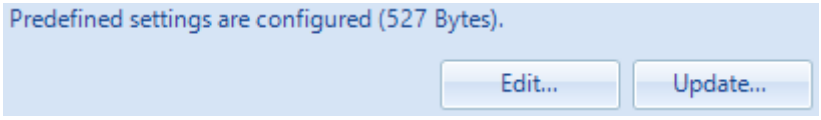
4.5.3 Modifying predefined settings

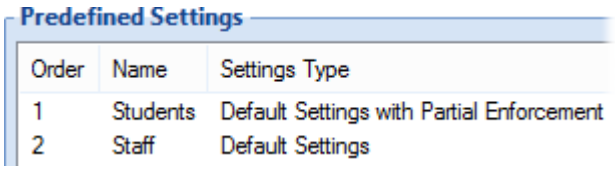
If you want to use different predefined settings, you can either click *Edit…* to modify the predefined settings that were installed or created previously or click *Update…* and select another profile archive.



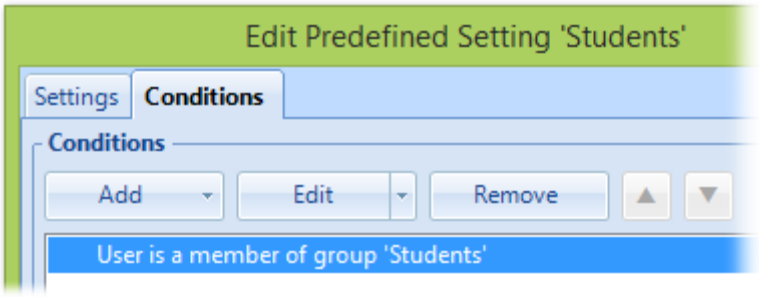
In the first case, Windows Explorer will open to a folder with the expanded predefined settings – see section 4.5.2.1 for more information. Note that only non-empty folders are displayed. Click the Add Default Folders button to add the other folders.

4.5.4 Multiple predefined settings

A single Flex config file can contain multiple predefined settings, to provide different settings to different groups of users, for instance.



If you want to specify multiple predefined settings, each must have so-called conditions defined that control whether the entry is applicable for a certain user. Conditions are specified on the *Conditions* tab of the predefined settings dialog and are described in detail in chapter 6.



FlexEngine will process the predefined settings in list order (indicated by the value in the *Order* column), evaluating conditions. As described in 4.5.1, the order can be changed using the *Move Up* and *Move Down* buttons.

If conditions match for multiple predefined settings, the last one "wins".

4.5.5 Placeholders

Predefined settings can contain so-called placeholders that are replaced with information from environment variables when imported. Placeholders can be used in file names and folder names (to create user-specific names) and in the content of text files.

4.5.5.1 Placeholder format

Placeholders have the format **[Flex#%*var*%]** where ***var*** is the name of an environment variable. For instance, if a predefined settings archive is imported containing a file named **Desktop\[Flex#%username%].txt**, this will result in a text file on the user's desktop with the file name being set to the user's name (i.e. the value of the **%username%** variable).

NOTE: The string **Flex** in the placeholder must be specified exactly like that, i.e. with a capital **F**. The name of the environment variable is not case-sensitive, however.

4.5.5.2 Using placeholders in text files

To have FlexEngine process placeholders within the contents of a text file (for instance **.REG** files, **.TXT** files, or **.INI** files), the file's name must contain a specific token: **[Flex#]** (case sensitive). This token will be removed on import, so it does not affect the resulting file name – it is just used to trigger placeholder replacement.

NOTE: When a predefined settings archive is built (cf. 4.5.2.1 and 4.5.3), the Flex Profiles.reg file in the *Registry* subfolder and **.txt**, **.ini**, or**.xml** files in any of the other folders are scanned for placeholders. If a placeholder is found, the file name is automatically marked up with the **[Flex#]** token.

The placeholders in the file contents are formatted in the same way as described in the previous section: use **[Flex#%*var*%]** where ***var*** is the name of an environment variable.

To extend the example from the previous section: by renaming the file in the predefined profile archive to **Desktop\[Flex#%username%][Flex#].txt**, its contents will be processed as well.

If that file contains the text

```
Hi [Flex#%username%], you are logged on to [Flex#%computername%].
```

and user **JohnDoe** logs on to computer **WIN7B91**, a file **JohnDoe.txt** will be created on the desktop, with the following contents:

```
Hi JohnDoe, you are logged on to WIN7B91.
```

NOTES

- When using placeholders in **.REG** files, use the alternative **[Flex#%*var*%#reg]** format. This will escape any backslashes or double quotes in the replaced content, in accordance with the **.REG** format.
- For applications that refer to users via their SID you can use **[Flex#%SID%]**. The special **%SID%** variable is replaced by the user's SID in the well-known **S-1-5-21-** format.
- When processing placeholders in text files, FlexEngine tries to determine the text encoding automatically. This auto-detect mechanism supports Unicode with a *Byte Order Mark* (in the UTF-8, UTF-16 Big Endian, and UTF-16 Little Endian variants) and the system's default encoding.
  If you need to process a file in a different encoding (or if the auto-detect fails in your scenario), you can explicitly specify the code page by using the special token **[Flex#*codepage*]** in the file name instead of **[Flex#]**.
  For instance, **Sample[Flex#1251].txt** would be interpreted as being encoded as Windows Cyrillic (code page 1251). For a list of valid code pages, see Code Page Identifiers on the Microsoft website.