.



Tips
This condition would typically be used to prevent multiple builds of the same project from being on the build queue at the same time.

3.2 If Project is Running

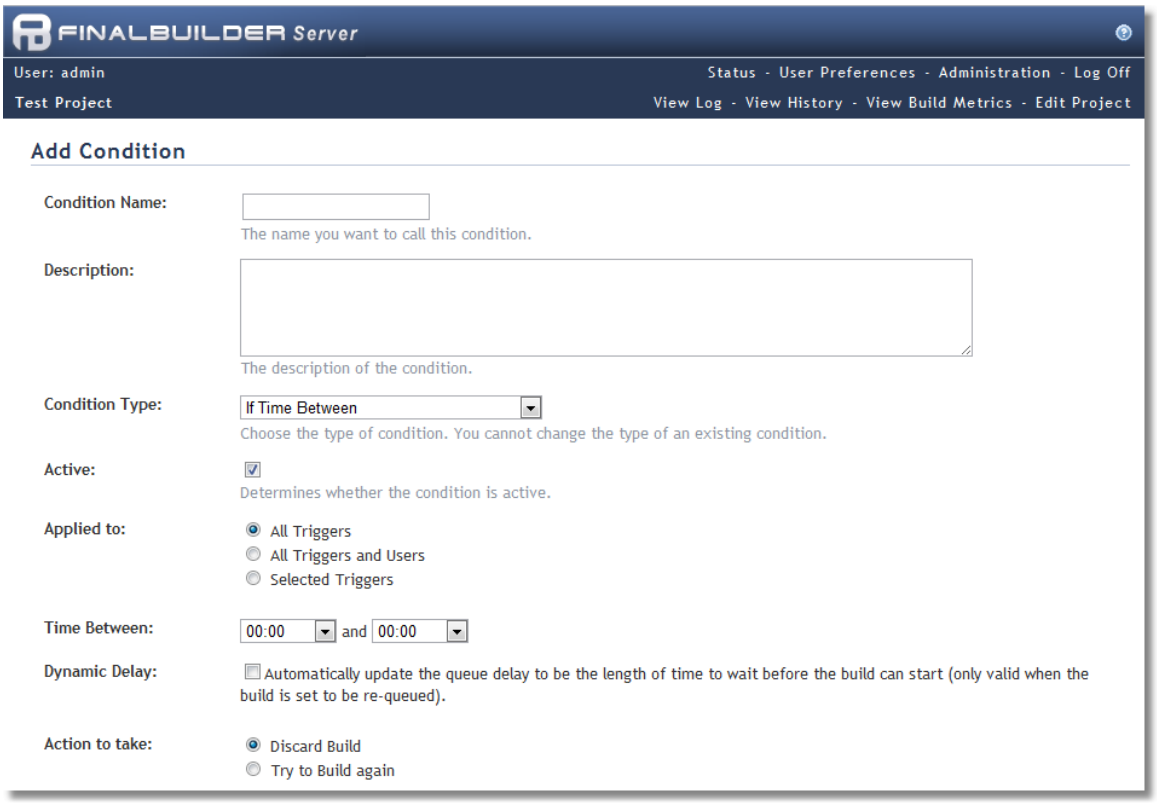
The 'If Project is Running' condition can be applied to a project when you require that a project does not run at the same time as either any project, or just the selected project.



Tips
When you require that a project does not run at the same time as another project you will need to ensure that this condition is applied to both projects.

3.3 If Time Between

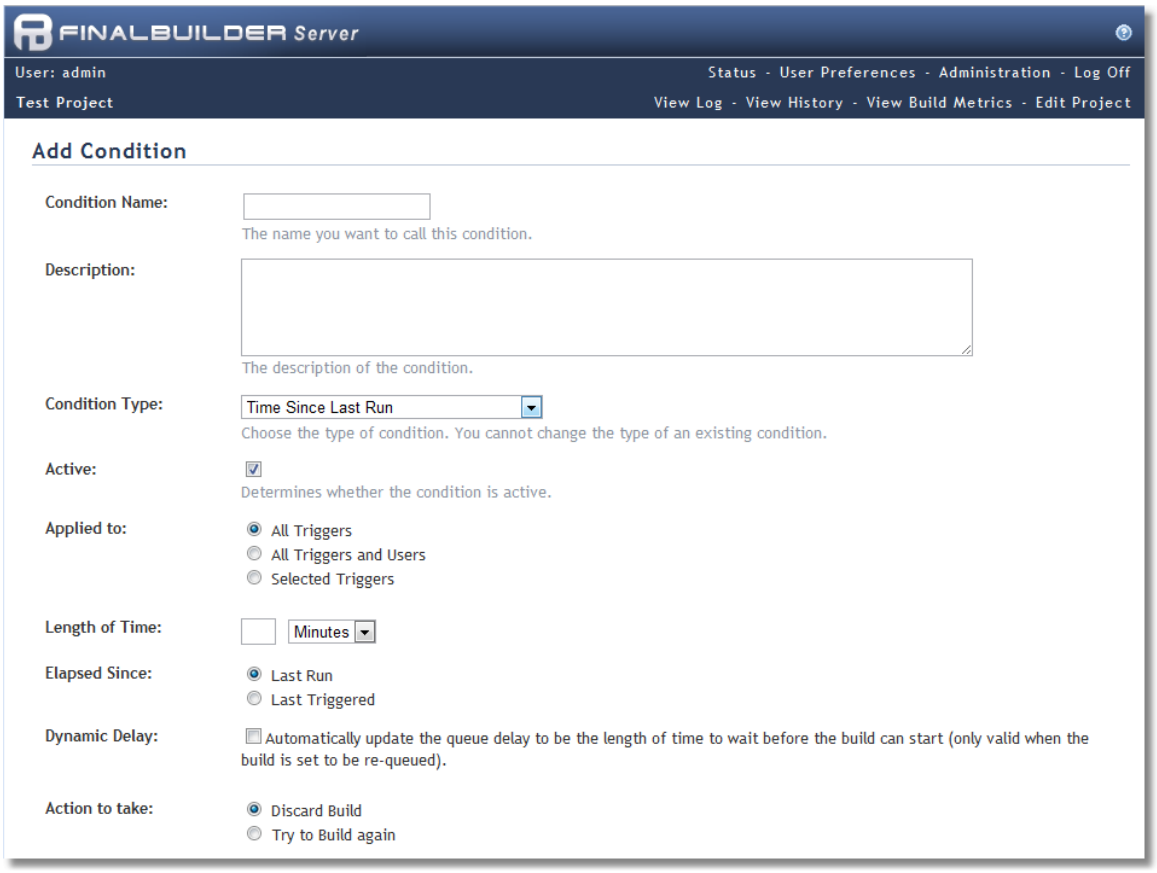
The 'If Time Between' trigger condition can be used to prevent a build from being started when the time is between a specified time.



Time Between
The times between which a build should not be started.

3.4 Time Since Last Run

The 'Time Since Last Run' trigger condition prevents a build from starting when a specified length of time has not past since the project was either last run or last triggered by a specific trigger.

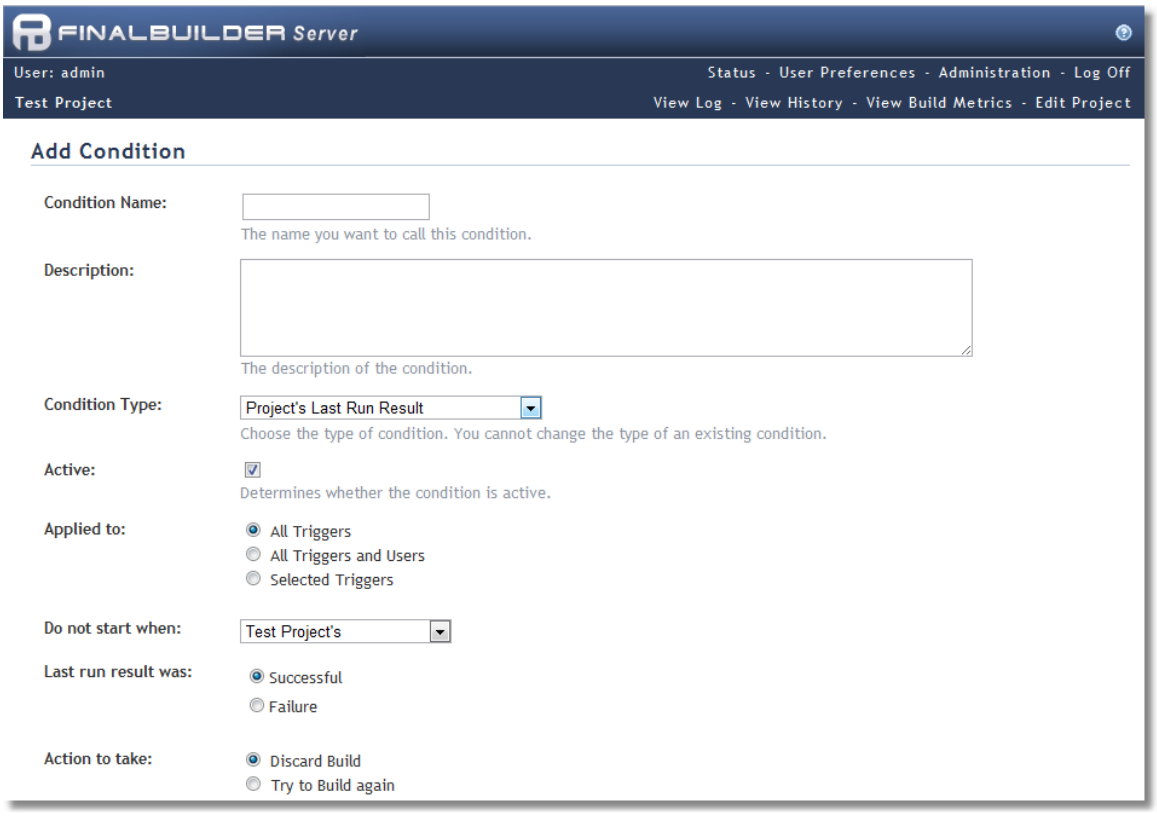


Length of Time
The length of time that should of elapsed since the project was last run or triggered.

Elapsed Since
Determines whether the condition should look at the last time the project was run (manually or by a trigger) or the last time it was triggered by a specific trigger.

3.5 Project's Last Run Result

The 'Project's Last Run Result' trigger condition prevents a build from starting when the specified project's last run result is either a success or failure.



Do not start when
The project who's last run result is checked to determine whether the project should be triggered.

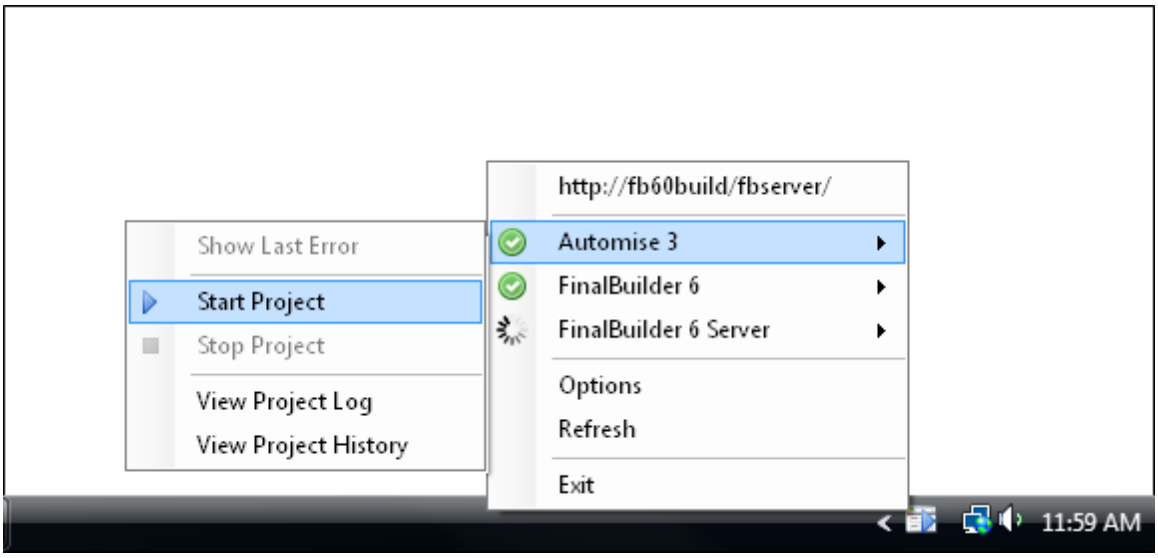
Last run result was
The result that the specified project needs to have for the trigger condition not to be met.

Part IV

Notification Application

4 Notification Application

The FinalBuilder Server Notification Application is a windows application that runs in the users notification area, allowing monitoring and control of multiple build servers. Users can configure to have the notification application show a visual alert, play a sound or execute a custom plug-in when the build completes.



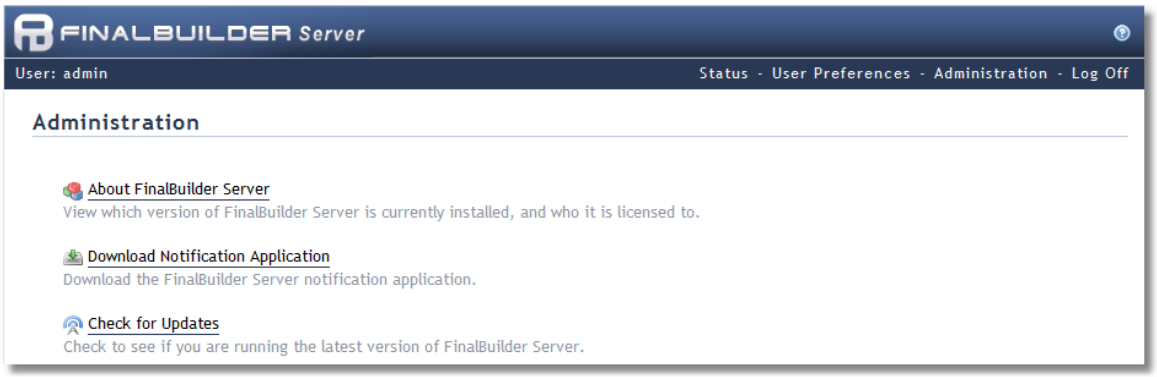
Requirements

- Microsoft Windows
- Microsoft .Net Framework 2.0

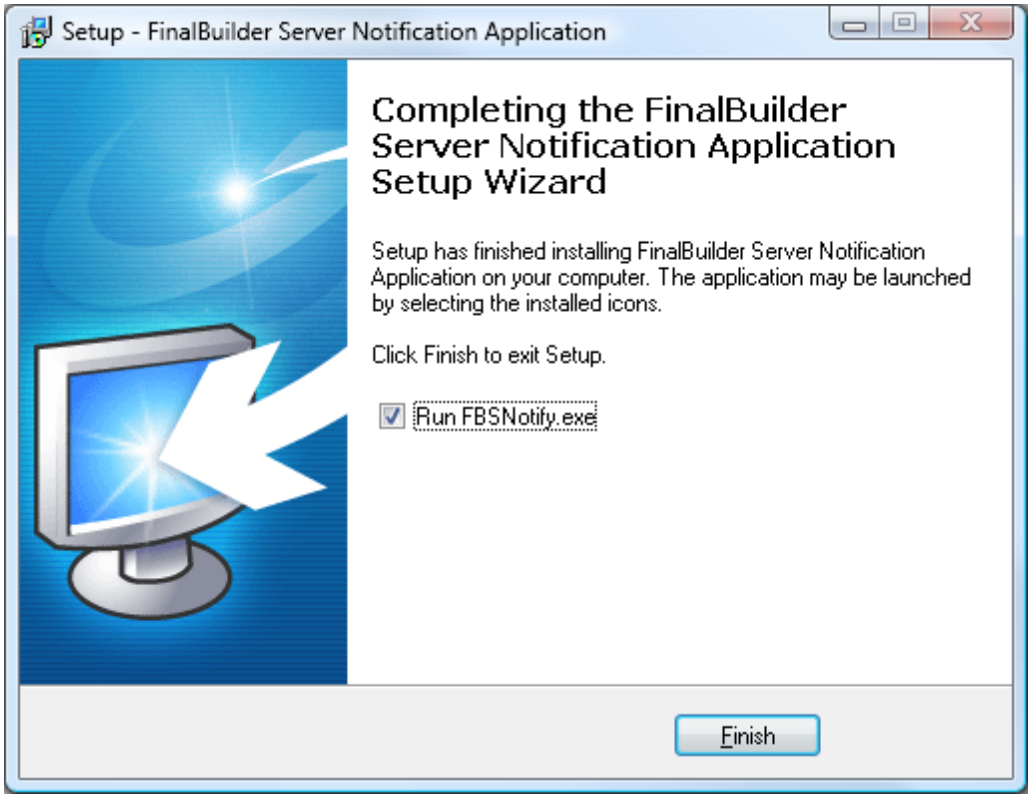
4.1 Installation

The Notification Application's installer is installed as part of the FinalBuilder Server web site.

1. Making sure you are logged into the FinalBuilder Server web interface, navigate to the ' Administration' page and click the 'Download Notification Application' link.



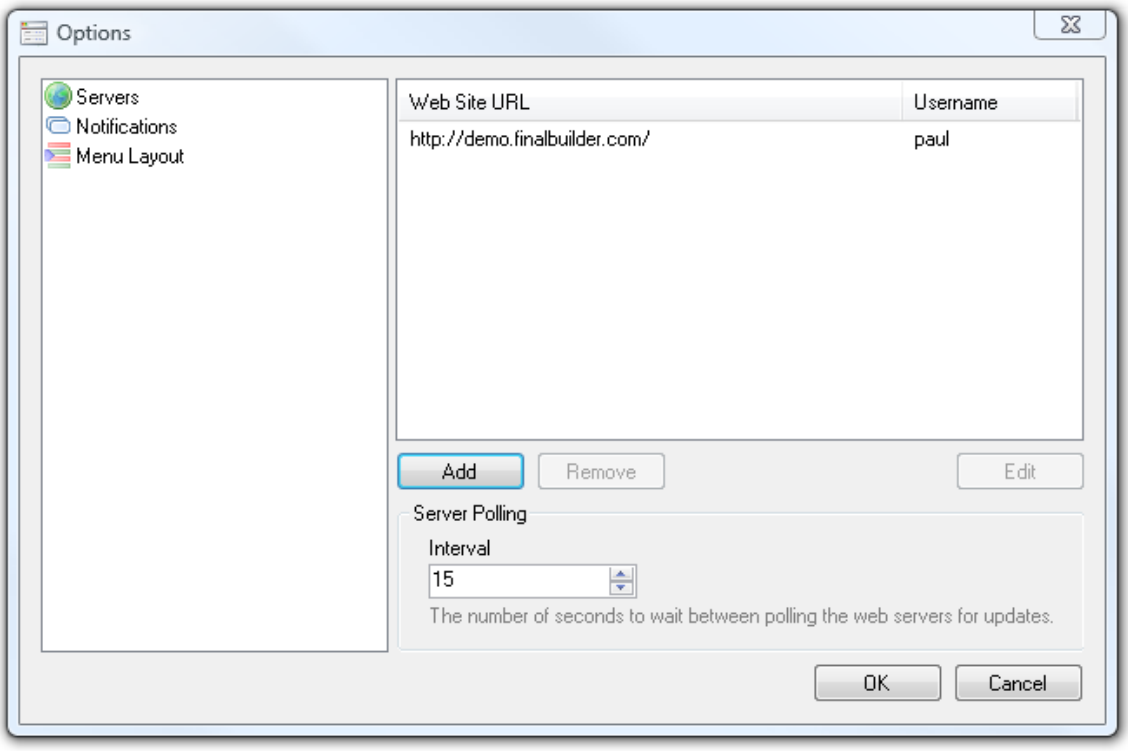
2. When prompted whether to run or download the installer, select run and follow the instructions provided by the installer. Once the installation has completed you will be asked whether you wish to run FBSNotify.exe.



3. The first time the notification application has run you will be presented with the options dialog where you can configure the servers you wish to monitor. See 'Configuration' for more information.

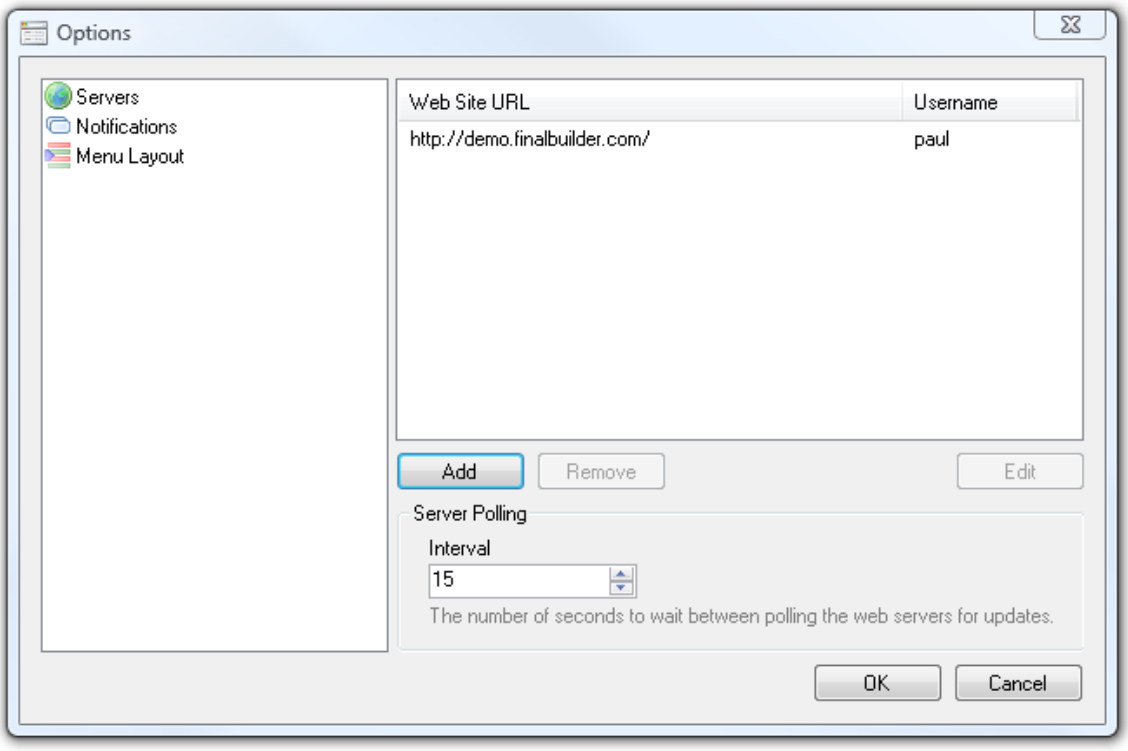
4.2 Configuration

The first time the Notification Application is run, or when no previous configuration can be found, you will be presented with the options dialog. From the options you can configured which servers the notification application is to monitor, how you would like to be notified and how you want the applications context menu to be layered out.



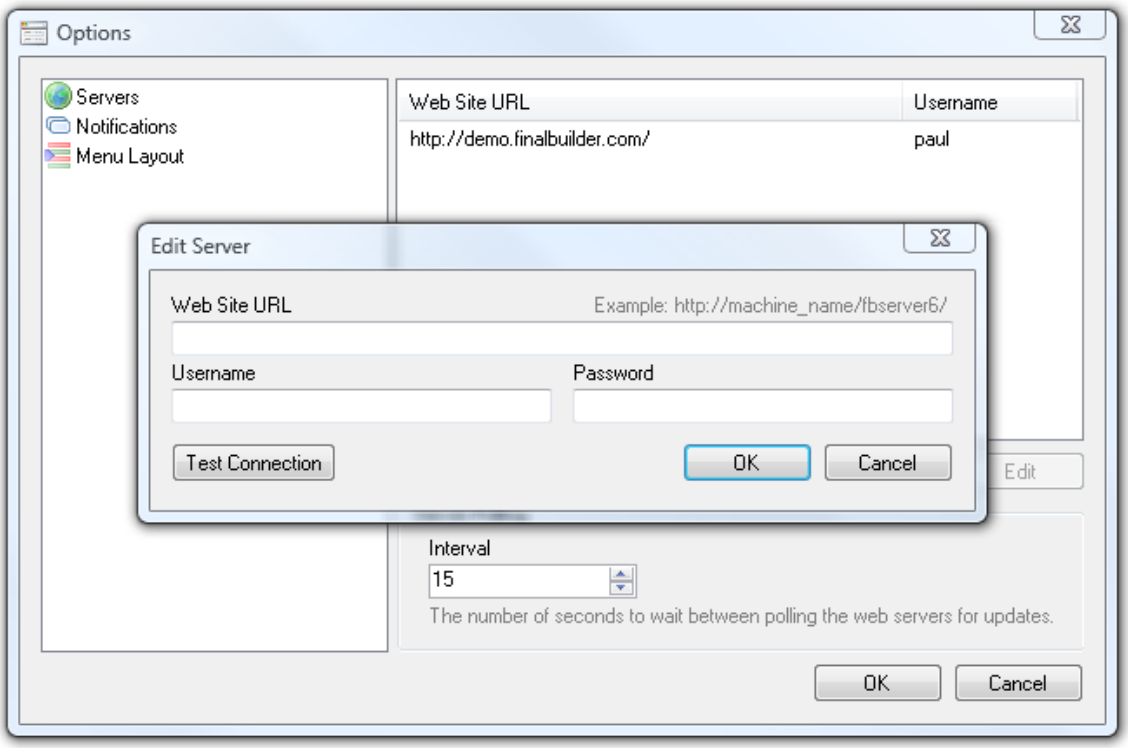
4.2.1 Web Servers

The FinalBuilder Server Notification Application enables you to monitor and control multiple FinalBuilder Server installations.



Adding a new server

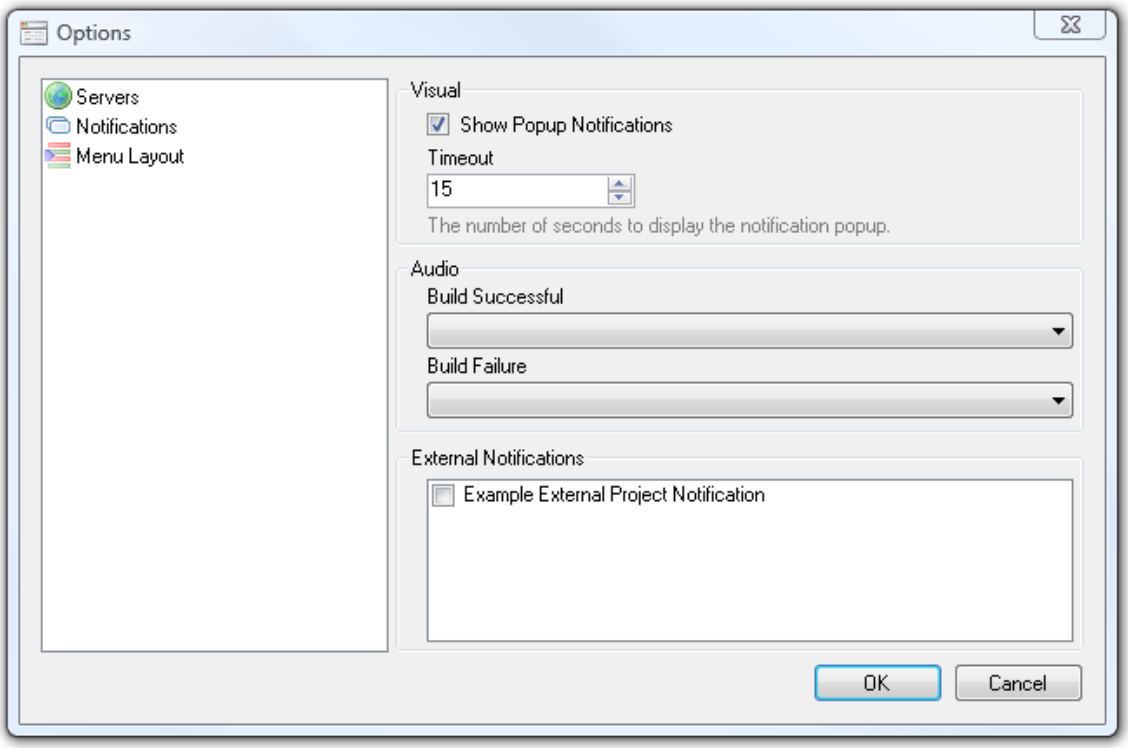
1. On the 'Servers' configuration page click on the 'Add' button to open the add server dialog.



2. Type in the FinalBuilder Server web site address (for example: http://machine_name/fbserver7/) and the username and password you use to login to the build server.
3. Press 'Test Connection' to make sure that the entered details are correct and that the web server can be contacted.
4. Once the connection has been tested, close the dialog by clicking 'OK'.

4.2.2 Notifications

You can configure how you would like to be notified of completed builds on the 'Notifications' page of the options dialog.



Visual
Visual notifications are displayed in the lower right corner of the primary screen and let you know which build finished, and whether it completed successfully. You can configure how long you want the notification to be shown by adjusting the 'Timeout' setting.

Audio
Audio notifications are played when a build finishes. The audio files must be wave audio files and can either be one of the files located in Windows\Media folder, or you can browse for a wave file by selecting the browse item.

External Notifications
You can further extend the notifications offered by the notification application by writing your own notification extensions. See 'Extensions' for more information.

4.2.2.1 Extensions

The FinalBuilder Server Notification Application allows you to extend the notifications by allowing you to create your own notification extensions in managed code.

Creating an extension

1. Create a new class that implements IProjectNotificationExtension and is optionally decorated with ExtensionDescriptionAttribute. These classes can be found in the FinalBuilderServer.Notification.Extension.dll assembly located in the notification application's program files directory.
2. Compile the assembly and place it in the 'Extensions' directory located in the Notification Applications program files directory.

3. Restart the Notification Application.
4. On the 'Notifications' tab of the options dialog your new extension will be listed in the external extensions list, enable it by checking the checkbox.

C# Example

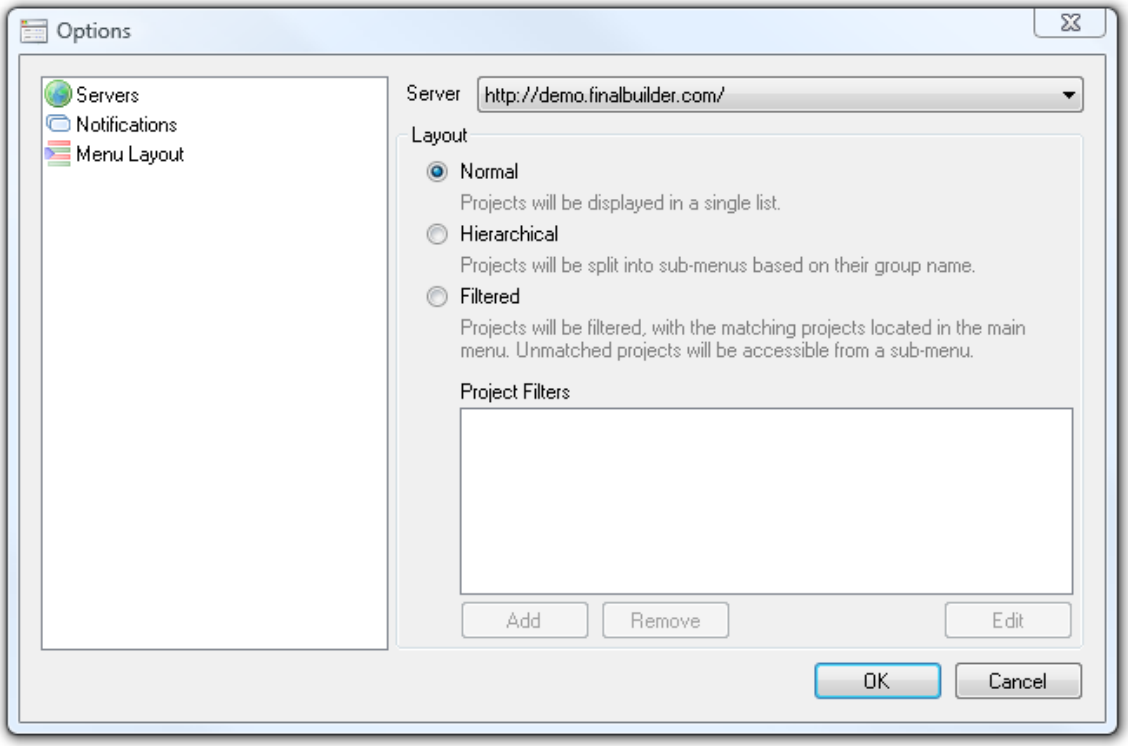
```
[ExtensionDescription("Example External Project Notification", "An example notification extension")]
public class ExtensionExample : IProjectNotificationExtension
{

   public void OnProjectSucceeded(string serverURL, string projectName)
   {
      MessageBox.Show(
         string.Format("{0} on {1} has completed successfully.", projectName, serverURL),
         "Example Notification - Project Successful",
         MessageBoxButtons.OK,
         MessageBoxIcon.Information);
   }

   public void OnProjectFailed(string serverURL, string projectName)
   {
      MessageBox.Show(
         string.Format("{0} on {1} has failed to complete successfully.", projectName, serverURL),
         "Example Notification - Project Failure",
         MessageBoxButtons.OK,
         MessageBoxIcon.Information);
   }

}
```

VB Example

```
<ExtensionDescription("Example External Project Notification", "An example notification extension")>
_
Public Class VBSampleExtension
   Implements IProjectNotificationExtension

   Public Sub OnProjectSucceeded(ByVal serverURL As String, ByVal projectName As String)
Implements IProjectNotificationExtension.OnProjectSucceeded
      MessageBox.Show(String.Format("{0} on {1} has completed successfully.", projectName,
serverURL), "Example Notification - Project Successful", MessageBoxButtons.OK, MessageBoxIcon.
Information)
   End Sub

   Public Sub OnProjectFailed(ByVal serverURL As String, ByVal projectName As String) Implements
IProjectNotificationExtension.OnProjectFailed
      MessageBox.Show(String.Format("{0} on {1} has failed to complete successfully.",
projectName, serverURL), "Example Notification - Project Failure", MessageBoxButtons.OK,
MessageBoxIcon.Information)
   End Sub

End Class
```

4.2.3 Menu Layout

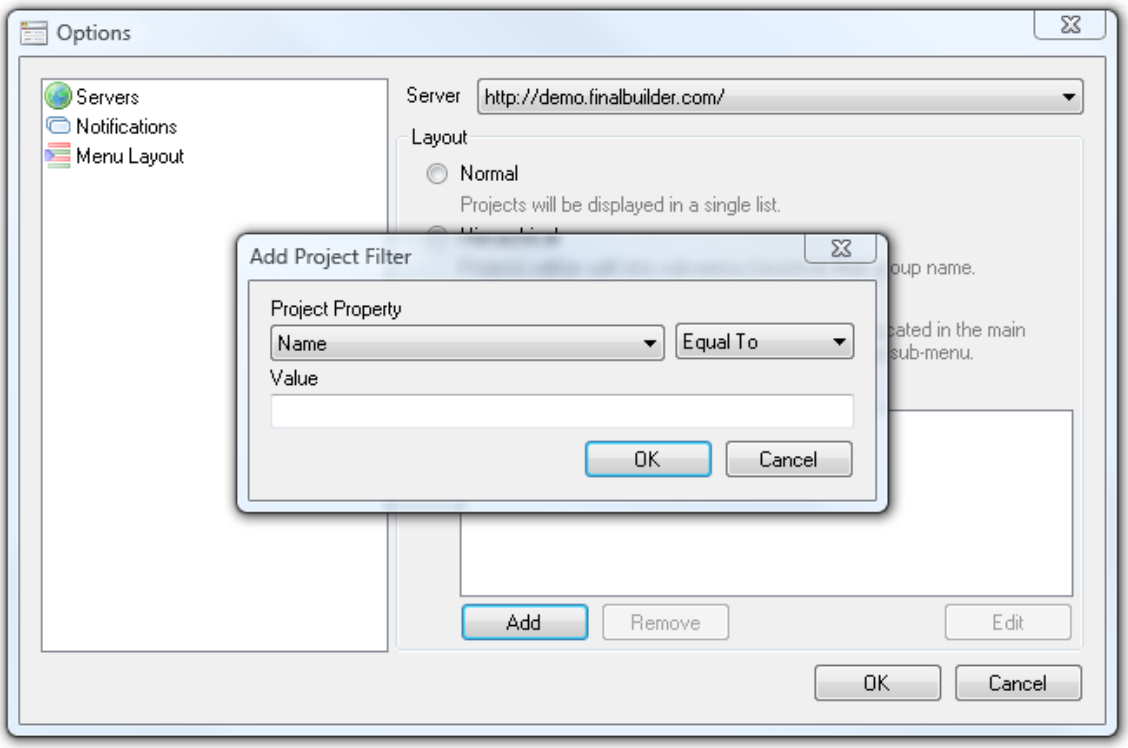
Once you have configured the server's you may choose to change how they are displayed in the context menu. There are three layout modes that can be selected, Normal, Hierarchical and Filtered.



Normal
The normal layout mode will place each project in the main context menu with no grouping.

Hierarchical
The hierarchical layout mode creates a new sub-menu for each group (as defined on the 'Project Edit' page) placing each project in there respective groups sub-menu. Project's that do not belong to any groups will be placed in the main context menu under the group sub-menu's.

Filtered
The filtered layout mode allows you to specify which projects you want to be displayed in the main context menu, with a single sub-menu which contains a hierarchical list of all the projects.



Part V

Product Support

5 Product Support

Forums - http://www.finalbuilder.com/forums.aspx

Email - support@finalbuilder.com

5.1 Installation Issues

FinalBuilder Server works fine on the machine it was installed on, but when you try to view the page from remote machines you receive an error.
Make sure that the firewall on the machine FinalBuilder Server was installed on has been configured to allow incoming traffic on the port which IIS is configured.

When opening FinalBuilder Server you receive a 403 Forbidden error, or all you see is HTML markup.
FinalBuilder Server requires that ASP.Net 2.0 is configured and working correctly on the web server.

You have FinalBuilder Server running on Windows XP and you receive a 403.9 [Access Forbidden: Too many users are connected Internet Information Services].
Windows XP by default will only allow up to 10 users to be connected simutaneosly to IIS. You can increase this limit to 40 by executing the following script from the command line.

"cscript.exe C:\Inetpub\AdminScripts\adsutil.vbs set w3svc/MaxConnections 40"

You are able to view the FinalBuilder Server web interface, but are unable to log in using Internet Explorer.
Make sure that the machine which is hosting the FinalBuilder Server web site does not contain an underscore in its name.

Index

- T -

- U -

- V -