

# ANT - DATA TYPES

[http://www.tutorialspoint.com/ant/ant\\_data\\_types.htm](http://www.tutorialspoint.com/ant/ant_data_types.htm)

Copyright © tutorialspoint.com

Ant provides a number of predefined data types. Do not confuse the term "data types" with those that are available in the programming language, instead consider them as a set of services that are built into the product already.

The following data types are provided by Apache Ant.

## Fileset

The fileset data types represents a collection of files. It is used as a filter to include or exclude files that match a particular pattern.

For example, refer the following code. Here, the src attribute points to the source folder of the project.

The fileset selects all .java files in the source folder except those contain the word 'Stub'. The case-sensitive filter is applied to the fileset which means a file with the name Samplestub.java will not be excluded from the fileset.

```
<fileset dir="${src}" casesensitive="yes">
  <include name="**/*.java"/>
  <exclude name="**/*Stub*"/>
</fileset>
```

## Pattern set

A pattern set is a pattern that allows to filter files or folders easily based on certain patterns. Patterns can be created using the following meta characters:

- ? - Matches one character only.
- \* - Matches zero or many characters.
- \*\* - Matches zero or many directories recursively.

The following example depicts the usage of a pattern set.

```
<patternset >
  <include name="src/**/*.java"/>
  <exclude name="src/**/*.Stub*"/>
</patternset>
```

The patternset can then be reused with a fileset as follows:

```
<fileset dir="${src}" casesensitive="yes">
  <patternset ref="/>
```

## File list

The filelist data type is similar to the file set except the following differences:

- filelist contains explicitly named lists of files and it does not support wild cards.
- filelist data type can be applied for existing or non-existing files.

Let us see the following example of the filelist data type. Here, the attribute **webapp.src.folder** points to the web application source folder of the project.

```
<filelist >
  <file name="applicationConfig.xml"/>
```

```
<file name="faces-config.xml"/>
<file name="web.xml"/>
<file name="portlet.xml"/>
</filelist>
```

## Filter set

Using a filterset data type along with the copy task, you can replace certain text in all files that matches the pattern with a replacement value.

A common example is to append the version number to the release notes file, as shown in the following code.

```
<copy todir="${output.dir}">
  <fileset dir="${releasenotes.dir}" includes="**/*.txt"/>
  <filterset>
    <filter token="VERSION" value="${current.version}"/>
  </filterset>
</copy>
```

In this Code:

- The attribute **output.dir** points to the output folder of the project.
- The attribute **releasenotes.dir** points to the release notes folder of the project.
- The attribute **current.version** points to the current version folder of the project.
- The copy task, as the name suggests, is used to copy files from one location to another.

## Path

The **path** data type is commonly used to represent a class-path. Entries in the path are separated using semicolons or colons. However, these characters are replaced at the run-time by the executing system's path separator character.

The classpath is set to the list of jar files and classes in the project, as shown in the example below.

```
<path >
  <pathelement path="${env.J2EE_HOME}/${j2ee.jar}"/>
  <fileset dir="lib">
    <include name="**/*.jar"/>
  </fileset>
</path>
```

In this code:

- The attribute **env.J2EE\_HOME** points to the environment variable **J2EE\_HOME**.
- The attribute **j2ee.jar** points to the name of the J2EE jar file in the J2EE base folder.