



Chapitre 3

tMap et les fonctions Java dans Talend

1. Introduction

Le composant tMap est le composant le plus important et le plus complet dans Talend.

Grâce à ce composant, il est possible d'effectuer plusieurs transformations et jointures diverses entre différentes sources de données tout en récupérant leurs schémas, de récupérer les lignes en rejet, de créer des variables, de désigner des clés étrangères, de supprimer une ligne en sortie, d'effectuer des conditions avec plusieurs lignes en entrée, etc.

Le composant tMap permet également de faire appel à des fonctions Java afin d'effectuer des transformations diverses et variées, mais aussi de faire appel aux routines système et à celles que nous allons créer dans ce chapitre.

Dans ce chapitre, nous allons créer des routines personnalisées et nous verrons les notions Java les plus importantes pour pouvoir résoudre des problèmes d'implémentation que nous n'aurions pas pu résoudre avec les routines système de Talend.

2. Quelques notions sur Java dans Talend

Dans cette section du chapitre, nous allons voir quelques-unes des notions indispensables sur l'utilisation de Java dans Talend, avec pour chaque expression sa signification en français :

- **==** : égal
- **!=** : différent
- **=** : déclaration ou valorisation d'une variable en Java
- **Xxxx == 100 ou xxxx > 100 ou xxxx < 100** : teste la valeur d'un champ numérique
- **"toto".equals("xxx") / !"toto".equals("xxx")** : teste l'égalité/inégalité d'un champ de type String
- **Xxxx == null ou xxxx != null et Relational.isNull(Xxxx) / ! Relational.isNull(Xxxx)** : teste la nullité ou non d'un champ
- **Xxxx.isEmpty() / ! XXXX.isEmpty()** : teste si le champ est vide ou pas (vide <> null : un champ peut être non null mais vide)
- **Xxxx.startsWith ("xx") / XXXX.endsWith ("xx")** : commence/termine par xxxx, LIKE en SQL ("xx%")/SQL ("%xx")
- **Xxxx.contains ("xx")** : contient xxxx, LIKE en SQL ("%xx%")
- **Boolean** : True / False (si la valeur est en string) ; 1/0 (si la valeur est en int)

3. Comment utiliser les routines système de Talend ?

Dans cette section, nous allons nous intéresser aux routines système de Talend les plus couramment utilisées. Les routines suivent l'ordre dans lequel elles sont affichées dans le dépôt. Elles sont regroupées en fonction de leur type. Chaque type est détaillé dans une section différente.

3.1 Routines numériques

Les routines numériques permettent de renvoyer des nombres entiers ou décimaux afin de les utiliser comme paramètres dans un ou plusieurs composants du job, par exemple pour ajouter des identifiants numériques.

Routine	Description	Syntaxe
sequence	renvoie un identifiant numérique incrémentiel	<code>Numeric.sequence ("Parameter name", start value, increment value)</code>
resetSequence	crée une séquence si elle n'existe pas et lui attribue une nouvelle valeur de départ	<code>Numeric.resetSequence (Sequence Identifier, startvalue)</code>
removeSequence	supprime une séquence	<code>Numeric.RemoveSequence (Sequence Identifier)</code>
random	renvoie un nombre entier aléatoire compris entre les valeurs maximale et minimale	<code>Numeric.random (minimum start value, maximum end value)</code>
convertImplied DecimalFormat	renvoie une décimale à l'aide d'un modèle décimal implicite	<code>Numeric.convertImplied DecimalFormat ("Target Format", value to be converted)</code>

Les trois routines `sequence`, `resetSequence` et `removeSequence` sont étroitement liées :

- **La routine `sequence`** est utilisée pour créer un identifiant de séquence, nommé `s1` par défaut, dans le job. Cet identificateur de séquence est global dans le job.
- **La routine `resetSequence`** peut être utilisée pour initialiser la valeur de l'identifiant de séquence créé par la routine `sequence`.
- **La routine `removeSequence`** est utilisée pour supprimer l'identificateur de séquence de la liste des variables globales dans le job.

3.2 Routines relationnelles

Les routines relationnelles permettent de vérifier les affirmations basées sur des booléens.

Routine	Description	Syntaxe
<code>ISNULL</code>	Elle vérifie si la variable fournie est une valeur nulle. Elle renvoie <code>true</code> si la valeur est <code>NULL</code> et <code>false</code> si la valeur n'est pas <code>NULL</code> .	<code>Relational.ISNULL(variable)</code>
<code>NOT</code>	renvoie le complément de la valeur logique d'une expression	<code>Relational.NOT(expression)</code>
<code>isNull</code>	Elle vérifie si la variable fournie est une valeur nulle. Elle renvoie 1 si la valeur est <code>NULL</code> et 0 si la valeur n'est pas <code>NULL</code> .	<code>Relational.isNull(variable)</code>

3.3 Routines sur les chaînes de caractères

Les routines `StringHandling` permettent d'effectuer différents types d'opérations et de tests sur des expressions alphanumériques, sur la base de méthodes Java.

Routine	Description	Syntaxe
ALPHA	Elle vérifie si l'expression est classée par ordre alphabétique. Elle retourne le booléen vrai ou faux en conséquence.	<code>StringHandling.ALPHA</code> ("Chaîne de caractères à vérifier")
IS_ALPHA	Elle vérifie si l'expression contient des caractères alphabétiques uniquement, ou sinon elle retourne un booléen vrai ou faux en conséquence.	<code>StringHandling.IS_ALPHA</code> ("Chaîne de caractères à vérifier ")
CHANGE	Remplace un élément d'une chaîne de caractères par un élément défini et renvoie la nouvelle chaîne de caractères.	<code>StringHandling.CHANGE</code> ("Chaîne de caractères à vérifier ", " Chaîne de caractères à remplacer ", "Chaîne de remplacement")
COUNT	Renvoie le nombre de fois qu'un caractère est contenu dans une chaîne de caractères.	<code>StringHandling.COUNT</code> ("Chaîne de caractères à vérifier ", "Sous-chaîne de caractères à compter")
DOWNCASE	Convertit toutes les lettres majuscules d'une expression en minuscules et renvoie la nouvelle chaîne de caractères.	<code>StringHandling.DOWNCASE</code> ("Chaîne de caractères à convertir")

Routine	Description	Syntaxe
UPCASE	Convertit toutes les lettres minuscules d'une expression en majuscules et renvoie la nouvelle chaîne de caractères.	StringHandling.UPCASE ("Chaîne de caractères à convertir ")
DQUOTE	Place une expression entre des doubles guillemets.	StringHandling.DQUOTE ("Chaîne de caractères à mettre entre guillemets")
EREPLACE	La fonction EREPLACE remplace toutes les occurrences de sous-chaînes qui correspondent à une expression régulière donnée dans une chaîne donnée par une nouvelle sous-chaîne de remplacement, et renvoie une nouvelle chaîne.	StringHandling.EREPLACE (Ancienne chaîne de caractères, regex, remplacement)
INDEX	Elle renvoie la position du premier caractère dans une sous-chaîne spécifiée, dans une chaîne entière. Si la sous-chaîne spécifiée n'existe pas dans la chaîne entière, la valeur - 1 est renvoyée.	StringHandling.INDEX ("Chaîne de caractères à vérifier", "sous-chaîne spécifiée")
LEFT	Spécifie une sous-chaîne qui correspond aux n premiers caractères d'une chaîne.	StringHandling.LEFT ("Chaîne de caractères à vérifier", nombre de caractères)

Routine	Description	Syntaxe
RIGHT	Spécifie une sous-chaîne qui correspond aux n derniers caractères d'une chaîne.	StringHandling.RIGHT ("Chaîne de caractères à vérifier", nombre de caractères)
LEN	Calcule la longueur d'une chaîne de caractères.	StringHandling.LEN ("Chaîne de caractères à vérifier ")
SPACE	Génère une chaîne de caractères composée d'un nombre défini d'espaces vides.	StringHandling.SPACE (Nombre d'espaces à générer)
SQUOTE	Place une expression entre des guillemets simples.	StringHandling.SQUOTE ("Chaîne de caractères à mettre entre guillemets simples")
STR	La fonction STR génère une chaîne contenant un caractère spécifique répété un certain nombre de fois.	StringHandling.STR (caractère à générer, nombre de fois)
TRIM	Supprime les espaces et les tabulations avant le premier caractère non vide d'une chaîne de caractères et après le dernier caractère non vide et renvoie la nouvelle chaîne de caractères.	StringHandling.TRIM ("Chaîne de caractères à vérifier ")
BTRIM	Supprime tous les espaces et tabulations après le dernier caractère non vide dans une chaîne de caractères et renvoie la nouvelle chaîne de caractères.	StringHandling.BTRIM ("Chaîne de caractères à vérifier ")