**Store Jenkins Job JAR Files in FTP Server (NAS)**

Features

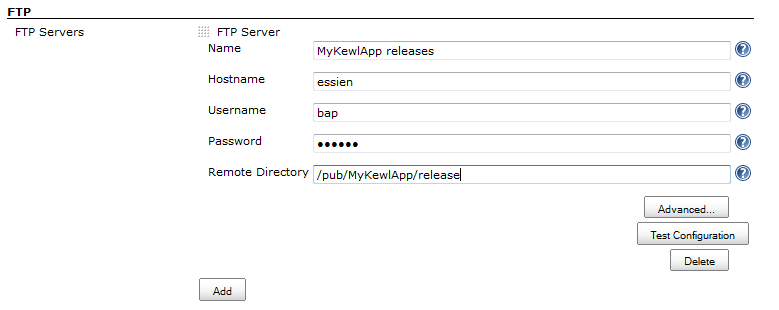
* Send files to FTP servers
* Publish to multiple servers in the "Post-build Actions"
* Transfer complete directory structures, partial structures or just files
* Option to delete all files in the remote directory before publishing
* Use passive (PASV) or active (PORT) transfers
* FTP can be used as a build step during the build process
* Passwords are encrypted in the configuration files and in the UI

Configure

**Create host configurations in the main Jenkins configuration**

From the Jenkins home page, click "Manage Jenkins" and the click on "Configure System"

Find the FTP section (as below) and click on the "Add" button next to "FTP Servers"  
https://wiki.jenkins.io/download/attachments/53608873/ftp_global_unconfigured.PNG?version=1&modificationDate=1297163872000&api=v2

You should now have the configuration options as below  


Fill in Name, Hostname, Username, Password and Remote Directory.

Given below :-

## **Host configuration**

Create one or more configurations that can be selected from the job configuration page.  
To change the order that the configurations appear in the drop down on the job configuration page, drag the square icon (next to FTP/SSH server) to reorder the list.  
As the host configuration specifies how to connect to the remote server, the configurations can be quite specific to the plugin (ie. SSH keys, passive mode FTP, etc), but somme common options are listed below.

When first configuring or changing a configuration, always click the "Test Configuration" to ensure that the configuration will work when it is used from a Job.  
If all is well, then you should see "SUCCESS", otherwise you will see a message in red which should hopefully give some information to help fixing the configuration.

##### Name

Give the configuration a name, this is the name that appears in the drop down box on the job configuration page.  
Having a name for the configuration allows multiple configurations to the same host (to login with different users, or set different remote directories).

##### Hostname

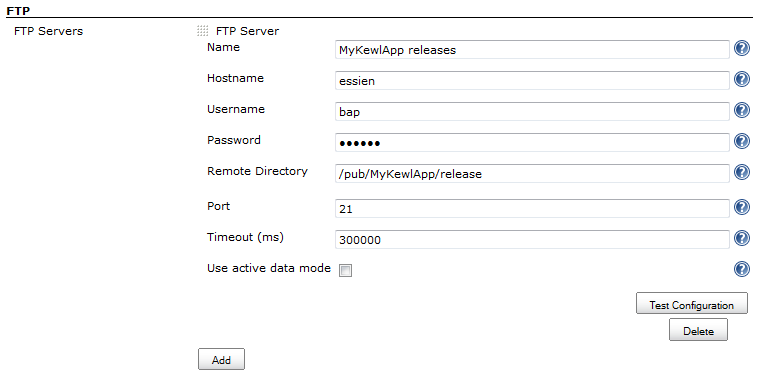
The hostname or IP address of the server. The hostname must be resolvable by any of the machines that may initiate a connection to the remote server.

##### Credentials

Not a field in itself, but the connection will need to know how to authenticate to the remote server.  
The Publish Over FTP Plugin uses Username/Password.  
The Publish Over SSH Plugin can use Username/Password, or SSH keys to authenticate when loging in as Username.

**Advanced**

If you click the "Advanced..." button for a configuration, then you will make more configuration options available (see below)



**Use active data mode :**

By default passive mode will be used to connect to the FTP Server.  
In passive mode the client initiates all connections which will allow the client to send files through a firewall that has been configured to refuse all inbound connections (and does not create dynamic rules).  
If you want to dissable passive mode, and have the client use PORT commands for the server to connect to the client to retieve the files, then you can check this box.

**Don't make nested dirs :**

Do not try to create nested directories with a single MKD command  
The default behaviour when creating directories is to try to create the target directory with single command, even if it is nested. If this fails, the plugin will fall back to creating each directory individually. Some FTP servers, when asked to create nested directories will respond with a success code, but will not create the directories correctly. This will result in directories being created in incorrect branches of the directory tree.  
This option will not prevent the creation of nested directories, it is just a fix for badly behaved FTP servers.

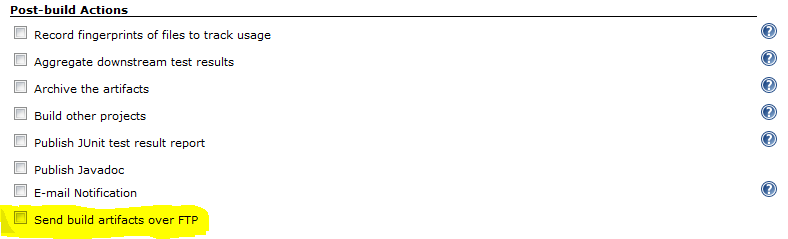
**Click "Test Configuration".**

**Add more server configurations (if required)**

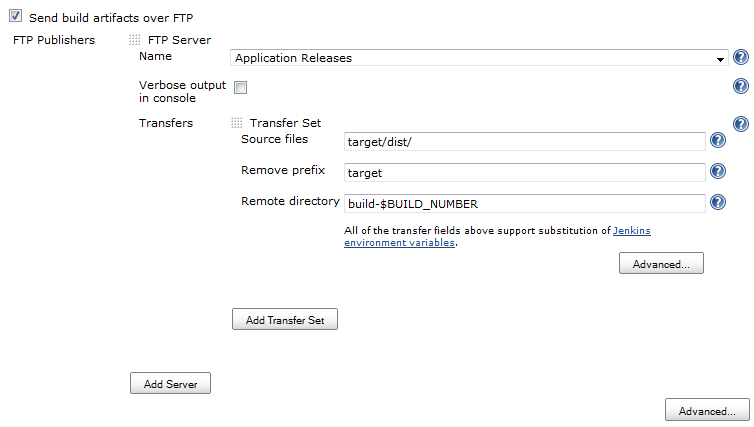
**Save**

**Configure a job to Publish Over FTP**

Open a jobs main page and then click "Configure" from the left hand menu.

Find the "Send build artifacts over FTP" checkbox in the "Post-build Actions" section (as below) and click in the box.  


You should now have the configuration options as below



## **Server**

One or more servers need to be configured to tell the publisher where to send the files.  
To add another server, click the "Add Server" button.  
The order in which the servers are used during a build can be changed by left clicking on the small square icon above Name (next to SSH/ FTP Server) and dragging to a new location.

##### Name

The name of the Server (Host configuration) to use when connecting.

## **Transfer set**

Each server will have one or more transfer sets to specify which files to send where.  
Click the "Add Transfer Set" button to add more sets.  
The order in which the transfers are performed during a build, left click the small square icon next to Transfer Set and drag to a new location.

**Environmnet variables**

Source files, Remove prefix, Remote directory and Exclude files can all use the Jenkins environment variables.  
ie. If Remote directory is build-$BUILD\_NUMBER, then for build number 9, the directory created would be build-9.  
From version 0.4, other build variables are available for substitution - most notably, matrix axis, eg $label.

##### Source files

This is an Maven include pattern see [Patterns](http://ant.apache.org/manual/dirtasks.html#patterns) in the Apache Maven Manual. Multiple includes can be specified by separating each pattern with a comma.  
See [Examples](https://wiki.jenkins.io/display/JENKINS/Publish+Over#PublishOver-examples) below.

##### Remove prefix

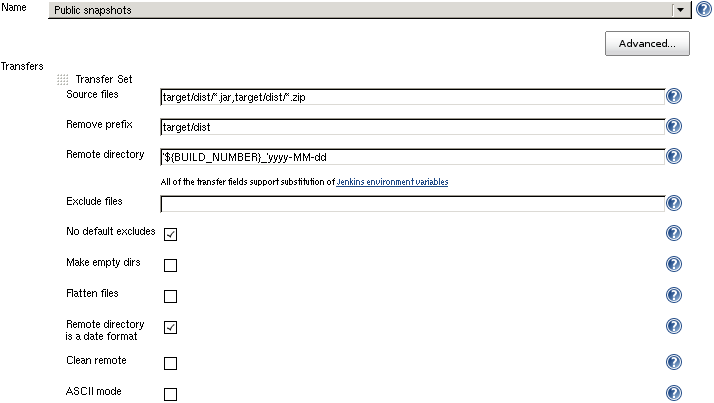
When transferring files, unless "Flatten files" has been selected, then the entire directory structure will be transferred from the base directory used for the Source files (usually the Workspace).  
This option allows the removal of the higher parts of the directory structure (nearest the base directory).  
This option is will be matched and removed from the front of the file path - whilst Jenkins environment variables will be substituted, it will **not** be expanded with shell like glob syntax or ant style patterns.

**Source files**

The build will fail if you do not select any Source files to transfer

**Advanced (Transfer Sets)**

If you click the "Advanced..." button for a Transfer Set, then you will make more configuration options available (see below)



**ASCII mode**

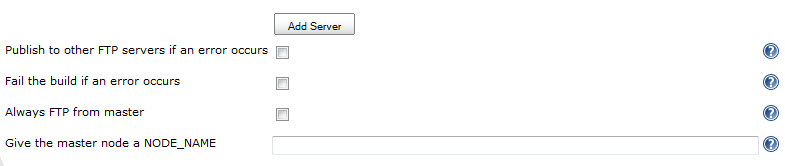
Transfer the files using ASCII mode.  
This means that during the transfer, new lines and carriage returns will be converted from the format used on the client operating system to the format used on the servers operating system.  
Do not use this option if your Source files pattern will match any binary files, as there is a good chance that binary files will be corrupted during transfer.

**Clean remote**

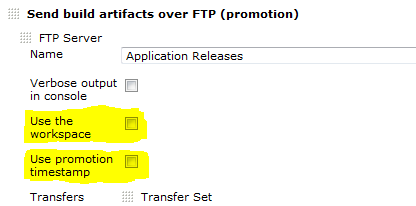
Delete all files and directories from the remote directory before uploading the new files.

**Advanced (Publisher)**

If you click the "Advanced..." button that is immediately below the "Add Server" button, then you will make more configuration options available (see below)



**Options to override the default Promotion behaviour**

If you are configuring an action in a promotion, new options will become available.  


* The Source files are rooted in the artifacts directory.   
  Any artifact that you want to transfer during a promotion should be "archived" in the post-build actions of the build. This will make it available during a promotion. This means that if you have builds 1, 2 and 3 and you choose to "promote" build #2, then the artifacts for build 2 will be available to the publisher.
* The original builds environment variables are restored. This means that $BUILD\_NUMBER will be replaced by the build number of the build under promotion. The environment variables for the promotion itself are available with the prefix promotion\_, so to get the build number for the promotion, use the variable $promotion\_BUILD\_NUMBER.
* If the "Remote directory is a date format" option is selected, then the date used is the date of the original build, not the date of the promotion. ([see below to change](https://wiki.jenkins.io/display/JENKINS/Publish+Over#PublishOver-promotime))

When configuring a publisher for a promotion, the following extra configuration options are available at the "Server" level (options will appear below the "Verbose in console" option).

##### Use the workspace

Use the workspace as the base directory for the Source files.  
This option can be used if files are generated in the workspace during a promotion which then need to be transferred.

##### Use promotion timestamp

This option will only have any effect if the "Remote directory is a date format" option is selected.  
When selected, this option will use the time of the promotion when formatting the Remote directory, instead of using the time of the original build that is currently undergoing promotion.

# Examples

To help illustrate the examples, a contrived workspace layout is presented below:

build.xml

src/my/code/HelloWorld.java

src/my/code/HelloWorldImpl.java

src/my/code/Main.java

target/classes/my/code/HelloWorld.class

target/classes/my/code/HelloWorldImpl.class

target/classes/my/code/Main.class

target/jar/hello-world.jar

target/test-classes/my/code/HelloWorldImplTest.class

test/my/code/HelloWorldImplTest.java

##### Eg 1 Transfer directory

**Source files** target/classes/\*\*/\*  
**Remove prefix**  
**Remote directory**

Result:

target/classes/my/code/HelloWorld.class

target/classes/my/code/HelloWorldImpl.class

target/classes/my/code/Main.class

##### Eg 2 Remove prefix

**Source files** target/classes/  
**Remove prefix** target  
**Remote directory**

Result:

classes/my/code/HelloWorld.class

classes/my/code/HelloWorldImpl.class

classes/my/code/Main.class

target/classes/ == target/classes/\*\* == target/classes/\*\*/\*

##### Eg 3 Environment variables

Transfer all files and folders beneath a directory and place in a directory named with the job name and then build number (given the job is called "Hello World")  
**Source files** target/classes/  
**Remove prefix** target/classes  
**Remote directory** $JOB\_NAME/$BUILD\_NUMBER

Result:

Hello World/99/my/code/HelloWorld.class

Hello World/99/my/code/HelloWorldImpl.class

Hello World/99/my/code/Main.class

##### Eg 4 Transfer .class files

**Source files** target/\*\*/\*.class  
**Remove prefix** target  
**Remote directory**

Result:

classes/my/code/HelloWorld.class

classes/my/code/HelloWorldImpl.class

classes/my/code/Main.class

test-classes/my/code/HelloWorldImplTest.class

##### Eg 5 Transfer files with flatten

**Source files** \*\*/\*.java  
**Remove prefix**  
**Remote directory** /java  
**Flatten files** checked

Result:

java/HelloWorld.java

java/HelloWorldImpl.java

java/Main.java

java/HelloWorldImplTest.java

##### Eg 6 Remote directory is a date format

**Source files** target/\*\*/\*.jar  
**Remove prefix**  
**Remote directory** 'builds/'yyyy/MM/dd/'build-$BUILD\_NUMBER'  
**Flatten files** checked  
**Remote directory is a date format** checked

And finally go and check your remote directory :

Result:

builds/2010/11/07/build-99/hello-world.jar