4.8.1 CAMT: Reference objects to be considered

🛠️ 4.8.1 CAMT: Reference objects to be considered

> **Reference objects to be considered – part of the CAMT Nuclet documentation (import and processing of CAMT.053/CAMT.054 bank statements (Cash Management)).**

HOW-TO | INTEGRATORS | UPDATED: JUL 2026 | APPLIES TO NUCLOS 4.2026.X

On this page

⚠️ **Machine-translated – under review**

This page was machine-translated from German as a first draft and is currently under review. The authoritative source remains the **German original** (see the language switch above).

The getReferences() method of the abstract class AbstractCAMT054Logic is executed before the import process. It serves to read in all already existing objects that are to be used as possible reference objects when importing the bank transactions (e.g. customer invoices that are still open and have not already been assigned to a bank transaction).

The default behaviour of the Nuclet considers all reference objects that are in one of those statuses defined via the getSourceStates() method of the ReferenceFacade class (see section 4.7.3).

If the default behaviour is to be modified, supplemented or extended for your own application, a custom getReferences() method must be created in the CAMT054Logic class or in your own implementation of AbstractCAMT054Logic.

org.nuclet.camt.logic.CAMT054Logic

```
/**
  * Fetches all references that need to be considered during the CAMT.054 import process, e.g.
  * all open client billings/invoices that have not yet been linked to other bank transactions
  *
  * @note A dummy implementation of <code>org.nuclet.mt940.logic.AbstractMt940Logic</code> has
  * been  provided with this class here, its methods are meant to be implemented with user
  * specific behaviour.
  *
  * @return all references that need to be considered during the CAMT.054 import process; the return
  * value comes as as <code>Map</code> in which the <code>BusinessObject</code> represents the
  * map's key while the text, which is meant to be used to link the reference to a bank
  * transaction, represents the map's value
  *
  * @throws BusinessException might be thrown in case of errors or other exceptions
  *
  */
 public Map<BusinessObject, String> getReferences() throws BusinessException
 {
     final HashMap<BusinessObject, String> mpReferences = new HashMap<BusinessObject, String>();

     final Query<ClientBilling> qryGetReferencesStateAB = QueryProvider.create(ClientBilling.
class)
                .where(ClientBilling.NuclosState.eq(ProcessClientBillingSM.State_AB.getId()))
                .and(ClientBilling.Facturated.eq(Boolean.FALSE));

     final List<Rechnung> lstReferencesStateAB = QueryProvider.execute(qryGetReferencesStateAB);

     for (final ClientBilling clientBilling : lstReferencesStateAB) {
         mpReferences.put(clientBilling, clientBilling.getBillingNumber());
     }

     final Query<ClientBilling> qryGetReferencesStateXY = QueryProvider.create(ClientBilling.
class);
             .where(ClientBilling.NuclosState.eq(ProcessClientBillingSM.State_XY.getId()))
                 .and(ClientBilling.Facturated.eq(Boolean.FALSE));

     final List<ClientBilling> lstReferencesStateXY = QueryProvider.execute
(qryGetReferencesStateXY);

     for (final ClientBilling clientBilling : lstReferencesStateXY) {
         mpReferences.put(clientBilling, clientBilling.getBillingNumber());
     }

     return mpReferences;

}
```

Related pages

📖 Nuclet: CAMT (available in the trade Nuclet)

CAMT Nuclet data sheet

Open

📖 Interfaces

All integrations

Open