Allgemein

Inhaltsverzeichnis

- 1 Allgemeine Attribute
- 2 Set
- 3 Print
- 4 ThrowError
- 5 ForEach
- 6 ContinueForEach
- 7 ExitForEach
- 8 Repeat
- 9 ContinueRepeat
- 10 ExitRepeat
- 11 Function
- 12 CallFunction
- 13 Eval
- 14 Sleep
- 15 Pause
- 16 Stop
- 17 ClearLog

Die Aktionen in der Kategorie Allgemein

1 Allgemeine Attribute

Die allgemeinen Attribute **IgnoreError**, **Variable** und **Condition** können bei allen Aktionen angegeben werden. Die Attribute sind optional und brauchen nur bei Bedarf hinterlegt werden. Wenn diese für eine Aktion nicht benötigt werden, können diese aber auch zur besseren Lesbarkeit des Skriptes entfernt werden.

IgnoreError

Das optionale Attribut **IgnoreError** gibt an, ob bei einem Fehler die Ausführung des Batchpad Skriptes abbricht oder das Skript weiter ausgeführt werden soll. Der Wert muss dem Typ Boolean (true oder false) entsprechen.

Variable

Das optionale Attribut **Variable** kann immer dann verwendet werden, wenn man das Ergebnis einer auszuführenden Aktion ermitteln möchte. Variable="{@ResultFileExists}".
Die Ergebnisse sind je nach ausgeführter Aktion vom Typ her unterschiedlich, oft ist es ein Boolean (true oder false) der angibt ob die Aktion erfolgreich war. Bei Aktionen für Zeichenketten sind die Ergebnisse dann eher vom Typ String usw.

Condition

Das optionale Attribut **Condition** gibt an, ob die Aktion ausgeführt werden soll. Hierzu wird der Inhalt des Attributes als logischer Ausdruck auf Wahr oder Falsch geprüft. Der Ausdruck sollte dem Typ Boolean (true oder false) entsprechen.
Der Ausdruck kann Funktionen aus VBScript enthalten, genauso wie Operatoren NOT, OR, AND...
Mit dem Condition Attribut wertet man in der Regel Variablen aus, die Ergebnisse aus zuvor durchgeführten Aktionen enthalten. Beispiel: Condition="NOT {@ResultFileExists}"

2 Set

Mit der Aktion **Set** kann ein Wert in eine Platzhalter-Variable gesetzt werden. Diese Variable kann im späteren Ablauf des Skripts beliebig oft verwendet werden.

```
<Set Variable="" Value="" Condition="" />
```

3 Print

Die Aktion **Print** erzeugt einen Text im Protokoll der ausgeführten Definition. Das Argument *Text* beinhaltet den zu protokollierenden Text. Das Attribut *Variable* enthält den Rückgabewert der Aktion vom Typ Boolean und gibt an, ob die Aktion erfolgreich ausgeführt wurde.

```
<Print Text="" Condition="" Variable="{@Result}" IgnoreError="false" />
```

4 ThrowError

Die Aktion **ThrowError** lässt das Skript z.B. abhängig von einer Bedingung in *Condition* mit dem angegebenen Text als Fehler beenden. Das Argument *Text* beinhaltet den Text für die Fehlermeldung.

<Print Text="" Condition="" Variable="{@Result}" IgnoreError="false" />

5 ForEach

Die Aktion **ForEach** iteriert über das Objekt welche dem Attribut *Data* zugewiesen werden. Auf die Werte kann innerhalb des Tags über {@Data:Value} zugegriffen werden.

Value ist dabei nur ein Beispiel. Für diesen Wert muss beispielsweise die Bezeichnung der Tabellenspalte stehen über die iteriert wird.

Bei jedem Schleifendurchlauf werden die in ForEach verschachtelten Aktionen durchgeführt.
<ForEach Data="{@myData}" Condition="" IgnoreError="false">

... {@Data: ... } ...

</ForEach>

6 ContinueForEach

Die Aktion **ContinueForEach** bricht die aktuelle Iteration der ForEach Schleife ab und führt die nächste Iteration aus.
<ContinueForEach Condition="" IgnoreError="false" />

7 ExitForEach

Die Aktion **ExitForEach** verlässt die ForEach Schleife. Es werden keinen weiteren Iterationen durchgeführt.
<ExitForEach Condition="" IgnoreError="false" />

8 Repeat

Die Aktion **Repeat** wiederholt die Schleife bis der {@RepeatCount} Wert erreicht wird. Dabei wird der Count bei jedem Durchlauf um 1 inkrementiert. Der Wert des Counts pro Iteration zur Laufzeit hält dabei die Variable {@CurrentCount}.

Hinweis: Die Variable {@CurrentCount} wird von der Aktion Repeat deklariert und muss daher nicht zusätzlich über die Aktion SetVariable deklariert werden.

Bei jedem Schleifendurchlauf werden die in Repeat verschachtelten Aktionen durchgeführt.

<Repeat Count="{@RepeatCount}" Condition="" Variable="{@CurrentCount}" IgnoreError="false">

...

</Repeat>

9 ContinueRepeat

Die Aktion **ContinueRepeat** bricht die aktuelle Iteration der Repeat Schleife ab und führt die nächste Iteration durch.

<ContinueRepeat Condition="" IgnoreError="false" />

10 ExitRepeat

Die Aktion **ExitRepeat** verlässt die Repeat Schleife. Es werden keinen weiteren Iterationen durchgeführt.

```
<ExitRepeat Condition="" IgnoreError="false" />
```

11 Function

Innerhalb der Aktion **Function** können Aktionen hinterlegt werden, welche nur durchlaufen werden, wenn die Funktion aufgerufen wird. Siehe CallFunction.

```
<Function Name="">

...

</Function>
```

12 CallFunction

Die Aktion **CallFunktion** ruft die im Attribut Name aufgeführte Aktion aus. Das Attribut *Variable* enthält den Rückgabewert der Aktion vom Typ Boolean und gibt an ob die Aktion erfolgreich ausgeführt wurde.

```
<CallFunction Name="" Condition="" Variable="{@Result}" IgnoreError="false" />
```

13 Eval

Die Aktion **Eval** führt ein VB-Skript zum Ermitteln eines Ergebnisses aus. Mit dem Argument *Script* wird der zu prüfende VB-Skript Ausdruck angegeben. Die angegebene Variable im Argument *Variable* erhält nach Prüfung des Skriptes das Ergebnis vom Typ Boolean.

```
<Eval Script="" Condition="" Variable="{@Result}" IgnoreError="false" />
```

14 Sleep

Die Aktion **Sleep** verzögert die Ausführung des Skripts für den im Attribut *Milliseconds* angegebenen Zeitraum. Das Attribut *Variable* enthält den Rückgabewert der Aktion vom Typ Boolean und gibt an ob die Aktion erfolgreich ausgeführt wurde.

```
<Sleep Milliseconds="" Condition="" Variable="{@Result}" IgnoreError="false" />
```

15 Pause

Die Aktion **Pause** pausiert die Ausführung des Skripts. Der Einsatzzweck der Aktion Pause ist das Debuggen des Skripts. Anschließend kann ab dieser Position im Einzelschritten fortgefahren werden.

```
<Pause Condition="" IgnoreError="false" />
```

16 Stop

Die Aktion **Stop** stoppt die Ausführung des Skripts.
<Stop Condition="" IgnoreError="false" />

17 ClearLog

Die Aktion **ClearLog** leert die Ausgabe im Ausgabefenster.
<ClearLog Condition="" IgnoreError="false" />