

4.8.2 CAMT: Zuordnungslogik

🌐 Sprache: **Deutsch** · [English](#)

📖 4.8.2 CAMT: Zuordnungslogik

Die Zuordnung von eingehenden Bankumsätzen zu Referenzobjekten (Rechnungen, o.ä.) ist in der Methode `findMatchingReferences()` der abstrakten Klasse `org.nuclet.camt.logic.AbstractCAMT054Logic` realisiert.

REFERENZ

INTEGRATOREN

STAND: JUL 2026

GILT FUER NUCLOS 4.2026.X

Auf dieser Seite

- [Verwandte Seiten](#)

Die Zuordnung von eingehenden Bankumsätzen zu Referenzobjekten (Rechnungen, o.ä.) ist in der Methode `findMatchingReferences()` der abstrakten Klasse `org.nuclet.camt.logic.AbstractCAMT054Logic` realisiert. Die Zuordnung erfolgt standardmäßig auf Grundlage der herausgestellten Referenznummer der Referenzobjekte:

- Ist die Referenznummer im Verwendungszweck eines Bankumsatzes zu finden, erfolgt eine Zuordnung.
- Andernfalls erfolgt keine Zuordnung.

Diese Logik wird durch den Ausdruck `boBankTransaction.getInformationToAccountOwner().indexOf(strReference) >= 0` (aus dem Code-Segment, s. u.) beschrieben.

Soll diese Regelung modifiziert werden (bspw. dahingehend, dass die Prüfung weniger strikt erfolgen soll), ist hier anzusetzen. Nach Möglichkeit sollte die Änderung nicht direkt in der Klasse `AbstractCAMT054Logic` vorgenommen werden. Die dafür vorgesehene Sourcecode-Stelle in der konkreten Implementierung `CAMT054Logic` ist entsprechend mit einem `@replace!`-Tag markiert.



Es wird empfohlen, die Änderungen in der konkreten Implementierung `CAMT054Logic` oder in einer eigenen Klasse vorzunehmen, die von der abstrakten Klasse `AbstractCAMT054Logic` erbt.

```
/**
 * Matches the given bank transaction with a map of references:
 *
 * If the transaction's field "information to account owner" contains the reference, the related
 * BusinessObject will be memorised, i.e. a list of all matching entries will be returned.
 *
 * @note A more sophisticated, application specific, behaviour might be implemented in
 * dedicated subclasses
 *
 * @param boBankTransaction a BusinessObject of type <code>BankTransaction</code>
 * @param mpReferences a map of references to be matched
 * @return a list of all BusinessObject that relate to a matching reference
 *
 * @throws BusinessException might be thrown by implementing classes in case of errors or other exceptions
 */
protected List<T> findMatchingReferences(final BankTransaction boBankTransaction,
    final Map<T, String> mpReferences)
throws
    BusinessException
{
    final ArrayList<T> lstMatchingReferences = new ArrayList<T>();
    final Set<T> stReferencedBusinessObjects = mpReferences.keySet();

    // Check, if a matching reference exists...
    for (final T businessObject : stReferencedBusinessObjects) {
        final String strReference = mpReferences.get(businessObject);

        logTrace("checking \"" + strReference + "\"...");

        if (strReference != null || strReference.trim().length() == 0) {
            if (boBankTransaction.getRemittanceInformation().indexOf(strReference) >= 0) {
                log("matching reference found: " + strReference);

                lstMatchingReferences.add(businessObject);
            }
        } else {
            logWarn("");
        }
    }
    return lstMatchingReferences;
}
```

Verwandte Seiten

Nuclet: CAMT(sind im Handelsnuclet verfügbar)

CAMT-Datenblatt

[Öffnen](#)

Schnittstellen

Alle Integrationen

[Öffnen](#)