Repository Rules

1. Copy entire repository into the agent's workspace

Note the **>** operator is always used, you can alternatively use **>>** which will empty the target directory before copying the repository.

Rule Example Type 1 - Single Repository

```
[1.1] $Source.MyRepo$ > Source\MyRepo
[1.2] $Source.ToolsRepo$ > Binaries
```

[1.1] - Takes the contents of the Repository **MyRepo** and puts it in the agent's workspace in the directory **Source\MyRepo**.

[1.2] - You don't need to place a repositories' contents in the **\Source** directory, it's more of a convention. This rule will take the contents of the Repository **ToolsRepo** and put it in **\Binaries** in the agent's workspace.

Note: By specifying to copy the entire repository like examples 1.1 and 1.2, you can take advantage of the Source descriptor when supplying values to actions. In other words, if you create a action, for example MSBuild and want to use the build file "**project.sln**" found in the **MyRepo** repository, you could use the syntax "**$Source.MyRepo$\project.sln**". The alternate way of getting a path to the file would be "**$Workspace$\Source\MyRepo\project.sln**".

2. Copy parts of a repository into the agent's workspace

Note the **>** operator is always used, you can alternatively use **>>** which will empty the target directory before copying the repository.

Rule Example Type 2 - Parts of a repository

```
[2.1] $Source.MyRepo$\WebProject\** > Source\MyRepoWebProject
[2.2] $Source.ToolsRepo$\**exe > Executables
```

[2.1] - Searches the contents of Repository **MyRepo** and gets all files out of the **WebProject** directory and puts them in the agent's workspace in the directory **Source\MyRepoWebProject**.

[2.2] - Searches the contents of Repository **ToolsRepo** and gets all files that have the extension "**.exe**" and puts them in the agent's workspace in the directory **Executables**..

Note: Since you can specify certain parts of the Repository, it's possible to put two parts in two different destination locations. Eg. You could put the **$Source.MyRepo$\WebProjects** directory in **\Source\WebProject** and **$Source.MyRepo$\WCFProject** in **\MyProject**. Because of this, you can't use the syntax **$Source.MyRepo$** in your action to reference the repository directory because it resolves to two different locations depending on which part of the repository you need. If you want to reference those parts of the repository you will need remember the destination directory within the workspace. For example, if you want to access a file in the **WebProjects** section of the repository you would use the syntax **$Workspace$\Source\MyRepoWebProject** and to reference something in the **WCFProject** section of the repository you would use **$Workspace$\Executables**.

3. Copy all repositories

Note the **>** operator is always used, you can alternatively use **>>** which will empty the target directory before copying the repository.

Rule Example Type 3 - All repositories

```
[3.1] $Source$ > Source
```

[3.1] - The **$Source$** descriptor is a special descriptor that refers to all repositories attached to the Configuration. When specifying a destination directory like we have with **Source**, each repository will be placed in its own directory which is then placed inside the **Source** directory. The directory name is determined by the repositories' name when it was created in Continua. For example, if you have two repositories **MyRepo1** and **MyRepo2** and you used a rule like 3.1, then your repository contents would be placed in **\Source\MyRepo1** and **\Source\MyRepo2**.

Note: Any paths or patterns specified after the **$Source$** descriptor is ignored, so if for some reason all your repositories have a common directory and you only want that directory copied then you will need to specify a rule for each repository.

4. Repository exclude rules

Note the **>** operator is always used, you can alternatively use **>>** which will empty the target directory before copying the repository.

Rule Example Type 4 - Excluding repositories and parts of repositories

```
[4.1a] $Source.MyRepo$ > MyRepo
[4.1b] - $Source.MyRepo$\WebProject\** > MyRepo

[4.2a] $Source$ > Source
[4.2b] - $Source.MyRepo$ > Source\MyRepo

[4.3a] $Source$ > Source
[4.3b] - $Source.MyRepo$\WebProject\** > Source\MyRepo
```

[4.1a] and [4.1b] - These two rules execute together. The first rule instructs Continua to copy all contents of the **MyRepo** repository into the **MyRepo** directory in the agent's workspace. However, the exclude rules instructs Continua to ignore and not copy any files in **MyRepo** repository which live in the **WebProject** directory. For this and any other excludes to work, you must specify a destination directory and it must be the directory in which the include rule has put the files. In this particular example, that directory is simply the one specified in the include rule's destination pattern.

[4.2a] and [4.2b] - The first rule instructs Continua to copy all contents of all repositories into their respective directory in the agent's workspace. The following exclude rule [4.2b] tells Continua to not copy files in the **MyRepo** repository to the destination the include rule was going to put it to. For an exclude rule to work correctly when using the **$Source$** descriptor, you need to know where the include rule is going to put a repositories' contents. Recall example [3.1] and note how the **$Source$** descriptor automatically creates a repositories' directory in the destination specified in the rule. In this case, the **MyRepo** repository was going to be placed in **Source\MyRepo**, so when specifying an exclude rule for the **MyRepo** repository, the destination must be **Source\MyRepo**.

[4.3a] and [4.3b] - These rules are identical to the rules in [4.2], but in this case I've specified to exclude a directory within a repository. Note how I didn't specify **Source\MyRepo\WebProject** as the exclude directory. You only need to specify the repositories root directory when excluding any or all parts of it.

5. Extracting archive files exported from the repository to the agent's workspace.

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

Rule Example Type 5 - Extracting archive rules

```
5.1 $Source.ReportRepo$/Report.zip: > Source/MyRepoWebProject/Report/ExtractedFiles
5.2 $Source.ReportRepo$/Report.zip: -> Report/ExtractedFlattened
5.2 $Source.ReportRepo$/Report.zip: >> Report/ExtractedFiles
5.2 $Source.ReportRepo$/Report.zip: ->> Report/ExtractedFlattened
```

The archive file in the repository is automatically extracted after being exported to the agent workspace. Note that the operators for preserving and emptying the destination folder are also taken into account when extracting.

[5.1] - Extracts all the files in the archive **Report.zip** in repository **ReportRepo** to the folder **Report/ExtractedFiles**. Preserves the directory structure within the zip file.
[5.2] - Extracts all the files in the archive **Report.zip** in repository **ReportRepo** to the folder **Report/ExtractedFlattened**. Flattens the zip directory structure so that all files are extracted directly to **Report/ExtractedFlattened**.
[5.3] - Extracts all the files in the archive **Report.zip** in repository **ReportRepo** to the folder **Report/ExtractedFiles**. Empties the destination folder before extracting files.
[5.4] - Extracts all the files in the archive **Report.zip** in repository **ReportRepo** 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**.

6. Extracting archives using wildcards

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

Rule Example Type 5 - Extracting archive rules with wildcards

```
6.1 $Source.ReportRepo$/Report.zip:/*.html > ReportHtmlFiles
6.2 $Source.ReportRepo$/Report.zip:/**.html > Report/HtmlFiles
6.3 $Source.ReportRepo$/Report.zip:/Main/**.xml > MainReport/XmlFiles
6.4 $Source.ReportRepo$/Report.zip:/Report/**/Main/**.html > MainReport/HtmlFiles
```

[6.1] - Extracts all the **html** files in the root folder of the archive **Report.zip** in repository **ReportRepo** to the folder **ReportHtmlFiles**. Preserves the directory structure within the zip file.
[6.2] - Extracts all the **html** files in the archive **Report.zip** in repository **ReportRepo** to the folder **Report/HtmlFiles**. Preserves the directory structure within the zip file.
[6.3] - Extracts all the **xml** files under the **Main** folder in the archive **Report.zip** in repository **ReportRepo** to the folder **MainReport/XmlFiles**. Preserves the directory structure under the Main folder within the zip file.
[6.4] - Extracts all the **html** files under the **Report** folder in the archive **Report.zip** in repository **ReportRepo** 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.