Objet : Annotation @Referential - chargement et injection automatique des rfrentiels dans Spring

Bonjour [Nom du destinataire],

Je vous partage un rsum du fonctionnement de l'annotation personnalise @Referential, utilise dans notre projet pour le chargement automatique de rfrentiels et leur injection directe via Spring.

Objectif

Cette annotation permet de lier dynamiquement des sources de donnes (souvent issues de fichiers ou de tables Spark) des objets Java structurs (List, Map, ou Broadcast) utiliss comme rfrentiels dans nos traitements.

Fonctionnement

- L'annotation @Referential se place directement sur une classe Java contenant les champs de type List<T>, Map<K, T> ou Broadcast<List<T>>.
- Lors de l'injection Spring (@Autowired), le systme dclenche automatiquement le chargement Spark de la ressource partir :
  - d'une requte SQL dfinie (query)
  - et/ou d'un filtre (filter)
- Si le champ est de type Map, la cl est automatiquement dtecte.
- Le paramtre reload = true permet de forcer le rechargement du rfrentiel chaque instanciation (scope prototype).

Avantages

- Simplifie l'accs aux rfrentiels dans les jobs Spark sans logique de chargement manuelle.

- Supporte plusieurs types de structure (List, Map, Broadcast).

- Intgration transparente avec Spring et injection automatique des donnes dans les steps (ex. FixedIncomeLoad).

Bien cordialement,

[Votre Prnom Nom]

Subject: @Referential annotation - automatic loading and injection of reference data in Spring

Hi [Recipient's Name],

Here is a short explanation of the custom annotation @Referential, used in our project to automatically load reference datasets and inject them directly via Spring.

Purpose

The @Referential annotation allows linking structured Java objects (List<T>, Map<K, T>, Broadcast<List<T>>) to reference datasets sourced from Spark queries or filtered tables.

How it works

- The annotation is applied directly to a Java class containing fields of type List, Map, or Broadcast.
- During Spring injection (@Autowired), the system automatically triggers the Spark loading process using:
  - a SQL query defined in the annotation
  - and/or a row-level filter
- If the field is a Map, the key type is auto-detected.
- The reload = true flag forces the reference data to be reloaded at each instantiation (prototype scope).

Benefits

- Simplifies access to reference data in Spark jobs with no manual loading logic.
- Supports multiple structures (List, Map, Broadcast).
- Seamless Spring integration with automatic injection into steps (e.g. FixedIncomeLoad).

Best regards,

[Your First Name Last Name]