Workspace Rules

1. Copy file(s) from the server's work space and put them in the agent's workspace.

Note the **>** operator tells us we're sending files from the server to the agent.

Rule Example Type 1 - Operators (Server to Agent)

```
[1.1] Test\** >   Output
[1.2] Test\** ->  OutputFlattened
[1.3] Test\** >>  OutputEmptied
[1.4] Test\** ->> OutputEmptiedFlattened
```

Each rule will find all files in the **Test** directory in the server's workspace and copy them to the agent's workspace directory.

[1.1] - Puts files and directories in **Output**. Preserves the directory structure.

[1.2] - Traverse the entire directory structure of **Test** and put all files matched into the **OutputFlattened** directory on the agent. Also so known as flattening the files.

[1.3] - Before copying the files and preserving their path, make sure **\OutputEmpied** is actually empty.

[1.4] - Before copying the files and flattening them, make sure **\OutputEmpiedFlattened** is actually empty.

2. Copy file(s) from the agent's work space and put them in the server's workspace.

Note the **<** operator tells us we're sending files from the agent to the server .

Rule Example Type 2 - Operators (Agent to Server)

```
[2.1] Test < Output\**
[2.2] TestFlattened\ <- Output\**
[2.3] TestEmptied\ << Output\**
[2.4] TestEmptiedFlattened\ <<- Output\**
```

Each rule will find all files in the **Output** directory in the agent's workspace and copy them to the server's workspace directory.

[2.1] - Puts files and directories in **Test**. Preserves the directory structure.

[2.2] - Traverse the entire directory structure of **Output** and put all files matched into the **TestFlattened** directory on the server. Also so known as flattening the files.

[2.3] - Before copying the files and preserving their path, make sure **\TestEmpied** is actually empty.

[2.4] - Before copying the files and flattening them, make sure **\TestEmpiedFlattened** is actually empty.

3. Copy files using different wildcards

The example rules below all use the preserve operator **>** or **<**, for these examples it's not the operator that's important so we've just used any.

Rule Example Type 3 - Wildcards

```
[3.1] Test\* > Output
[3.2] Test\image_?.png > Output
[3.3] Test\*.??? > Output
[3.4] Test\*\**.dll > Output
[3.5] Test\**.xml > Output
```

[3.1] - Matches any **files** (not directories) in the server's **\Test** directory and puts them in the agent's **\Output** directory.

[3.2] - Matches any files with the name "**image_"** followed by a single character, followed by "**.png"** and puts them in the agent's **\Output** directory.

[3.3] - Matches any **files** (not directories) in the server's **\Test** directory which end with a period "**.**" followed by 3 characters. It basically matches any files that have a 3 character long file extension.

[3.4] - Matches any files ending with "**.dll**" and are two directories down from the server workspace's **\Test** directory. The first wild card matches a single level of directories.

[3.5] - Matches any files ending with "**.xml**" anywhere underneath the server workspace's **\Test** directory.

4. Copy files and maintaining directory structure

Using the preserve paths operator won't preserve the entire path of the file that's matched. It will only preserve paths matched by the directories the wild card matches. If you specify a directory name with no wildcards in your pattern and that directory comes before any wildcards, then to keep that exact structure you must put those directories in the destination path.

Rule Example Type 4 - Entire path preserving

```
[4.1] Test\**.xml > Output
[4.2] Test\Stuff\**.xml > Output

# Notice how the destination pattern directories mimic the non-wildcard directories in the source pattern.
[4.3] Test\**.xml > Output\Test
[4.4] Test\Stuff\**.xml > Output\Test\Stuff
```

[4.1] and [4.2] - Any files matched are put in the **\Output** directory and the directory they were found in will be created in **\Output**.

[4.3] - Any files matched are put in the **\Output\Test** directory and the directory they were found in will be created in **\Output\Test**.

[4.4] - Any files matched are put in the **\Output\Test\Stuff** directory and the directory they were found in will be created in **\Output\Test\Stuff** .

5. Copy files to the workpace's root directory

The root of the workspace can be specified by either a forward slash "**/**" or a backslash "****".

Rule Example Type 5 - Workspace root

```
[5.1] AppInstallers*.exe > /
[5.2] / < AppInstallers*.exe
```

[5.1] - Matches any files in the root of the server's workspace that start with "**AppInstallers**" and ends with "**.exe**" and puts them in the agent's workspace root.

[5.2] - Matches any files in the root of the agent's workspace that start with "**AppInstallers**" and ends with "**.exe**" and puts them in the server's workspace root.

6. Excluding files from being copied

Exclude rules are grouped by direction (Server to Agent/Agent to Server) and applied to **all** include patterns that match the direction.

Rule Example Type 6 - Exclude patterns

```
[6.1a] Output\** > LotsOfStuff
[6.1b] - Output\**.xml >

[6.2a] Installers < Output\**
[6.2b] - < Output\**.xml
[6.2c] - < Output\**.dll
```

[6.1a] and [6.1b] execute together. The server pattern **Output**** starts generating a list of files it matches, as a match is found it's checked against every exclude pattern, in this example there is only one **Output**.xml**. The result of combining these two rules is every file in the server's **\Output** directory is copied to the agent's **\LotsOfStuff** directory except for files that end with "**.xml**".

[6.2a], [6.2b] and [6.2c] also executed together. The agent pattern Output** starts generating a list of files it matches, as a match is found it's checked against the two exclude patterns [6.2b] and [6.2c]. This example will copy all files in **Output** except for those that end with "**.xml**" or "**.dll**" and puts them in the **\Installers** directory on the server-side workspace.

7. Extracting archive files copied from the server's work space to the agent's workspace.

Add the colon ":" operator after the zip extension on the left-hand side of a server-to-agent rule to specify that the archive file should be extracted. Note that only zip archive files are currently supported.

Rule Example Type 7 - Archive extracting rules

```
7.1 Report.zip: > Report/ExtractedFiles
7.2 Report.zip: -> Report/ExtractedFlattened
7.2 Report.zip: >> Report/ExtractedFiles
7.2 Report.zip: ->> Report/ExtractedFlattened
```

The archive file in the server workspace is automatically extracted after being copied to the agent workspace. Note that the operators for preserving and emptying the destination folder are also taken into account when extracting. Also note that archive files can only be extracted when copying from the server workspace to the agent workspace.

[7.1] - Extracts all the files in the archive **Report.zip** to the folder **Report/ExtractedFiles**. Preserves the directory structure within the zip file.
[7.2] - Extracts all the files in the archive **Report.zip** to the folder **Report/ExtractedFlattened**. Flattens the zip directory structure so that all files are extracted directly to **Report/ExtractedFlattened**.
[7.3] - Extracts all the files in the archive **Report.zip** to the folder **Report/ExtractedFiles**. Empties the destination folder before extracting files.
[7.4] - Extracts all the files in the archive **Report.zip** to the folder **Report/ExtractedFlattened**. Empties the destination folder before extracting files. Flattens the zip directory structure so that all files are extracted directly to **Report/ExtractedFlattened**.

8. Extracting archives using wildcards

You can also add a pattern to specify which files to extract from the server archive file.

Rule Example Type 8 - Archive extracting rules with wildcards

```
8.1 Report.zip:/*.html > ReportHtmlFiles
8.2 Report.zip:/**.html > Report/HtmlFiles
8.3 Report.zip:/Main/**.xml > MainReport/XmlFiles
8.4 Report.zip:/Report/**/Main/**.html > MainReport/HtmlFiles
```

[8.1] - Extracts all the **html** files in the root folder of the archive **Report.zip** to the folder **ReportHtmlFiles**. Preserves the directory structure within the zip file.
[8.2] - Extracts all the **html** files in the archive **Report.zip** to the folder **Report/HtmlFiles**. Preserves the directory structure within the zip file.
[8.3] - Extracts all the **xml** files under the Main folder in the archive **Report.zip** to the folder **MainReport/XmlFiles**. Preserves the directory structure under the Main folder within the zip file.
[8.4] - Extracts all the **html** files under the Report folder in the archive **Report.zip** which are under a sub-folder named Main to the folder **MainReport /HtmlFiles**. Preserves the directory structure under the Report folder within the zip file.

9. Compressing files in the agent's workspace to an archive to copy to the server's work space.

Specifying a zip file extension on the left-hand side of a agent-to-server rule means that the agent files should be compressed. The colon **":"** after the zip name operator is optional, but can be used to specify a destination sub-folder (see 10. below). Note that only zip archive files are currently supported.

Rule Example Type 9 - Archive creating rules

```
9.1 Report.zip < Report/Files/**
9.2 Report.zip: < Report/Files/*.html
9.3 Report.zip: <- Report/Files/**.html
9.4 Report.zip: << Report/**/Main/*.xml
```

The matching files in the agent workspace are compressed to an archive file which is then copied to the server workspace. Note that the operators for preserving and emptying the destination folder are also taken into account when extracting. Also note that files can only be compress to an archive when copying from the agent workspace to the server workspace.

[9.1] - Compress **all** the files under the folder **Report/Files** to the archive **Report.zip**. Preserves the directory structure under **Report/Files** when adding files to the zip file.
[9.2] - Compress all the **html** files directly in the folder **Report/Files** to the archive **Report.zip**. Preserves the directory structure under **Report/Files** when adding files to the zip file.
[9.3] - Compress all the **html** files under the folder **Report/Files** to the archive **Report.zip**. Flattens the directory structure so that all files are compressed to the root folder of the zip file.
[9.4] - Compress all the **xml** files directory under the folder **Report** directly under a sub-folder named **Main** to the archive Report.zip. Preserves the directory structure under **Report** when adding files the zip file. Deletes any existing Report.zip files created in previous rules and creates a new zip file.

10. Compressing archives to a sub-folder within an archive

You can also defined a sub-folder within the zip file to compress the files to.

Rule Example Type 10 - Archive creating rules with sub-folder

```
10.1 Report.zip:MainReport < Report/Files/**
10.2 Report.zip:MainReport/Html < Report/Files/**.html
```

The matching files in the agent workspace are compressed to the specified subfolder in the archive file which is then copied to the server workspace.

[10.1] - Compress all the files under the folder **Report/Files** to the **MainReport** sub-folder in the archive **Report.zip**. Preserves the directory structure under **Report/Files** when adding files to the zip file.
[10.2] - Compress all the html files under the folder **Report/Files** the **MainReport/Htm**l sub-folder in the archive **Report.zip**. Preserves the directory structure under **Report/Files** when adding files to the zip file.