GK/Retail OmniPOS

Simply Retail.

App Enablement 2.0

Version: 5.19.1

Copyright

Table of contents

1 App Enablement API

1.1 Summary

The term **app-enabled** describes the ability to add custom functionality to a client application like, for example, a POS Client by using apps and by allowing the interaction between such an app and the supporting client application.

Such an app can call backbone services in the enterprise or may provide its own business logic in its own backend. It may react to events generated by the client and call functions provided by the App Enablement API to make the supporting client react to an action or an event in the app.

The interaction between an app and the supporting client is based on listener mechanisms and events:

- The communication from client to app is realized by a listener concept:
  To allow the app to react to events in the client, listeners can be registered (method `registerListener`, see below).
  As a result, an app can, for example, react to a scan event or to the registration of an item by showing item details or recommendations that are called by the app from a respective backend system.
- The communication from app to client is realized by a set of JavaScript API functions:
  The App Enablement API provides a set of methods that can be called to make the supporting client react to the events or actions in the app or actively ask the client for more information.
  As a result, the system can for example take an item that has been searched and found in the app and register it to the transaction in the client.

An example of application using App Enablement could be a recommendation application running in the POS. This application might be hooked on the registration process and recommend additional, better, or cheaper products based on currently registered items. Such applications could be created by a partner company and loaded from a remote server.

This document has three chapters: The first chapter describes the general API that is provided for apps and the second chapter illustrates the respective implementation in the POS Client (Full Client/Thin Client).

1.2 App Enablement API

The functionality is provided for a JavaScript application running in the host application.

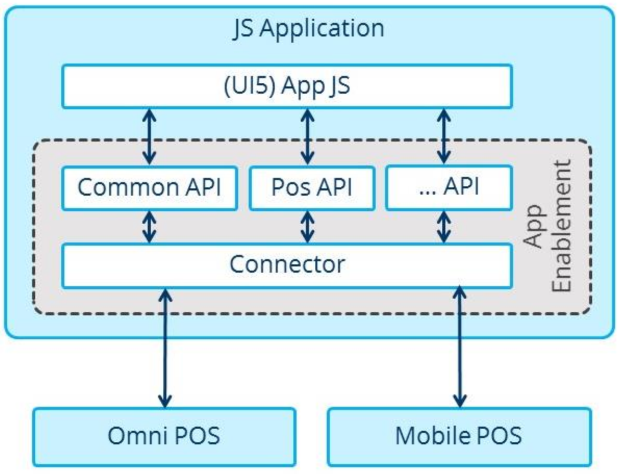
GK/Retail Omnichannel Point-of-Sale of version 5.5 supports a new version of the App Enablement API that is consistent in structure, terminology, and functionality. The new version of the API provides different JavaScript files, each focusing on a certain functionality and bundling functions that are relevant for a certain group of clients.

So, for example, for a POS Client, all JavaScript files of the API may be implemented while for other clients only one or selected files are relevant.

1.2.1 Architecture

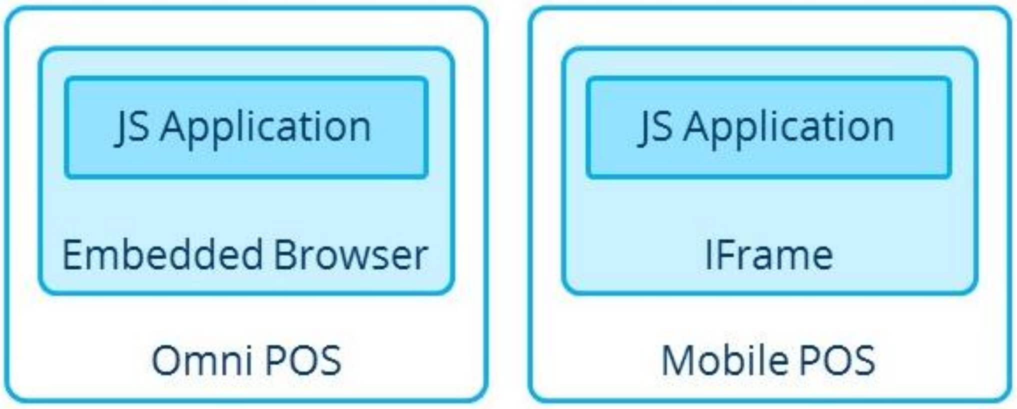
A JavaScript application is hosted in a host application (POS Full or Thin Client) that serves as container. App Enablement provides functionality of the host application for the JavaScript application running in it.

The JavaScript application contains a client part of App Enablement - connector and APIs. The connector establishes communication between APIs and the host, APIs provide the functionality bundled for different purposes and client groups.



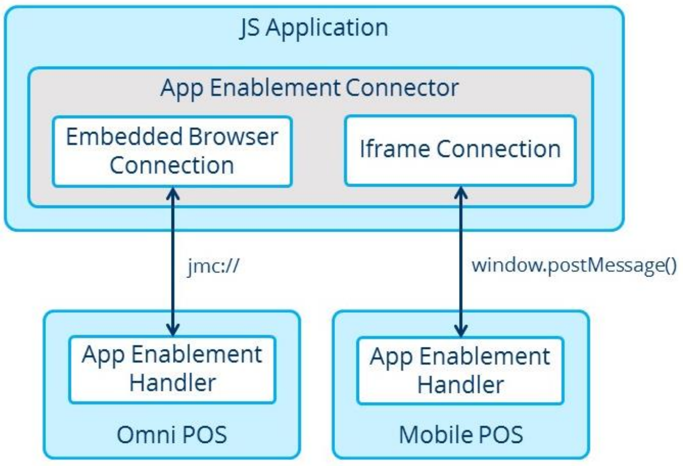
It differs how the JavaScript application is running in the host application:

The POS Client (full and thin) is a desktop Java application featuring an embedded browser that is used to run the JavaScript application.



If the JavaScript application is running in an embedded browser, the Connector calls an URL in order to invoke a functionality in the host application. The URL contains a special protocol name `jmc`. There is a protocol handler registered for this special protocol in the embedded browser (on server side). The handler parses the request and calls the API.

If the JavaScript application is running in an IFrame, the Connector uses a `window.postMessage()` in order to invoke a functionality in the host application. This method comes from HTML5 and allows sending messages between windows/frames across domains.



1.2.2 Packages

1.2.2.1 File Common.js

Description

The file provides basic App Enablement functionality, that is, general functions that are relevant for all kinds of clients.

Namespace

`comGkSoftwareGkrAppEnablementApi.Common`

Supported methods

Method	Parameters	Description	Example	Since

oAppEnablement
 	
CommonInstance.
 	
getSessionContext
	sResultFunction, sErrorFunction	Calls for context information on the current user session of the hosting client, like, for example, the user language or the tenant or store ID to which the user is logged in.	• Get the user's language to display texts in the respective language.
			• Get tenant and store to display store-specific content.
			
			Example response object:
			`{`
			`  businessUnitGroupID: "100000000000000001",`
			`  businessUnitID: "9090", // Equivalent to store ID`
			`  isoCurrencyCode: "USD",`
			`  storeLanguage: "en_US",`
			`  tenantID: "004",`
			`  userLanguage: "zh_CN",`
			`  workstationID: "107"`
			`}`
				2.0.0

oAppEnablement
 	
CommonInstance.
 	
createRegister
 	
ListenerRequest
	`sEventName, sEventListenerName,`
	
	`bPassData`
		Creates a request object for the function `registerListener`. The function will also "stringify" the complete object for the passed parameters.	Example request object:	2.0.0

Method	Parameters	Description	Example	Since
			`{ "event": "EVENT_TRANSACTION_UPDATED", "listener": "returnEventNotification", "passData": false || true }`	
oAppEnablement CommonInstance. registerListener	sRegisterListenerRequest	Registers a listener for the given event so that the app can react to that event. If `passData` in the request object is set to `true`, the function hands back the notification and the complete event including the event message.	Allow the app to react for example to a hardware-related event like the opening of the cash drawer or to an error event like an error during printing.	2.0.0
oAppEnablement CommonInstance. createUnregister ListenerRequest	sEventName, sEventListenerName	Creates a request object for the function `unregisterListener`. The function will also "stringify" the complete object for the passed parameters.	Example request object: `{ itemID: "030039", language: "en_US", isoCurrencyCode: "USD" }`	2.0.0
oAppEnablement CommonInstance. unregisterListener	sEvent	Removes the listener for the given event.	Unregister the listeners so that the app can log off the event bus.	2.0.0
oAppEnablement CommonInstance. closeBrowser		Closes the App and disposes of browser instance.		2.1.0
oAppEnablement CommonInstance. hideBrowser		Hides the App without closing it (keeps browser state, all registered listener continue to receive and handle events).		2.1.0
oAppEnablement CommonInstance. getOperatorData	sResultFunction, sErrorFunction	Calls for information on the current operator, like, for example, the operatorID, the name of the operator and the operators current rights.	Example response object: `{ operatorID: "1", workerID: "1", salutation: "Mr.", firstName: "John", lastName: "Doe", rightsSet: ["S.00000000001.00","S.00000000001.01"] }`	2.1.0

1.2.2.2 File ExternalMasterdata.js

Description

The file provides functions that deal with master data that is accessible from an external repository.

Namespace

`comGkSoftwareGkrAppEnablementApi.ExternalMasterdata`

Supported methods

Method	Parameters	Description	Example	Since
oAppEnablementExternalMasterdataInstance.createGetItemByCriteriaRequest	`sItemID, oContext`	Creates a request object for function `getItemByCriteria`. The function will also "stringify" the complete object for the passed parameters.	Example request object: `{ "event": "EVENT_TRANSACTION_UPDATED", "listener": "returnEventNotification" }`	2.0.0
oAppEnablementExternalMasterdataInstance.getItemByCriteria	sResultFunction, sErrorFunction, oGetItemByCriteriaRequest	Calls and hands back details of an item from an external repository.	Call details of an item that is available in a connected web shop.	2.0.0
oAppEnablementExternalMasterdataInstance.createGetItemListBySearchCriteriaRequest	`sSearchQuery, oContext`	Creates a request object for function `getItemListBySearchCriteria`. The function will also "stringify" the complete object for the passed parameters.	Example request object: `{ query: "camera", language: "en_US", isoCurrencyCode: "USD", recordCount: 60 }`	2.0.0
oAppEnablementExternalMasterdataInstance.getItemListBySearchCriteria	sResultFunction, sErrorFunction, oGetItemListBySearchCriteriaRequest	Calls and hands back a list of items from an external repository.	Search for items in a connected web shop, display a list of results.	2.0.0
oAppEnablementExternalMasterdataInstance.createGetExternalImageUrlRequest	`sItemID, oContext`	Creates a request object for the function `getImageUrl`. The function will also "stringify" the complete object for the passed parameters.	Example request object: `{ itemID: "030039", type: "item", language: "en_US", isoCurrencyCode: "USD" }`	2.0.0
oAppEnablementExternalMasterdataInstance.getImageUrl	sResultFunction, sErrorFunction, oGetExternal	Provides an image URL for external items.		2.0.0

Method	Parameters	Description	Example	Since
	ImageUrlRequest			

1.2.2.3 File Masterdata.js

Description

The file provides functions that deal with internal master data like, for example, item details for items that are contained in the master data.

Namespace

`comGkSoftwareGkrAppEnablementApi.Masterdata`

Supported methods

Method	Parameters	Description	Example	Since
`oAppEnablement MasterdataInstance . createGetItem DataByIDRequest`	`sItemID`	Creates a request object for function `getItemDataByID`. The function will also "stringify" the complete object for the passed parameters.	Example request object: `{ itemID: "030039" }`	2.0.0
`oAppEnablement MasterdataInstance . getItemDataByID`	`sResultFunction, sErrorFunction, oGetItemInformationByIDRequest`	Provides GK master data item information for a given item ID.	Make the client open the item information screen for the given item ID. Show item details for a selected item. Thus, for example, additional information about a recommended item can be displayed.	2.0.0
`oAppEnablement MasterdataInstance . createGetItem DataListByID ListRequest`	`aItemIDList`	Creates a request object for function `getItemDataListByIDList`. The function will also "stringify" the complete object for the passed parameters.	Example request object: `{ itemIDList : ["03039", "65005", "65007"] }`	2.0.0
`oAppEnablement Masterdata`	`sResultFunction, sErrorFunction, oGetItemDataListByIDListReques t`	Returns list of items for the list of given item ids (Note that only existing items are handed back; therefore, the return index of the resulting items is not the same than that of the IDs handed in).	• Display a list of items. • Analyze list and display	2.0.0

Method	Parameters	Description	Example	Since
`Instance.` getItemDataListByIDList			result of analysis.	
`oAppEnablement` `MasterdataInstance` `.` `createGetImage` `UrlRequest`	`sItemID`	Creates a request object for function `getImageUrl`. The function will also "stringify" the complete object for the passed parameters.	Example request object: `{ itemID: "03039", type: "item" }`	2.0.0
oAppEnablement MasterdataInstance. getImageUrl	`sResultFunction, sErrorFunction, oGetImageUrlRequest`	Returns the image URL for the given item ID.	Use URL to call and display item images.	2.0.0

1.2.2.4 File Pos.js

Description

The file provides functions that are specific for POS Clients, for example, dealing with POS transactions.

Namespace

`comGkSoftwareGkrAppEnablementApi.Pos`

Supported methods

Method	Parameters	Description	Example	Since
oAppEnablement PosInstance. createGetPOS ItemInformation ByIDRequest	sItemID	Creates a request object for function `getPOSItemInformationByID`. The function will also "stringify" the complete object for the passed parameters.	Example request object: `{ itemID: "030039" }`	2.0.0
oAppEnablement PosInstance. getPOSItem InformationByID	• `InformationByIDRequest` • `oGetPOSItem` • `sErrorFunction` • `sResultFunction`	Returns item information for the given item ID. Enriched with additional POS-specific information	Make the client open the item information screen for the given item ID. Show item details for a selected item. Thus, for example, additional information about a recommended item can be displayed.	2.0.0

Method	Parameters	Description	Example	Since
		provided by other services like, for example, stock.		

oAppEnablement
 	
PosInstance.
 	
createRegister
 	
LineItemRequest
	sItemID, oContext	Creates a request object for function `registerLineItem`.
		
		The function will also "stringify" the complete object for the passed parameters.
			Example request object:
			
			`{`
			`  "itemID": "030039",`
			`  "language": "en_US",`
			`  "isoCurrencyCode": "USD"`
			`}`
				2.0

oAppEnablement PosInstance. registerLineItem	• `oRegisterLineItemRequest` • `sErrorFunction` • `sResultFunction`	Registers a line item for the given item ID.	Register an item and thus add the item to the transaction. Add items that are displayed in the app (for example, recommended items) to the transaction.	2.0

oAppEnablement
 	
PosInstance.
 	
createRegister
 	
ExternalLine
 	
ItemRequest
	<u>Required attributes:</u>
	
	• `iQuantity`
	• `sActualUnitPrice`
	• `sItemID`
	• `sItemType`
	• `sMainPOSItemID`
	• `sPosItemID`
	• `sReceiptText`
	• `sRegistrationNumber`
	• `sTaxGroupID`
	• `sUnitOfMeasureCode`
	
	<u>Optional attributes:</u>
	
	• `aLineItemExtensionList`
	• `aPrintAdditionalLineItemTextLineList`
	• `aRetailPriceModifierList`
	• `aRetailTransactionLineItemI18NTextList`
	• `aSaleReturnLineItemCharacteristicList`
	• `aSaleReturnLineItemMerchandiseHierarchyGroupList`
		Creates a request object for function `registerExternalLineItem.`
		
		The function will also "stringify" the complete object for the passed parameters.
			Example request object (minimal request, contains all required attributes):
			
			`{`
			`  "posItemID":"3636",`
			`  "itemID":"4711",`
			`  "unitOfMeasureCode":"PCE",`
			`  "itemType":"CO",`
			`  "actualUnitPrice":15.55,`
			`  "quantity":1,`
			`  "receiptText":"Test Item",`
			`  "registrationNumber":"3636",`
			`  "mainPOSItemID":"3636",`
			`  "taxGroupID":"A1"`
			`}`
				2.0

Method	Parameters	Description	Example	Since
	• bAllowFoodStampFlag • bDiscountFlag • bFrequentShopperPointsEligibilityFlag • bNotConsideredByLoyaltyEngineFlag • bProhibitReturnFlag • bProhibitTaxExemptFlag • bWicFlag • oRetailTransactionLineItemAdditionalParameterList • oSaleReturnLineItemSalesOrderObject • sDepositTypeCode • sDiscountTypeCode • sHeight • sItemClassCode • sLength • sMainMerchandiseHierarchyGroupID • sMainMerchandiseHierarchyGroupIDQualifier • sMerchandiseHierarchyGroupDescription • sMerchandiseHierarchyGroupName • sPosDepartmentID • sPriceChangeTypeCode • sPriceTypeCode • sQuantityInputMethod • sReasonCode • sReasonCodeGroupCode • sReasonDescription • sReceiptDescription • sRegularUnitPrice • sSerialNumber • sTareCount • sTaxExemptCode • sUnits • sWarrantyDuration			

Method	Parameters	Description	Example	Since
	• `sWidth`			
oAppEnablement PosInstance. registerExternal LineItem	• `LineItemRequest` • `oRegisterExternal` • `sErrorFunction` • `sResultFunction`	Registers a line item based on external item data.	Register an item and thus add the item to the transaction. Add items that are displayed in the app (for example, recommended items) to the transaction.	2.0.0
oAppEnablement PosInstance. getCurrent Transaction	sResultFunction, sErrorFunction	Returns the current transaction.		2.0.0
oAppEnablement PosInstance. getCurrent CustomerList	sResultFunction, sErrorFunction	Returns all customers that are registered at the current transaction.	• Display customers. • Get list of customers and use that to call additional information about these customers via other (project-specific) services.	2.0.0
oAppEnablement PosInstance. createCancel CurrentTransaction Request	`sReasonCode, sReasonDescription`	Creates a request object for function cancelCurrent Transaction.	`{ "reasonCode": "abc", "reasonDescription": "A cancellation." }`	2.1.0
oAppEnablement PosInstance. cancelCurrent Transaction	sResultFunction, sErrorFunction, {}	Allows to cancel the current transaction in the POS.		2.0.0
`oAppEnablementPosInstance.createRegisterCustomerRequest`	`sCustomerId`	Creates a request object for function registerCustomer.	`{ "customerId": "10012" }`	2.0.0
oAppEnablement PosInstance. registerCustomer	sResultFunction, sErrorFunction, oRegister CustomerRequest	Allows to register a customer within the current transaction in the POS.		2.0.0

Method	Parameters	Description	Example	Since
`oAppEnablementPosInstance.createAddTransactionExtensionRequest`	`sExtensionKey, sExtensionValue`	Creates a request object for function `addTransactionExtension`.	`{ "extensionKey": "key", "extensionValue": "val" }`	2.0.0
oAppEnablement PosInstance. addTransaction Extension	`sResultFunction, sErrorFunction, oAddTransactionExtensionRequest`	Allows to add an extension to the current transaction in the POS.		2.0.0
`oAppEnablementPosInstance.createUpdateTransactionExtensionRequest`	`sExtensionKey, sExtensionValue`	Creates a request object for function `updateTransactionExtension`.	`{ "extensionKey": "key", "extensionValue": "val" }`	2.0.0
`oAppEnablementPosInstance.updateTransactionExtension`	`sResultFunction, sErrorFunction, oUpdateTransactionExtensionRequest`	Allows to update an extension in the current transaction in the POS.		2.0.0
`oAppEnablementPosInstance.createDeleteTransactionExtensionRequest`	`sExtensionKey`	Creates a request object for function `deleteTransactionExtension` .	`{ "extensionKey": "key" }`	2.0.0
`oAppEnablementPosInstance.deleteTransactionExtension`	`sResultFunction, sErrorFunction, oDeleteTransactionExtensionRequest`	Allows to delete an extension in the current transaction in the POS.		2.0.0

`oAppEnablementPosInstance.createAddAdditionalTransactionReportRequest`	`sReportIdentification,`
	
	`aPrintAdditional`
	
	`LineItemTextLineList`
		Creates a request object for function
		
		addAdditional
		
		TransactionReport.
			`{ "reportIdentification" :"AppReport", "printAdditionalLineItemTextLineList" : [{ "text" : "Lorem ipsum dolo", "sortOrder" : "", "styleID" : "NormalPlain" }] }`
			
			reportIdentification the value as in report.properties:
			
			PrintoutManipulationConfigs.Default.receiptPrinting.0.externalReportId
				2.0.0

Method	**Parameters**	**Description**	**Example**	**Since**
oAppEnablement PosInstance. addAdditional TransactionReport	`sResultFunction, sErrorFunction, oAdditionalTransactionReportRequest`	Allows to add an additional receipt to the current transaction in the POS.		2.0.0

oAppEnablement
PosInstance.
createEnter
CouponRequest
	sCouponNumber, sPrivilegeType, dPrivilegeValue	Creates a request object for function
		enterCoupon.
			Example request object:
			`{`
			`"couponNumber": "1",`
			`"privilegeType": "RP",`
			`"privilegeValue": "1"`
			`}`
				2.1.0

oAppEnablement PosInstance. enterCoupon	sResultFunction, sErrorFunction, oEnterCouponRequest	Allows to enter a coupon to the current transaction in the POS.		2.1.0
oAppEnablement.PosInstance. getLastNotVoidedTransaction	sResultFunction, sErrorFunction	Get the last not voided transaction.		
oAppEnablement.PosInstance. printReport	sResultFunction, sErrorFunction, oTransaction	This function will print the given transaction		

oAppEnablement.PosInstance.
createSelectItemVariantRequest
	sItemID, sItemUOMCode	Creates a request object for function
		selectItemVariant.
			{
			itemID: "030039"
			itemUOMCode: "PCE"
			}
				2.2.1

oAppEnablement.PosInstance. selectItemVariant	sResultFunction, sErrorFunction, oSelectItemVariantRequest	Allows to select an item variant within the item information in the POS		2.2.1-alpha.2

1.2.3 Supported Events

The App Enablement API allows the apps to register to standard events of the POS Client. For this purpose, the `registerListener` method is provided. It expects two parameters: firstly, the name of an event sent by the POS and secondly, the name of a JavaScript function to be called when this event occurs.

Each client has to implement the events that shall be reacted on by the apps. The following table shows the events implemented for both POS Clients:

Description	Event name
Event fired when transaction is updated	EVENT_TRANSACTION_UPDATED

Note that this list will be iteratively enhanced in future releases.

1.3 Implementation in POS Client (Full Client / Thin Client)

1.3.1 Implementation details

The following sections describe the technical realization of the app-enabled feature in Omnichannel Point-of-Sale.

For embedding applications in form of HTML5 with JavaScript support, the embedded JxBrowser is used. JxBrowser is based on the modern Chromium engine.

There are three basic interfaces that make up the structure for the app-enabled feature:

- The **GkAppComponent** is the UI element that represents the app. This is nothing more than a simple Swing component that hosts the intrinsic app.
- The **GkAppApi** is the Java side representation of the API that is also exposed to the JavaScript side.
- The **GkAppApiFactory** is a factory to produce **GkAppApi** instances on behalf of a **GkAppComponent**. All these interfaces are an abstraction from the concrete technology, due to that there are usually abstract implementations and JxBrowser-specific implementations.
- The **AppApiHandler** are used to encapsulate the Java method implementation for all methods per namespace.

1.3.1.1 GkAppApi interface

The **GkAppApi** is intended to encapsulate the API that is exposed to the app and also acts as a wrapper for the app. The interface is quite thin - it allows calling a function in an asynchronous fashion and register and unregister listeners. Since this interface is intended to be technology neutral, all input and output parameters are declared here as strings.

1.3.1.2 AbstractGkAppApiImpl

The **AbstractGkAppApiImpl** is a convenience, abstract base implementation of this interface. It basically implements just the listener management (besides invocation of listeners) and has an injected *ServiceLocator* in order to access POS Services. In addition, this abstract class is technology neutral but provides some infrastructure common to all other implementations.

Although this method has no implementation for the *call* method (and therefore no real strategy how to delegate the method calls to real Java calls), it makes the basic assumption that each invocation of *call* will end up in a corresponding method call of this class. Due to that, this class also contains concrete implementations of API methods that are technology-neutral (such as `getCurrentTransaction`) but it provides no strategy how to call it - this is the job of concrete subclasses.

1.3.1.3 JxBrowserGkAppApiV2Impl

The **JxBrowserAppApiV2Impl** is the implementation that is tied to the embedded browser technology. Technically, this class is a wrapper for the **Chromium Browser Engine** that extends the **AbstractGkAppApiImpl**. It basically uses two JxBrowser techniques to provide the implementation:

- By implementing the interface **DefaultNetworkDelegate** and adding it to JxBrowser, it is possible to resolve installation-specific paths (for example, the location of the JavaScript API that comes with the installation).

- By implementing the interface **ScriptContextListener**, it is possible to decode JavaScript requests from the app and call corresponding Java methods. The interface has been already implemented by JxBrowser's abstract class **ScriptContextAdapter**. There are two methods: `onScriptContextCreated` (which is invoked when a JavaScript context has been created) and `onScriptContextDestroyed` (which is invoked when a JavaScript context has been destroyed). With creating an anonymous class of **ScriptContextAdapter**, these methods can be overwritten without the need of extending **ScriptContextAdapter** and the class can be added as a listener to JxBrowser.

Implementation of protocol listeners

The **JxBrowserAppApi** registers some protocol listeners for the following protocols:

Protocol	Meaning
`posjs://`	Protocol to import product/project specific JavaScript files, also used to encode function calls.
`jmc://`	Protocol to encode Java method calls.

The protocol evaluation is done through the invocation of the implemented method `DefaultNetworkDelegate.onBeforeURLRequest` , which decides what to do in each case. In case of `pojs://` , some resources should be served (technically, it only overrides the URL to the specific static resource) and in case of `jmc://` , the call is translated to some Java method call.

Implementation of the ScriptContextListener.onScriptContextCreated

Function calls on JavaScript side are handled in this way: Consider you have a JavaScript method with the following signature:

```
oAppEnablementCommonInstance.getSessionContext(resultFunction, errorFunction)
```

Semantics: Gets the session context information as described in the API. In case everything works as expected and a session context can be created and returned, the method `resultFunction` is called, otherwise `errorFunction` . The implementation of this method constructs a URL for the `jmc://` protocol. The path part of the URL is the name of the namespace plus the Java method to be called (typically the same as the Java method). The query part contains the parameters and the callbacks.

So you receive the following URL:

```
jmc://comGkSoftwareGkrAppEnablementApi.Common/getSessionContext?onResult=resultFunction&onError=errorFunction
```

This URL is passed to the injected browser function. In this method, the request is decoded (each input and output parameter is interpreted as a JSON object) and the corresponding Java method is called. Upon completion of the Java method, the `onError` or `onResult` callback is invoked.

1.3.2 Additional events supported by the POS Client (Full Client/Thin Client)

1.3.2.1 Hardware-related events and message prefixes

Description	Event name
Scanner data event	EVENT_POS_INPUT_SCANNER_DATA
MSR data event	EVENT_POS_INPUT_MSR_DATA
Cash drawer event - cash drawer opened	EVENT_CASH_DRAWER_OPENED
Cash drawer event - cash drawer closed	EVENT_CASH_DRAWER_CLOSED
General printer error event	ERROR_EVENT_PRINTER
Printer event - offline	ERROR_EVENT_PRINTER_OFFLINE
Printer event - hangup	ERROR_EVENT_PRINTER_HANGUP
Printer event - out of paper	ERROR_EVENT_PRINTER
Printer event - cover opened	ERROR_EVENT_PRINTER_COVER_OPENED
Printer event - cover closed	EVENT_PRINTER_COVER_CLOSED
Printer event - status OK	EVENT_PRINTER_STATUS_OK
Print finished event	EVENT_PRINTER_PRINT_FINISHED
General terminal events	EVENT_TERMINAL
General terminal error event	ERROR_EVENT_TERMINAL
Terminal event - signon started	EVENT_TERMINAL_SIGNON_STARTED
Terminal event - signon finished	EVENT_TERMINAL_SIGNON_FINISHED
Terminal event - signoff started	EVENT_TERMINAL_SIGNOFF_STARTED
Terminal event - signoff finished	EVENT_TERMINAL_SIGNOFF_FINISHED
Terminal event - payment started	EVENT_TERMINAL_PAYMENT_STARTED
Terminal event - payment finished	EVENT_TERMINAL_PAYMENT_FINISHED

1.3.2.2 Flow events

Description	Event name
Event fired when the POS is started (start of the main flow)	FLOW_EVENT_POS_STARTED
Event fired when the POS is signed on	FLOW_EVENT_SIGNED_ON
Event fired when the POS is signed off	FLOW_EVENT_SIGNED_OFF
Event fired when the POS is locked	FLOW_EVENT_POS_LOCKED
Event fired when the POS is unlocked	FLOW_EVENT_POS_UNLOCKED
Event fired when the inactivity timer timeout is reached	FLOW_EVENT_INACTIVITY_TIMER
Event fire when the POS entered item registration main step	FLOW_EVENT_REGISTRATION_MAIN_ENTERED
Event fired when the POS is in registration mode without a transaction	FLOW_EVENT_REGISTRATION_NO_TRANSACTION
Event fire when the POS entered payment main step	FLOW_EVENT_PAYMENT_MAIN_ENTERED
Event fired when the POS is in payment mode with grand total in base currency	FLOW_EVENT_PAYMENT_GRANDTOTAL
Event fired when the POS is in payment mode and the currency changed	FLOW_EVENT_PAYMENT_GRANDTOTAL_CURRENCYCHANGED
Event fired when the POS is in change mode	FLOW_EVENT_CHANGE
Event fired when payment mode is canceled	FLOW_EVENT_PAYMENT_CANCELED
Event fire when the POS entered customer flow payment end transaction finished step	FLOW_EVENT_CUSTOMER_FLOW_PAYMENTEND_TRANSACTION_FINISHED_ENTERED
Event fired when the customer flow timer timeout is reached	FLOW_EVENT_CUSTOMER_FLOW_PAYMENTEND_TIMER
Event fired when the PaymentEndSco flow timer timeout starts	FLOW_EVENT_PAYMENTEND_SCO_TRANSACTION_FINISHED_ENTERED
Event fired when the PaymentEndSco flow timer timeout is reached	FLOW_EVENT_PAYMENTEND_SCO_TIME
Event fired when transaction is payed	EVENT_TRANSACTION_PAYED
Event fired when transaction is finalized or canceled	EVENT_TRANSACTION_CLOSED

Description	Event name
Event fired when transaction is recovered	EVENT_TRANSACTION_RECOVERED
Events fired when a sale return line item is created	EVENT_SALERETURNLINEITEM_UPDATED_REGISTERED
Event fired when a sale return line item quantity is updated	EVENT_SALERETURNLINEITEM_UPDATED_QUANTITY
Event fired when a sale return line item price is updated	EVENT_SALERETURNLINEITEM_UPDATED_PRICE
Event fired when a sale return line item serial number is updated	EVENT_SALERETURNLINEITEM_UPDATED_SERIALNUMBER
Event fired when a sale return line item is recovered	EVENT_SALERETURNLINEITEM_UPDATED_RECOVERED
Events fired when a voids line item is created or updated	EVENT_VOIDSLINEITEM_UPDATED_REGISTERED
Events fired when a voids line item is recovered	EVENT_VOIDSLINEITEM_UPDATED_RECOVERED
Event fired when line item discount applied	EVENT_LINEITEM_DISCOUNT_APPLIED
Event fired when line item reduction applied	EVENT_LINEITEM_REDUCTION_APPLIED
Event fired when a line item was closed	EVENT_LINE_ITEM_CLOSED
Event fired when a total is created	EVENT_TOTAL_CREATED
Events fired when a tender line item was voided	EVENT_TENDERLINEITEM_VOIDED
Events fired when a tender line item is created or updated	EVENT_TENDERLINEITEM_UPDATED_REGISTERED
Events fired when the currency changed and a tender line item is updated	EVENT_TENDERLINEITEM_UPDATED_CURRENCYCHANGED
Event fired when transaction discount applied	EVENT_TRANSACTION_DISCOUNT_APPLIED
Event fired when transaction reduction applied	EVENT_TRANSACTION_REDUCTION_APPLIED
Event fired when a customer is registered	EVENT_CUSTOMER_REGISTERED
Event fired when POS switched to offline mode	POS_STATUS_OFFLINE_EVENT
Event fired when POS switched to online mode	POS_STATUS_ONLINE_EVENT

1.3.3 Project Extensibility

A project can implement new JavaScript methods with the backend Java methods and override Java backend methods for already existing JavaScript methods of the App Enablement API.

- **Override Java backend methods**
-
  - Implement the interface `com.gk_software.pos.api.ui.component.app.AppApiExtension.`
  - Implement the methods that shall be overridden.
  - Define responsible Namespaces for this extension class (handled via `AppApiExtension.isNamespaceHandler(...)` method).
  - In your `component-descriptor.xml`, export the implementation (via `component:export-bean`) as a bean with the mandatory name **appApiExtension** and interface `com.gk_software.pos.api.ui.component.app.AppApiExtension.`
  - **Implement new JavaScript methods and Java backend**
-
  - Create a new JavaScript file, for example, ***api_ext.js***. For inclusion of this new JavaScript file, there are two possibilities:

    1. It can be bundled and included into apps like product API (standard way to include JavaScript files), for example:

```
<script type="text/javascript" src="./libs/appEnablement/api/api_ext.js"></script>
```

2. It can be bundled with the POS

   1. Add the file to deployment so that it is available at runtime in the POS_ROOT_DIR directory (for example, ***POS_ROOT_DIR/x/y/api_ext.js***).
   2. Include the new JavaScript file to your application by using special protocol `posjs`:

   ```
   <script src="posjs://x/y/posapi_ext.js">
   ```

   Hence, JavaScript extensions can be placed everywhere under POS_ROOT_DIR, you only have to guarantee that the same path is used in the script tag in HTML files.

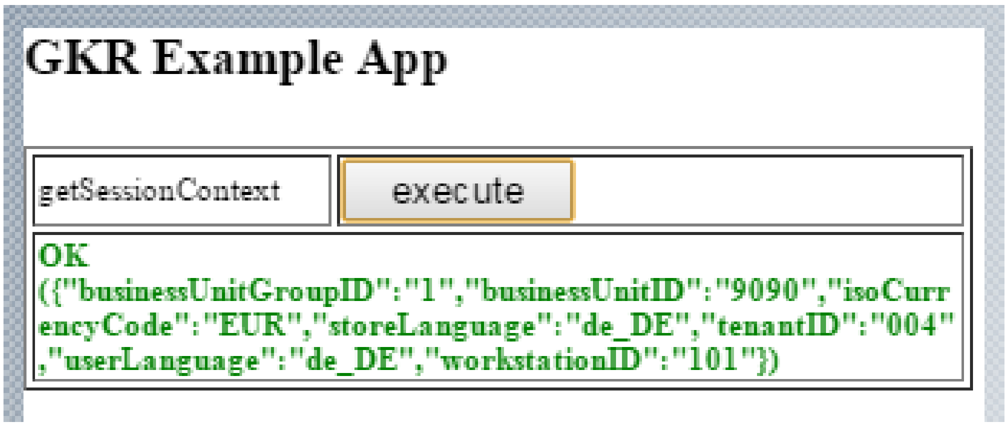
- Implement the corresponding Java backend methods. The steps are the same as in point "Override Java backend methods".

2 Samples and Best Practises

2.1 Example App

The following example shows how to create an app and how to use the App Enablement API.

```
<!DOCTYPE html>
<html dir="ltr" lang="de-DE">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <script type="text/javascript" src="./libs/appEnablement/AppEnablementConnector.js"></script>
    <script type="text/javascript" src="./libs/appEnablement/api/Common.js"></script>
    <script type="text/javascript" src="./libs/appEnablement/api/Masterdata.js"></script>
    <script type="text/javascript" src="./libs/appEnablement/api/ExternalMasterdata.js"></script>
    <script type="text/javascript" src="./libs/appEnablement/api/Pos.js"></script>
    <script type="text/javascript">
      var oAppEnablementCommonInstance = new comGkSoftwareGkrAppEnablementApi.Common();
      var oAppEnablementExternalMasterdataInstance = new
comGkSoftwareGkrAppEnablementApi.ExternalMasterdata();
      var oAppEnablementMasterdataInstance = new comGkSoftwareGkrAppEnablementApi.Masterdata();
      var oAppEnablementPosInstance = new comGkSoftwareGkrAppEnablementApi.Pos();
      function good(id, val) {
        val = JSON.stringify(val);
        setHTML(id, "<div style='width:320px; color: green; font-weight: bold; word-wrap: break-word;'>OK (" +
val + ")</div>");
      }
      function fail(ID, err) {
        setHTML(id, "<div style='color: orange; font-weight: bold;'>ERR (" + err + ")</div>");
      }
      function setHTML(id, html) {
        if (document.getElementById(id) != null) {
          document.getElementById(id).innerHTML = html;
        }
      }
      // getSessionContext
      function getSessionContext() {
        oAppEnablementCommonInstance.getSessionContext('currentSessionContextFound',
'noCurrentSessionContext');
      }
      function currentSessionContextFound(context) {
        this.context = context;
        good('statusGetSessionContext', context);
      }
      function noCurrentSessionContext(err) {
        fail('statusGetSessionContext', 'FAIL(' + err + ')');
      }
    </script>
</head>
<body style="margin:0; padding:0;">
<div style="font-size:110%; font-weight:bold; padding-bottom:20px;">
GKR Example App
</div>
<table border="1" style="font-size:70%;">
    <form onsubmit="getSessionContext();" action="javascript:void(0);">
        <tr>
            <td style="width:100px;>
                <div style="font-weight: bold">getSessionContext</div>
            </td>
            <td><input type=submit value="execute" style="width: 80px; " /></td>
        </tr>
        <tr>
            <td colspan="2" id="statusGetSessionContext">
                <div style='color: red; font-weight: bold;'>---</div>
            </td>
        </tr>
    </form>
</table>
</body>
</html>
```



```
OK
({"businessUnitGroupID":"1","businessUnitID":"9090","isoCurrencyCode":"EUR","storeLanguage":"de_DE","tenantID":"004","userLanguage":"de_DE","workstationID":"101"})
```

2.2 Register to barcode-scan event on the POS Client (Full Client/Thin Client)

In case that an app needs to react to barcode-scan events, the following steps must be performed:

1. A listener must be registered for event with ID "`EVENT_POS_INPUT_SCANNER_DATA`" and `passData` must be set to true to retrieve the scan data message.

```
oAppEnablementCommonInstance.registerListener
(oAppEnablementCommonInstance.
createRegisterListenerRequest
("EVENT_POS_INPUT_SCANNER_DATA", "processBarcode", true));
```

2. Implement a callback function which is called when a scan event occurs:

```
var processBarcode = function (scanData) {
    alert("Received scanner data: " + JSON.stringify(scanData));
};
```

The passed value for the `scanData` object has the following structure:

```
{
  "messageHeader" : {
    "messageKey" : "EVENT_POS_INPUT_SCANNER_DATA",
    "senderName" : null,
    "messagingStyle" : "ASYNCHRON",
    "processEventSource" : "SC",
    "recipient" : null
  },
  "payload" : "4002919000115",
  "scanData" : "NDAwMjkxOTAwMDExNQ==",
  "scanDataTypes" : [ 104 ],
  "messageKey" : "EVENT_POS_INPUT_SCANNER_DATA",
}
```
Code Block 1 Example message for scanned EAN13 barcode with value 4002919000115

In case that a POS version < 5.5.3 is used, the app must register on event ID "`EVENT_POS_INPUT_`" instead of the one given above.

2.3 Register Internal Line Item

2.3.1 Minimal Request

```
{
   "itemID":"1"
}
```

2.3.2 Extended Request

2.3.2.1 Minimal Promotion Data

```
{
   "itemID":"1",
   "language":"de_DE",
   "isoCurrencyCode":"EUR",
   "customerOrderID":"3",
   "customerOrderSequenceNumber":"666",
   "salesOrderTypeCode":"20",
   "quantity":1,
   "salesOrderDeliveryTypeCode":"03",
   "requestedDeliveryDate":"2017-12-24",
   "actualUnitPrice":33.33,
   "itemType":"CO",
   "retailPriceModifierList":[
      {
         "retailPriceModifierSequenceNumber":1,
         "amount":5,
         "extendedAmountBeforeModification":33.33,
         "extendedAmountAfterModification":28.33,
         "retailTransactionPriceDerivationRule":
            {
               "promotionID":333,
               "receiptPrinterName":"Test Promotion"
            }
      }
   ],
   "lineItemExtensionList":[
      {
         "extensionKey":"TestKey",
         "extensionValue":"TestValue"
      }
   ]
}
```

2.3.2.2 Maximal Promotion Data

```
{
   "itemID":"1",
   "language":"de_DE",
   "isoCurrencyCode":"EUR",
   "customerOrderID":"3",
   "customerOrderSequenceNumber":"666",
   "salesOrderTypeCode":"20",
   "quantity":1,
   "salesOrderDeliveryTypeCode":"03",
   "requestedDeliveryDate":"2017-12-24",
   "itemType":"CO",
   "retailPriceModifierList":[
      {
         "retailPriceModifierSequenceNumber":1,
         "percent":null,
         "amount":5,
         "extendedAmountBeforeModification":33.33,
         "extendedAmountAfterModification":28.33,
         "appliedQuantity":1,
         "triggerSequenceNumber":0,
         "extraAmount":0.0,
         "roundingAmount":0.0,
         "calculationBaseAmount":0.0,
         "retailTransactionPriceDerivationRule":{
            "promotionID":333,
            "priceDerivationRuleID":334,
            "priceDerivationRuleEligibilityID":335,
            "promotionDescription":"Test Description",
            "receiptPrinterName":"Test Promotion",
            "promotionPriceDerivationRuleSequence":100,
            "promotionPriceDerivationRuleResolution":200,
            "promotionPriceDerivationRuleTypeCode":"ZRKR",
            "priceModificationMethodCode":"RP",
            "priceDerivationRuleDescription":"Test Rule Description",
            "promotionOriginatorTypeCode":"01",
            "externalPromotionID":"7777",
            "externalPriceDerivationRuleID":"7778",
            "triggerQuantity":1.0,
            "giftCertificateExpirationDate":"2017-12-24",
            "discountMethodCode":"00",
            "prohibitPrintFlag":false,
            "tenderTypeCode":"ZTPR",
            "promotionTypeName":"Test Promotion Type",
            "calculationBase":"00",
            "noEffectOnSubsequentPriceDerivationRulesFlag":false,
            "prohibitTransactionRelatedPriceDerivationRulesFlag":false,
            "couponPrintoutID" : "9959999999998",
            "couponPrintoutRule" : "00",
            "couponPrintoutText" : "<CouponPrintoutText><Line><TextLine><Text>Buy a towel
and</Text><Format>BIG</Format><Format>BOLD</Format></TextLine></Line><Line><TextLine><Text>get 10%
discount!</Text><Format>BIG</Format><Format>BOLD</Format></TextLine></Line></CouponPrintoutText>",
            "exclusiveFlag":false,
            "concurrenceControlVector":"0000000000",
            "appliedCount":1.0,
            "printoutValidityPeriod":0.0
         },
         "saleReturnLineItemPromotionTriggerList":[{
            "triggerSequenceNumber" : 0,
            "triggerType" : "CO",
            "triggerValue" : "13",
            "privilegeType" : "RS",
            "privilegeValue" : 5.0,
            "reasonCode" : "6010",
            "reasonDescription" : "Reason Description",
            "triggerSequenceAddend" : 1
         }]
      }
   ],
   "lineItemExtensionList":[
      {
         "extensionKey":"TestKey",
         "extensionValue":"TestValue"
```

```
        }
    ]
}
```

2.3.3 Sales Order Pickup

```
{
    "itemID":"1",
    "customerOrderID":"3",
    "customerOrderSequenceNumber":"666",
    "salesOrderTypeCode":"20",
    "quantity":1,
    "salesOrderDeliveryTypeCode":"03",
    "requestedDeliveryDate":"2017-12-24",
    "itemType":"PU"
}
```

2.4 Register External Line Item

2.4.1 Minimal Request

```
{
    "posItemID":"3636",
    "itemID":"4711",
    "unitOfMeasureCode":"PCE",
    "itemType":"CO",
    "actualUnitPrice":15.55,
    "quantity":1,
    "receiptText":"Test Item",
    "registrationNumber":"3636",
    "mainPOSItemID":"3636",
    "taxGroupID":"A1"
}
```

2.4.2 Extended Request

2.4.2.1 Minimal Promotion Data

```
{
   "posItemID":"3636",
   "itemID":"4711",
   "posDepartmentID":"123",
   "unitOfMeasureCode":"PCE",
   "itemType":"CO",
   "regularUnitPrice":17.99,
   "actualUnitPrice":15.55,
   "quantity":1,
   "units":1.0,
   "quantityInputMethod":"01",
   "receiptText":"Test Item",
   "receiptDescription":"Test Item Description",
   "wicFlag":true,
   "allowFoodStampFlag":true,
   "registrationNumber":"54321",
   "discountFlag":true,
   "frequentShopperPointsEligibilityFlag":true,
   "discountTypeCode":null,
   "priceChangeTypeCode":"01",
   "priceTypeCode":"01",
   "notConsideredByLoyaltyEngineFlag":false,
   "merchandiseHierarchyGroupName":"Merchandise Group Name",
   "merchandiseHierarchyGroupDescription":"Merchandise Group Description",
   "itemClassCode":"icc4",
   "prohibitTaxExemptFlag":false,
   "prohibitReturnFlag":false,
   "warrantyDuration":12,
   "depositTypeCode":"00",
   "taxExemptCode":null,
   "mainPOSItemID":"963852",
   "mainMerchandiseHierarchyGroupIDQualifier":"MAIN",
   "mainMerchandiseHierarchyGroupID":"060104",
   "taxGroupID":"A1",
   "tareCount":0.0,
   "saleReturnLineItemCharacteristicList":[
        {
            "characteristicID" : "COLOR",
            "characteristicValueID" : "1",
            "characteristicValueName" : "red"
        }
   ],
   "saleReturnLineItemMerchandiseHierarchyGroupList":[
        {
            "merchandiseHierarchyGroupIDQualifier" : "MAIN",
            "merchandiseHierarchyGroupID" : "060104"
        }
   ],
   "retailTransactionLineItemI18NTextList":[
        {
            "textSequenceNumber" : 1,
            "languageID" : "de_DE",
            "category" : "SATE",
            "text" : "Test Item Information",
            "pictureFlag" : false
        },
        {
            "textSequenceNumber" : 2,
            "languageID" : "de_DE",
            "category" : "SAIC",
            "text" : "bio_product",
            "pictureFlag" : true
        },
        {
            "textSequenceNumber" : 3,
            "languageID" : "de_DE",
            "category" : "SAIC",
            "text" : "duration_low_price",
            "pictureFlag" : true
        }
   ],
   "serializedUnitModifer":{
```

```
            "serialNumber":"SN123456"
         },
         "saleReturnLineItemSalesOrder":{
            "externalCustomerOrderID":"ID4711",
            "customerOrderSequenceNumber":97,
            "salesOrderTypeCode":"10",
            "salesOrderDeliveryTypeCode":"00",
            "requestedDeliveryDate":"2017-12-24"
         },
         "reasonCode":null,
         "reasonCodeGroupCode":null,
         "reasonDescription":null,
         "retailTransactionLineItemAdditionalParameterList":[

         ],
         "retailPriceModifierList":[
            {
               "retailPriceModifierSequenceNumber":1,
               "amount":7.55,
               "extendedAmountBeforeModification":15.55,
               "extendedAmountAfterModification":8.00,
               "retailTransactionPriceDerivationRule":{
                  "promotionID":333,
                  "receiptPrinterName":"Test Promotion"
               }
            }
         ],
         "lineItemExtensionList":[
            {
               "extensionKey":"TestExtension",
               "extensionValue":"TestValue"
            }
         ]
      }
```

2.4.2.2 Maximal Promotion Data

```
{
   "posItemID":"3636",
   "itemID":"4711",
   "posDepartmentID":"123",
   "unitOfMeasureCode":"PCE",
   "itemType":"CO",
   "regularUnitPrice":17.99,
   "actualUnitPrice":15.55,
   "quantity":1,
   "units":1.0,
   "quantityInputMethod":"01",
   "receiptText":"Test Item",
   "receiptDescription":"Test Item Description",
   "wicFlag":true,
   "allowFoodStampFlag":true,
   "registrationNumber":"54321",
   "discountFlag":true,
   "frequentShopperPointsEligibilityFlag":true,
   "discountTypeCode":null,
   "priceChangeTypeCode":"01",
   "priceTypeCode":"01",
   "notConsideredByLoyaltyEngineFlag":false,
   "merchandiseHierarchyGroupName":"Merchandise Group Name",
   "merchandiseHierarchyGroupDescription":"Merchandise Group Description",
   "itemClassCode":"icc4",
   "prohibitTaxExemptFlag":false,
   "prohibitReturnFlag":false,
   "warrantyDuration":12,
   "depositTypeCode":"00",
   "taxExemptCode":null,
   "mainPOSItemID":"963852",
   "mainMerchandiseHierarchyGroupIDQualifier":"MAIN",
   "mainMerchandiseHierarchyGroupID":"060104",
   "taxGroupID":"A1",
   "tareCount":0.0,
   "saleReturnLineItemCharacteristicList":[
        {
            "characteristicID" : "COLOR",
            "characteristicValueID" : "1",
            "characteristicValueName" : "red"
        }
   ],
   "saleReturnLineItemMerchandiseHierarchyGroupList":[
        {
            "merchandiseHierarchyGroupIDQualifier" : "MAIN",
            "merchandiseHierarchyGroupID" : "060104"
        }
   ],
   "retailTransactionLineItemI18NTextList":[
        {
            "textSequenceNumber" : 1,
            "languageID" : "de_DE",
            "category" : "SATE",
            "text" : "Test Item Information",
            "pictureFlag" : false
        },
        {
            "textSequenceNumber" : 2,
            "languageID" : "de_DE",
            "category" : "SAIC",
            "text" : "bio_product",
            "pictureFlag" : true
        },
        {
            "textSequenceNumber" : 3,
            "languageID" : "de_DE",
            "category" : "SAIC",
            "text" : "duration_low_price",
            "pictureFlag" : true
        }
   ],
   "serializedUnitModifer":{
        "serialNumber":"SN123456"
   },
   "saleReturnLineItemSalesOrder":{
      "externalCustomerOrderID":"ID4711",
      "customerOrderSequenceNumber":97,
      "salesOrderTypeCode":"10",
      "salesOrderDeliveryTypeCode":"00",
      "requestedDeliveryDate":"2017-12-24"
   },
   "reasonCode":null,
   "reasonCodeGroupCode":null,
   "reasonDescription":null,
   "retailTransactionLineItemAdditionalParameterList":[

   ],
   "retailPriceModifierList":[
      {
         "retailPriceModifierSequenceNumber":1,
         "percent":null,
         "amount":7.55,
         "extendedAmountBeforeModification":15.55,
         "extendedAmountAfterModification":8.00,
         "appliedQuantity":1,
         "triggerSequenceNumber":0,
         "extraAmount":0.0,
         "roundingAmount":0.0,
         "calculationBaseAmount":0.0,
         "retailTransactionPriceDerivationRule":{
            "promotionID":333,
            "priceDerivationRuleID":334,
            "priceDerivationRuleEligibilityID":335,
            "promotionDescription":"Test Description",
            "receiptPrinterName":"Test Promotion",
            "promotionPriceDerivationRuleSequence":100,
            "promotionPriceDerivationRuleResolution":200,
            "promotionPriceDerivationRuleTypeCode":"ZRKR",
            "priceModificationMethodCode":"RP",
            "priceDerivationRuleDescription":"Test Rule Description",
            "promotionOriginatorTypeCode":"01",
            "externalPromotionID":"7777",
            "externalPriceDerivationRuleID":"7778",
            "triggerQuantity":1.0,
            "giftCertificateExpirationDate":"2017-12-24",
            "discountMethodCode":"00",
            "prohibitPrintFlag":false,
            "tenderTypeCode":"ZTPR",
            "promotionTypeName":"Test Promotion Type",
            "calculationBase":"00",
            "noEffectOnSubsequentPriceDerivationRulesFlag":false,
            "prohibitTransactionRelatedPriceDerivationRulesFlag":false,
            "couponPrintoutID" : "9959999999998",
            "couponPrintoutRule" : "00",
            "couponPrintoutText" : "<CouponPrintoutText><Line><TextLine><Text>Buy a towel
and</Text><Format>BIG</Format><Format>BOLD</Format></TextLine></Line><Line><TextLine><Text>get 10%
discount!</Text><Format>BIG</Format><Format>BOLD</Format></TextLine></Line></CouponPrintoutText>",
            "exclusiveFlag":false,
            "concurrenceControlVector":"0000000000",
            "appliedCount":1.0,
            "printoutValidityPeriod":0.0
         },
         "saleReturnLineItemPromotionTriggerList":[{
            "triggerSequenceNumber" : 0,
            "triggerType" : "CO",
            "triggerValue" : "13",
            "privilegeType" : "RS",
            "privilegeValue" : 5.0,
            "reasonCode" : "6010",
            "reasonDescription" : "Reason Description",
            "triggerSequenceAddend" : 1
         }]
      }
   ],
   "lineItemExtensionList":[
      {
         "extensionKey":"TestExtension",
         "extensionValue":"TestValue"
      }
   ]
}
```

2.4.3 Pay-in Line Item

```
{
    "itemType":"PI",
    "actualUnitPrice":15.55,
    "quantity":1,
    "reasonCode" : "6301",
    "reasonCodeGroupCode" : "E",
    "reasonDescription" : "Pay-in Reason Description",
    "retailTransactionLineItemAdditionalParameterList":[
     {
         "externalParameterID":"A1",
         "parameterName":"Additional Parameter",
         "parameterValue":"Value"
     }
    ]
}
```

2.4.4 Pay-out Line Item

```
{
    "itemType":"PO",
    "actualUnitPrice":15.55,
    "quantity":1,
    "reasonCode" : "6401",
    "reasonCodeGroupCode" : "A",
    "reasonDescription" : "Pay-out Reason Description",
    "retailTransactionLineItemAdditionalParameterList":[
     {
         "externalParameterID":"A1",
         "parameterName":"Additional Parameter",
         "parameterValue":"Value"
     }
    ]
}
```

2.4.5 Sales Order Pickup

```
{
   "posItemID":"3636",
   "itemID":"4711",
   "unitOfMeasureCode":"PCE",
   "itemType":"PU",
   "actualUnitPrice":15.55,
   "quantity":1,
   "receiptText":"Test Item",
   "registrationNumber":"3636",
   "mainPOSItemID":"3636",
   "taxGroupID":"A1",
   "saleReturnLineItemSalesOrder":{
      "externalCustomerOrderID":"ID4711",
      "customerOrderSequenceNumber":97,
      "salesOrderTypeCode":"10",
      "salesOrderDeliveryTypeCode":"00",
      "requestedDeliveryDate":"2017-12-24"
   }
}
```

2.4.6 Down Payment Clearing

```
{
   "itemType":"DC",
   "actualUnitPrice":15.55,
   "quantity":1,
   "saleReturnLineItemSalesOrder":{
      "externalCustomerOrderID":"ID4711",
      "customerOrderSequenceNumber":97,
      "salesOrderTypeCode":"10",
      "salesOrderDeliveryTypeCode":"00",
      "requestedDeliveryDate":"2017-12-24"
   }
}
```

2.5 Cancel Current Transaction

2.5.1 Minimal Request

```
{ }
```

2.5.2 Maximal Request

```
{
   "reasonCode":"CTNMC",
   "reasonDescription":"No money"
}
```

2.6 Register Customer

2.6.1 Minimal Request

```
{
   "customerId":"10065"
}
```

2.6.2 Maximal Request

```
{
   "customerId":"10065",
   "customerServiceTypeCode":"SAP_ERP",
   "preferredReceiptPrintoutTypeCode":"PRINTANDMAIL"
}
```

2.7 Enter Coupon

2.7.1 Minimal Request

```
{
    "couponNumber": "9959999999981"
}
```

2.7.2 Maximal Request

```
{
    "couponNumber": "9959999999981",
    "privilegeType": "RS",
    "privilegeValue": "2.0"
}
```

2.8 Create Transaction Extension

```
{
   "extensionKey":"txKey1",
   "extensionValue":"txValue1"
}
```

2.9 Update Transaction Extension

```
{
   "extensionKey":"txKey1",
   "extensionValue":"txValue2"
}
```

2.10 Delete Transaction Extension

```
{
   "extensionKey":"txKey1"
}
```

2.11 Add Printout Data

2.11.1 Register Internal Line Item with Additional Printout Data

```
{
   "itemID":"1",
   "printAdditionalLineItemTextLineList": [{
        "text" : "AFTER AFTER AFTER",
        "sortOrder":"afterLineItem",
        "styleID":"NormalPlain"
    },
    {
        "text" : "BEFORE BEFORE BEFORE",
        "sortOrder":"beforeLineItem",
        "styleID":"NormalPlain"
    }]
}
```

2.11.2 Register External Line Item with Additional Printout Data

```
{
   "posItemID":"3636",
   "itemID":"4711",
   "unitOfMeasureCode":"PCE",
   "itemType":"CO",
   "actualUnitPrice":15.55,
   "quantity":1,
   "receiptText":"Test Item",
   "registrationNumber":"3636",
   "mainPOSItemID":"3636",
   "taxGroupID":"A1",
    "printAdditionalLineItemTextLineList": [{
        "text" : "AFTER AFTER AFTER",
        "sortOrder":"afterLineItem",
        "styleID":"NormalPlain"
    },
    {
        "text" : "BEFORE BEFORE BEFORE",
        "sortOrder":"beforeLineItem",
        "styleID":"NormalPlain"
    }]
}
```

2.11.3 Add Additional Transaction Report

```
{
    "reportIdentification" : "AppReport",
    "printAdditionalLineItemTextLineList" : [ {
            "text" : "Example Text Line 1",
            "sortOrder" : "",
            "styleID" : "NormalPlain"
        },
        {
            "text" : "Example Text Line 2",
            "sortOrder" : "",
            "styleID" : "NormalPlain"
        }
    ]
}
```

reportIdentification is set to "AppReport" considering it's set in the report.properties to the externalReportId property to *help the POS client to find the proper report for printing.*

Contact

GK SOFTWARE SE
Waldstraße 7
08261 Schöneck
Germany

Tel.: +49 (0) 3 74 64 84 – 0
Fax: +49 (0) 3 74 64 84 – 15

Email: documentation@gk-software.com
www.gk-software.com