The Retail Innovators

GK/Retail OmniPOS

AppEnablement

Version: 5.25.0-FD

COPYRIGHT

© GK Software Group 2025. All rights reserved.

These materials are provided by GK Software SE and/or its affiliated companies (GK Software Group) for informational purposes only without representation or warranty of any kind, and GK Software Group shall not be liable for errors or omissions with respect to the materials.

The only warranties provided by GK Software Group are those that are set forth in the express warranty statements in the individual agreements between GK Software Group and its Clients or Partners. Nothing herein should be construed as constituting an additional warranty.

No part of these materials shall be reproduced or transmitted to third parties in any form or for any purpose without the express permission of GK Software Group. The information contained herein may be changed without prior notice.

All company names of GK Software Group, GK Software Group’s product/services names as well GK Software Groups logos mentioned herein are registered trademarks and intellectual property of GK Software Group.

Internal Document Information: 1247941073 | 2025-01-15

TABLE OF CONTENTS

1 AppEnablement API

1.1 Summary

The term **AppEnablement** 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 realised 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 realised by a set of JavaScript API functions:
  The AppEnablement 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 AppEnablement 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.

1.2.1 Architecture

1.2.1.1 AppEnablement before 2.5

Introduction

AppEnablement of the versions before 2.5 are only for existing projects. Maintenance only. No new API.
GK/Retail Omnichannel Point-of-Sale of version 5.5 supports a new version of the AppEnablement 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.

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

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

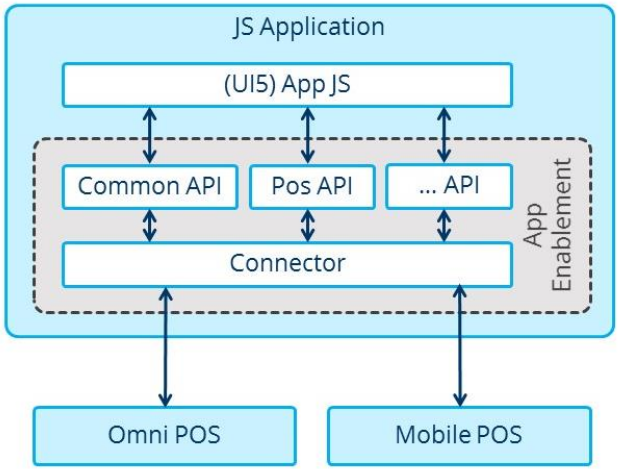


Figure 1 App Enablement before 2.5

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

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

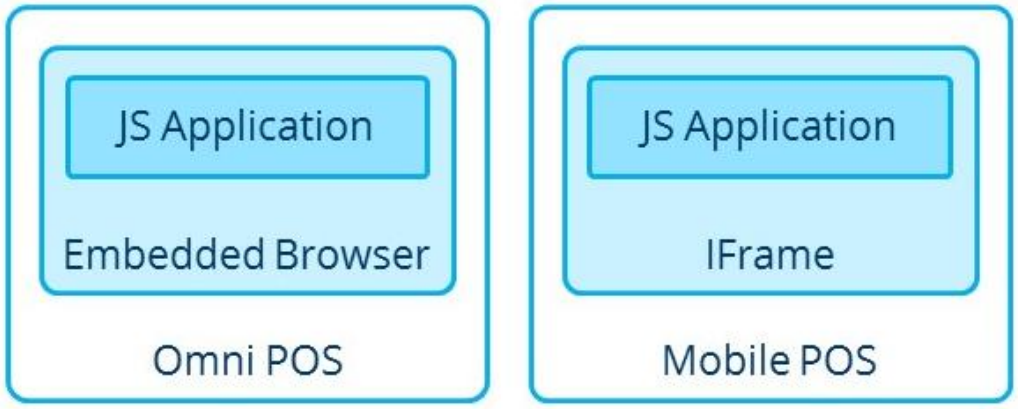


Figure 2 App Enablement

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.

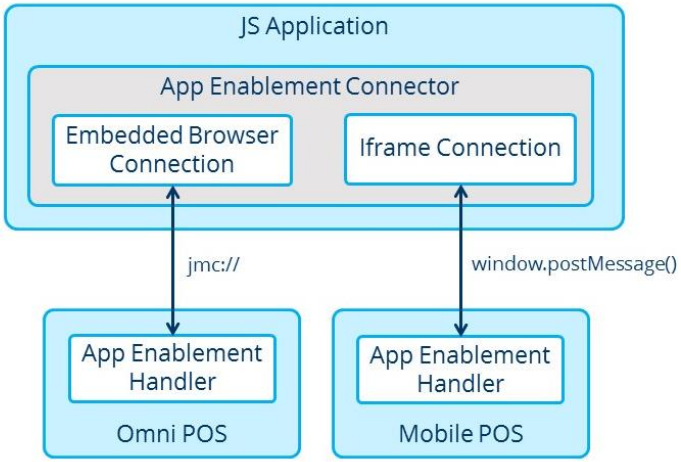


Figure 3 App Enablement before 2.5

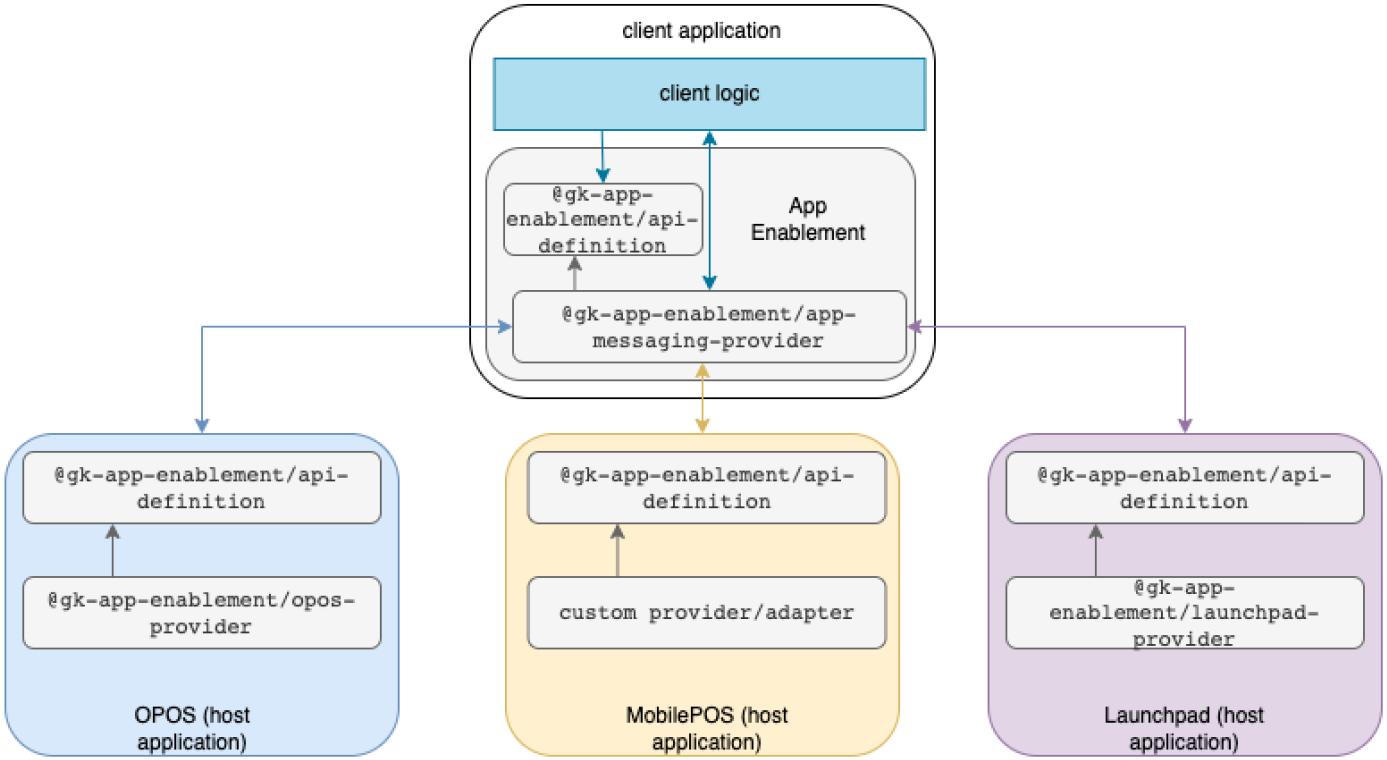
1.2.1.2 AppEnablement from 2.5

Introduction

From 5.17 new version of AppEnablement 2.5 was introduced. It brings full support of TypeScript for the clients. We provide universal API in package @gk-app-enablement/app-messaging-provider that allows our customers create applications that will work in any of our app launchers (OPOS, Launchpad, MobilePOS). Applications in our launchers are running in iFrames and communicates with launchers via postMessages. Every launcher has its own AppEnablement provider that is part of launcher distribution and its main purpose is to transfer received postMessages to calls specific for launcher/platform.

All providers and api-definition are NPM packages that can be build and distribute independently. However all providers has dependency on @gk-app-enablement/api-definition package. Currently there are two shared providers: @gk-app-enablement/opos-provider and @gk-app-enablement/launchpad-provider.

See basic overview of usage of NPM packages.

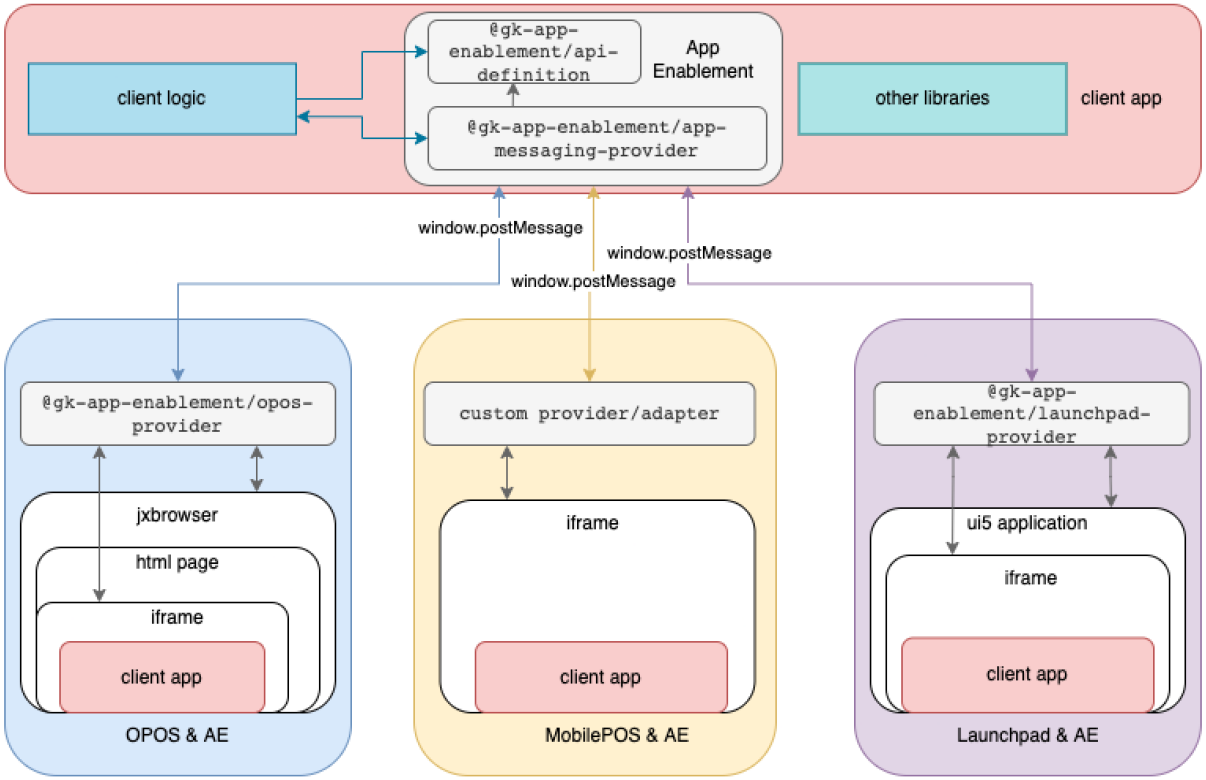


Another provider which is built in in MobilePOS was developed inside MobilePOS repository. It is similar to @gk-app-enablement/opos-provider.

What's new?

AppEnablement before version 2.5 was implemented for POS Client (Full POS/Thin POS) in very specific way and does not correctly work in other launchers. Also was written in a way that makes implementation in modern frameworks very hard. AppEnablement from version 2.5 is written in TypeScript, but can be used with both TypeScript and JavaScript. Customers only have to include messaging with API, rest of the implementation is on launcher side. No more GLOBAL VARIABLES for clients!

Clients application do not even handle the postMessages itself. Every API call needs reference for success/error callback, which is provided by client application. When @gk-app-enablement/app-messaging-provider received postMessages it will trigger referenced callback. It is non-blocking communication. Implementation of waiting for some operation to finish is clients app responsibility, if they want such a behaviour.



Launchpad

In launchpad we have an application RemoteApp that allows run 3rd party applications from launchpad menu. A boolean parameter *appEnablement* was added to this application which enable AppEnablement.

@gk-app-enablement/launchpad-provider does not provide all methods that are available in AppEnablement API. For example launchpad does not have access to transactions. User is be informed by some error warning message.

```
<menu id="other_menu_id" caption="${other_menu.caption}">
    <node id="3rd_party_node_id" caption="${3rd_party_node.caption}" type="title">
        <function   id="RemoteApp1"
                    caption="${custom_name.caption}"
                    applicationUrl="/gkr-launchpad/app/remoteApp/"
                    applicationId="com.gk_software.gkr.ui5.launchpad.remoteApp"
                    icon="https://remoteAppUrl.net/app/icon.svg">
            <parameters>
                <parameter name="remoteAppUrl" value="https://remoteAppUrl.net/app/"/>
                <parameter name="appEnablement" value="true"/>
            </parameters>
        </function>
    </node>
</menu>
```
Code Block 1 RemoteApp

UI5 application will be loaded with iFrame and included @gk-app-enablement/launchpad-provider package.

POS Client (Full Client / Thin Client)

Because of backward compatibility both AppEnablement before and from 2.5 are working in OPOS. To launch application with AppEnablement from 2.5 in POS Client a prefix *iframe:* has to be used before url.

```
iframe:file\://C:/Projects/app-debug-ts/build/index.html
```
Code Block 2 RemoteApp

The major changes from 2.5 AppEnablement are that client application does not have to use global variables. Responses and events from POS Client are handled in @gk-app-enablement/opos-provider which is preloaded in jxBrowser.

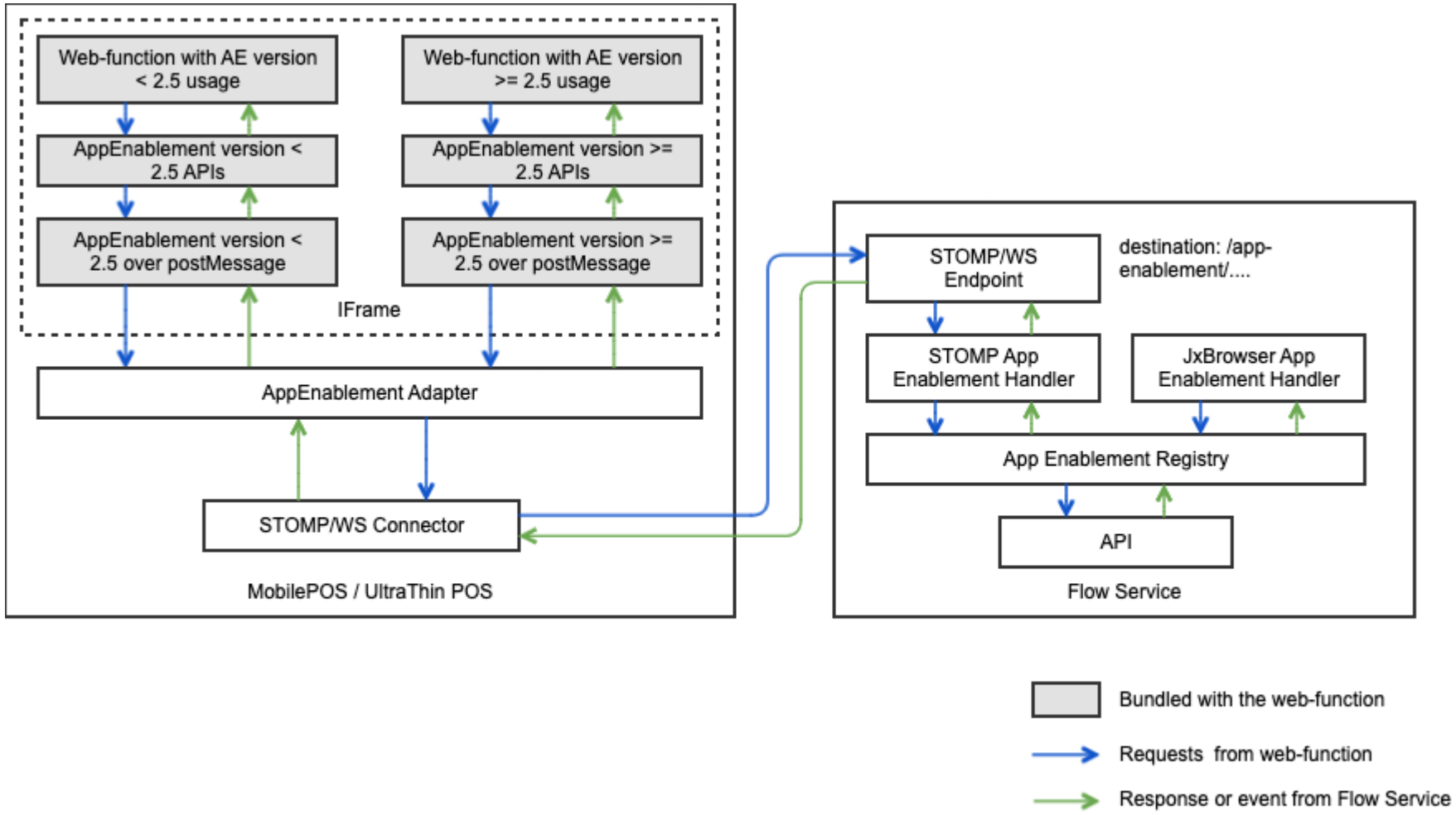
MobilePOS/UltraThin POS & FlowService

App Enablement function runs in UltraThin POS in iFrame. App Enablement APIs and connector are bundled with the web-function. App Enablement APIs are loaded from the server where web-function/client app is hosted (UltraThin POS does not provide them).

UltraThin POS contains adapter for App Enablement. These adapter process postMessages sent from client app in iFrame. Adapter converts the messages and send them to FlowService over STOMP/WS connector. Adapter also receives App Enablement messages from FlowService over STOMP/WS, convert them and pass them to the web-function/client app over App Enablement adapter through postMessage.

FlowService (OPOS) contains App Enablement Handler that takes App Enablement messages from UltraThin POS and triggers the APIs that are registered App Enablement Registry. This is done similarly like the App Enablement processing from JxBrowser. AppEnablement Handler in FlowService ensures to return a response and registered events (web-function may register for specific FlowService events) to UltraThin POS client.

There is a dedicated destination for STOMP/WS communication (e.g. /app-enablement/...) for AppEnablement. The destination /flowengine/... is not reused for it.



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

Supported by opos provider, mobilePOS provider and partially by launchpad provider

Method	Parameters	Description	Example	Since
`oAppEnablement CommonInstance. getSessionContext`	`appCallback: ApiResponse<IContext>`	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: "10000000000000001", businessUnitID: "9090", // Equivalent to store ID isoCurrencyCode: "USD", storeLanguage: "en_US", tenantID: "004", userLanguage: "zh_CN", workstationID: "107" }`	2.0.0
`oAppEnablement CommonInstance. registerListener`	`eventHandler: (data: any) => void, eventName: string, registrationId: string, expectData: boolean`	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. Example request object: `{ "event": "EVENT_TRANSACTION_UPDATED", "listener": "returnEventNotification", "passData": false || true }`	2.0.0
`oAppEnablement CommonInstance. unregisterListener`	`eventName: string`	Removes the listener for the given event.	Unregister the listeners so that the app can log off the event bus. Example request object: `{ "event": "EVENT_TRANSACTION_UPDATED" }`	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	appCallback: ApiResponse<IOperator>	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

Supported by opos provider, mobilePOS only

Method	Parameters	Description	Example	Since

oAppEnablement ExternalMasterdata Instance.get ItemByCriteria	appCallback: ApiResponse<MasterDataDto.ILineItem>, data: MasterDataDto.IExternalItemDataID	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. Example request object:
			{
			query: "camera",
			language: "en_US",
			isoCurrencyCode: "USD"
			}
				2.0.0

oAppEnablement	appCallback: ApiResponse<MasterDataDto.ILineItemList>,	Calls and hands back a list of items from an	Search for items in a connected web shop, display a list of results.	2.0.0

`ExternalMasterdata Instance.getItem ListBySearch Criteria`	`data: MasterDataDto.IExternalItemDataList`	external repository.	Example request object: `{ query: "camera", language: "en_US", isoCurrencyCode: "USD", recordCount: 60 }`	
`oAppEnablement ExternalMasterdata Instance.getImageUrl`	`appCallback: ApiResponse<String> , data: MasterDataDto.IExternalItemImgUrl`	Provides an image URL for external items.	Example request object: `{ itemID: "030039", type: "item", language: "en_US", isoCurrencyCode: "USD" }`	2.0.0

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

Supported by opos provider, mobilePOS provider and partially by launchpad provider

Method	Parameters	Description	Example	Since
`oAppEnablement MasterdataInstance. getItemDataByID`	`appCallback: ApiResponse<MasterDataDto.ILineItem>, data: MasterDataDto.IItemDataID`	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. Example request object: `{ itemID: "030039" }`	2.0.0

`oAppEnablement Masterdata Instance. getItemDataListByIDList`	`appCallback: ApiResponse<MasterDataDto.ILineItemList>, data: MasterDataDto.IItemDataList`	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 result of analysis. Example request object: `{ itemIDList: ["03039", "65005", "65007"] }`	2.0.0
`oAppEnablement MasterdataInstance. getImageUrl`	`appCallback: ApiResponse<string>, data: MasterDataDto.IItemImgUrl`	Returns the image URL for the given item ID.	Use URL to call and display item images. Example request object: `{ itemID: "03039", type: "item" }`	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

Supported by opos provider, mobilePOS provider only

Method	Parameters	Description	Example	Since
`oAppEnablement PosInstance. getPOSItem InformationByID`	`appCallback: ApiResponse<PosDto.IItemInfo>, data: PosDto.IPOSItemInformationByID`	Returns item information for the given item ID. Enriched with additional POS-specific information provided by other services like, for example, stock.	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. Example request object: `{ itemID: "030039" }`	2.0.0

oAppEnablement PosInstance. registerLineItem	appCallback: ApiResponse<String>, data: PosDto.ILineItemRegistration	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. Example request object: `{ "itemID": "030039", "language": "en_US", "isoCurrencyCode": "USD" }`	2.0.0
oAppEnablement PosInstance. registerExternalLineItem	appCallback: ApiResponse<String>, data: PosDto.IRegisterExternalLineItem	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. Example request object: `{ "posItemID":"3636", "itemID":"4711", "unitOfMeasureCode":"PCE", "itemType":"CO", "actualUnitPrice":15.55, "quantity":1, "receiptText":"Test Item", "registrationNumber":"3636", "mainPOSItemID":"3636", "taxGroupID":"A1" }` Required attributes: • quantity • actualUnitPrice • itemID • itemType • mainPOSItemID • posItemID • receiptText • registrationNumber • taxGroupID • unitOfMeasureCode	2.0.0
oAppEnablement PosInstance. getCurrent Transaction	appCallback: ApiResponse<PosDto.ITransaction>	Returns the current transaction.		2.0.0
oAppEnablement PosInstance. getCurrent CustomerList	appCallback: ApiResponse<PosDto.IRetailTransactionCustomer>	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. cancelCurrent Transaction	appCallback: ApiResponse<string>, data: PosDto.ICurrentTransaction Cancellation	Allows to cancel the current transaction in the POS.	Example request object: `{ "reasonCode": "abc", "reasonDescription": "A cancellation." }`	2.0.0
oAppEnablement PosInstance. registerCustomer	appCallback: ApiResponse<string>, data: PosDto.ICustomerRegistration	Allows to register a customer within the current transaction in the POS.	Example request object: `{ "customerId": "10012" }`	2.0.0
oAppEnablement PosInstance. addTransaction Extension	appCallback: ApiResponse<string>, data: PosDto.ITransactionExtensionValueRequest	Allows to add an extension to the current transaction in the POS.	Example request object: `{ "extensionKey": "key", "extensionValue": "val" }`	2.0.0
oAppEnablementPosInstance.updateTransactionExtension	appCallback: ApiResponse<string>, data: PosDto.ITransactionExtensionValueRequest	Allows to update an extension in the current transaction in the POS.	Example request object: `{ "extensionKey": "key", "extensionValue": "val" }`	2.0.0
oAppEnablementPosInstance.deleteTransactionExtension	appCallback: ApiResponse<string>, data: PosDto.ITransactionExtensionRequest	Allows to delete an extension in the current transaction in the POS.	Example request object: `{ "extensionKey": "key" }`	2.0.0
oAppEnablement PosInstance. addAdditional TransactionReport	appCallback: ApiResponse<string>, data: PosDto.IAdditionalTransactionReport	Allows to add an additional receipt to the current transaction in the POS.	Example request object: `{ "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
oAppEnablement PosInstance. enterCoupon	appCallback: ApiResponse<string>, data: PosDto.ICouponRequest	Allows to enter a coupon to the current transaction in the POS.	Example request object:	2.1.0

			<pre>{
			"couponNumber": "1",
			"privilegeType": "RP",
			"privilegeValue": "1"
			}</pre>	

oAppEnablement.PosInstance.getLastNotVoidedTransaction	appCallback: ApiResponse<PosDto.ICompleteTransaction>	Get the last not voided transaction.		2.2.0
oAppEnablement.PosInstance.printReport	appCallback: ApiResponse<string>, data: PosDto.ITransaction	This function will print the given transaction		2.2.0

oAppEnablement.PosInstance.selectItemVariant	appCallback: ApiResponse<void>, data: PosDto.ISelectItemVariantRequest	Allows to select an item variant within the item information in the POS	Example request object:
			<pre>{
			itemID: "030039"
			itemUOMCode: "PCE"
			}</pre>
				2.2.1-alpha.2

oAppEnablement.PosInstance.getPosContext	appCallback: ApiResponse<string>	Returns the POS context.		2.6.0-b11

oAppEnablement.PosInstance.voidLineItemInternally	appCallback: ApiResponse<string>, data: PosDto.IVoidLineItemInternallyRequest	Voids line item internally by retailTransactionLineItemSequenceNumber parameter.	Example request object:
			<pre>{
			"retailTransactionLineItemSequenceNumber": 1
			}</pre>
				2.6.1-b09

1.2.2.5 File Authorization.js

Description

The file provides specialized handling for authorization related errors.

Namespace

`comGkSoftwareGkrAppEnablementApi.Authorization`

Supported methods

Supported by opos provider, mobilePOS provider and launchpad provider

Method	Parameters	Description	Example	Since
oAppEnablement	---	Triggers specialized handling of authorization errors (such as 401),		2.6.0-

AuthorizationInstance. userUnauthorizedError		which will trigger a reload of the page. Initially defined for launchpad use cases.		b07
oAppEnablement AuthorizationInstance. getAuthorizationContext	appCallback: ApiResponse<IAuthorizationContext>	Provides authorization information like token or set of user rights for logged user		

1.2.2.6 File Fuel.js

Description

The file provides specialized handling for fuel related operations.

Namespace

comGkSoftwareGkrAppEnablementApi.Fuel

Supported methods

Supported by opos provider, mobilePOS provider only

Method	Parameters	Description	Example	Since

oAppEnablement
FuelInstance.
registerFuelLineItemFromStack
	appCallback: ApiResponse<string>, data:
	FuelDto.RegisterFuelLineItemFromStackRequest
		Registers fuel line item from stack.	Example request object:
			{
			"units":"6.12",
			"actualUnitPrice":"1.111",
			"fuelingPointId":2,
			"gradeId":1,
			"stackPosition":20000,
			"fuelSaleStatusCode":"POSTPAY",
			"presetAmount":0.0,
			"actualAmount":6.8,
			"fuelDiscount":0.0,
			"customerId":"10004",
			"customerServiceTypeCode":"MASTER_DATA",
			"preferredReceiptPrintoutTypeCode":""
			}
				2.6.0-b11

oAppEnablementFuelInstance. registerPrepayFuelLineItem	appCallback: ApiResponse<string>, data: FuelDto.RegisterPrepayFuelLineItemReques	Registers pre-pay fuel line item.	Example request object:	2.6.0-b11

	t		<pre>{
			"actualUnitPrice":"50.0",
			"fuelingPointId":2,
			"gradeId":1,
			"fuelSaleStatusCode":"PREPAY"
			}</pre>	

oAppEnablement FuelInstance. registerPrepayRestOnPumpFuelLineItem	appCallback: ApiResponse<string>, data: FuelDto.RegisterPrepayFuelLineItemRequest	Registers pre-pay on pump.	{ "actualUnitPrice":"50.0", "fuelingPointId":2, "gradeId":1, "fuelSaleStatusCode":"PREPAY" }	2.6.0 -b11
oAppEnablementFuelInstance. getFuelConfig	appCallback: ApiResponse<FuelDto.FuelConfig>	Returns the fuel configuration		2.6.0 -b11

1.2.2.7 File FunctionAPI.ts

Description

The file provides specialized handling for launchpad functions.

Namespace

`comGkSoftwareGkrAppEnablementApi.Function`

Supported methods

Supported by launchpad provider only

Method	Parameters	Description	Example	Since
oAppEnablement FunctionInstance. getFunctionContext	appCallback: ApiResponse<IFunctionContext>	Provide the context for function to run like functionId or run parameters	Response	2.6.0

			<pre>{
			  "functionId": "123456",
			
			"functionParameters": [
			  {
			  "name": "param1",
			  "value": "value1"
			  }
			  ]
			}</pre>	

oAppEnablement FunctionInstance.getFunctionCache	appCallback: ApiResponse<Transferable>, data: IFunctionCacheData	Loads data stored for current function with given key. Stored data fulfills interface Transferable	<pre>{key: String}</pre> Transferable	2.6.0
oAppEnablement FunctionInstance.setFunctionCache	appCallback: ApiResponse<IFunctionCacheWriteStatus>, data: IFunctionCacheData	Saves data for current function with given key. Stored data fulfills interface Transferable	Request <pre>{ key: String, value: Transferable }</pre> Response <pre>{successful: boolean}</pre>	2.6.0
oAppEnablement FunctionInstance.closeFunction		Closes currently opened function (suicide button)		2.6.0

oAppEnablement FunctionInstance.launchFunction	functionContext: IFunctionContext	Launches other function, current function is called	Request
			<pre>{
			  "functionId": "123456",
			
			"functionParameters": [
			  {
			  "name": "param1",
			  "value": "value1"
			  }
			  ]
			}</pre>
				2.6.0

1.2.2.8 File MessagingAPI.ts

Description

The file provides specialized handling for events from Messaging (currently Event-Service).

Namespace

`comGkSoftwareGkrAppEnablementApi.Messaging`

Supported methods

Supported by launchpad provider only at the moment

Method	Parameters	Description	Example	Since
`oAppEnablement MessagingInstance.registerEventListener`	`appCallback: ApiResponse<IEvent[]>`	Register function to receive events from Messaging (currently Event-Service). It work similar way like scanner, you register callback and it is called every time event is received. For the first time you'll get all messages queued for your function.	Response `[{ "topic": "abcdefg", "headers": { "id": "id", "header2": "aaa" }, "content": "particular event content" }]`	2.6.0
`oAppEnablement MessagingInstance.sendEvent`	`appCallback: () => void, data: IEvent`	Send event outside of the function	Request `{ "topic": "abcdefg", "headers": { "id": "id", "header2": "aaa" }, "content": "particular event content" }`	2.6.0

1.2.2.9 File ScannerApi.ts

Description

The file provides specialized handling for built-in/connected scanners/camera scanner on mobile clients.

Namespace

`comGkSoftwareGkrAppEnablementApi.Scanner`

Supported methods

Supported by launchpad provider only at the moment

Method	Parameters	Description	Example	Since
`oAppEnablement ScannerInstance.getAvailableScannerTypes`	`appCallback: ApiResponse<[IScannerType]>`	Provides list of available scanners. Usually CAMERA and/or HW scanner. Method returns empty array or null/undefined in case that scanner is not supported at all	Response `["CAMERA", "HW"]`	2.6.0
`oAppEnablement ScannerInstance.openCameraScanner`	`appCallback: ApiResponse<IScanned>`	Opens camera scanner window, once some object (barcode, QR code) is scanned it return value via callback	Response `{scanned: "1234567890"}`	2.6.0

1.2.3 Supported Events

The AppEnablement 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.

For the OPOS client and the new MPOS client (based on FlowService) you will find the list of possible events in com.gk_software.pos.api.messagebus.MessageConstants.

The list of send events is enhanced for each OPOS/Flowservice version and depends on the configuration of the application and the use cases.

As an example the app can register on the "EVENT_TRANSACTION_UPDATED" events, which is fired, when the transaction is updated.

(Please note, that the old MPOS client only sends this EVENT_TRANSACTION_UPDATED event).

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

1.3.1 Implementation details

The following sections describe the technical realisation of the AppEnablement 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: on`ScriptContextCreated` (which is invoked when a JavaScript context has been created) and on`ScriptContextDestroyed` (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 `posjs://` , 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
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:
        i. 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>
```

        ii. 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 Quick Start Example Apps

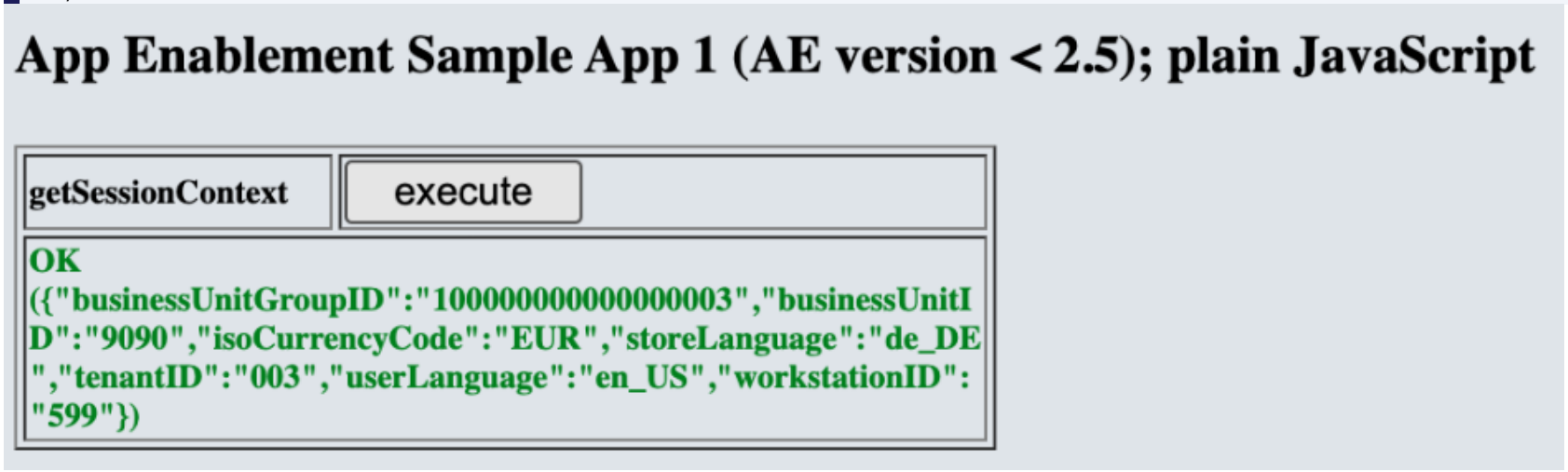
2.1.1 Sample app 1 (before version 2.5)

Use @gk-software/app-enablement-adapter. Sample is written in plain JavaScript.

The following example shows how to create an app and how to use the AppEnablement 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/AppEnablementConnector.js"></script>
    <script type="text/javascript" src="./libs/api/Common.js"></script>
    <script type="text/javascript">
        var oAppEnablementCommonInstance = new comGkSoftwareGkrAppEnablementApi.Common();
        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 + ')');
        }
        // for test purpose only
        window.addEventListener('message', event => {
            console.log('received message', event && event.data ? event.data : 'Unknown event')
        }, false)
    </script>
</head>
<body style="margin:0; padding:0;">
<div style="font-size:110%; font-weight:bold; padding-bottom:20px;">
    App Enablement Sample App 1 (AE version < 2.5); plain JavaScript
</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>
```



@gk-software/app-enablement-adapter is only for existing projects. Maintenance only. No new API.

2.1.2 Sample app 1 (from version 2.5)

Use @gk-app-enablement/api-definition and @gk-app-enablement/app-messaging-provider. Sample is written in TypeScript.

The following example shows how to create an app and how to use the AppEnablement API. TypeScrip code has to be compiled.

```
<!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="sample-app.js"></script>
    <script type="text/javascript">
        const commonApi = SampleApp.init()
        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 setHTML(id, html) {
            if (document.getElementById(id) != null) {
                document.getElementById(id).innerHTML = html;
            }
        }
        // getSessionContext
        function getSessionContext() {
            commonApi.getSessionContext((context => {
                currentSessionContextFound(context)
            }))
        }
        function currentSessionContextFound(context) {
            this.context = context;
            good('statusGetSessionContext', context);
        }
        // for test purpose only
        window.addEventListener('message', event => {
            console.log('received message', event && event.data ? event.data : 'Unknown event')
        }, false)
    </script>
</head>
<body style="margin:0; padding:0;">
<div style="font-size:110%; font-weight:bold; padding-bottom:20px;">
    App Enablement Sample App 1 (AE version >= 2.5); TypeScript
</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>
```

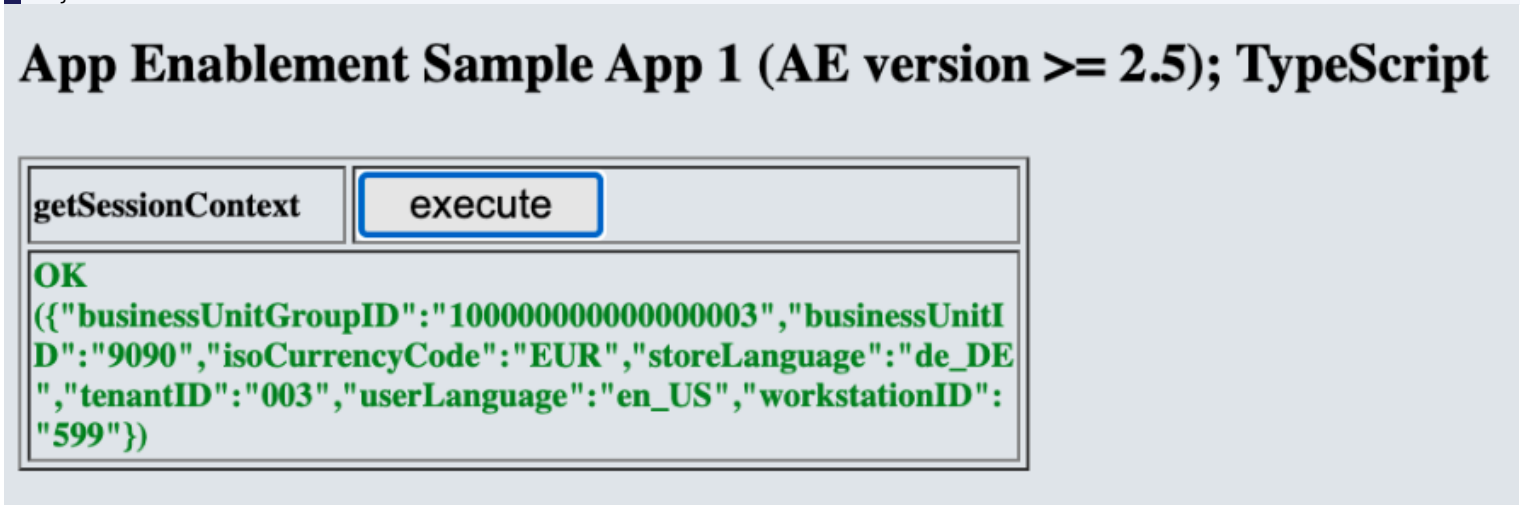
```
</table>
</body>
</html>
```

```
import * as AppEnablement from "@gk-app-enablement/app-messaging-provider";

export function init(): AppEnablement.CommonApi.ICommonApi {

    const ApiEnablement: AppEnablement.IAppEnablement = AppEnablement.EnablementProvider.getEnablementApi();
    const commonApi: AppEnablement.CommonApi.ICommonApi = ApiEnablement.getApiByName(AppEnablement.CommonApi.apiName) as AppEnablement.CommonApi.ICommonApi;

    return commonApi
}
```



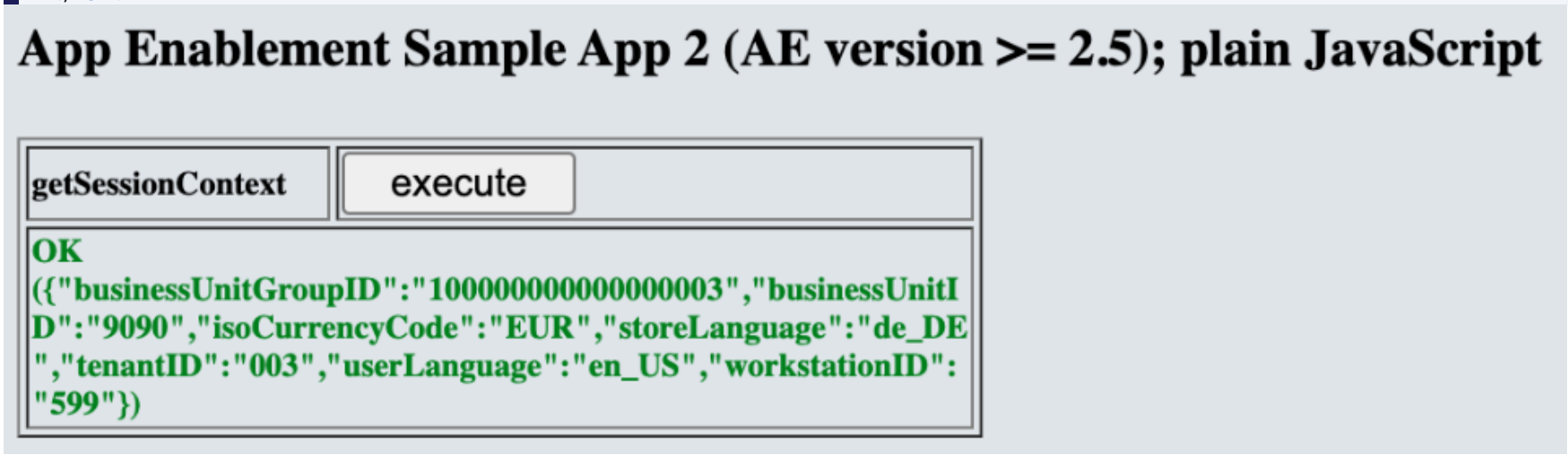
2.1.3 Sample app 2 (from version 2.5)

Use @gk-app-enablement/api-definition and @gk-app-enablement/app-messaging-provider. Sample is written in plain JavaScript.

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/api-definition/api-definition.js"></script>
    <script type="text/javascript" src="./libs/app-messaging-provider/messaging-adapter.js"></script>
    <script type="text/javascript">
        const ApiEnablement = AppEnablementMessaging.EnablementProvider.getEnablementApi();
        const commonApi = ApiEnablement.getApiByName(AppEnablementMessaging.CommonApi.apiName);
        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 setHTML(id, html) {
            if (document.getElementById(id) != null) {
                document.getElementById(id).innerHTML = html;
            }
        }
        // getSessionContext
        function getSessionContext() {
            commonApi.getSessionContext((context => {
                currentSessionContextFound(context)
            }))
        }
        function currentSessionContextFound(context) {
            this.context = context;
            good('statusGetSessionContext', context);
        }
        // for test purpose only
        window.addEventListener('message', event => {
            console.log('received message', event && event.data ? event.data : 'Unknown event')
        }, false)
    </script>
</head>
<body style="margin:0; padding:0;">
<div style="font-size:110%; font-weight:bold; padding-bottom:20px;">
    App Enablement Sample App 2 (AE version >= 2.5); plain JavaScript
</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>
```



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 3 Example message for scanned EAN13 barcode with value 4002919000115

In case that a POS version less than 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.*

3 Configuration Options

3.1 POS Client (Full Client/Thin Client)

In this section we will check how to configure BrowserApp in POS Client (Full Client/Thin Client)

3.1.1 jxBrowserAppConfigs.properties

File in *config/standard/parameter/client*.

Configuration of ***customApp1*** BrowserApp.

Purpose of the configuration file:

- it defines browserApp specific configurations
    - keepListenersEnabled true|false defines if registered AppEnablement listeners are enabled|disabled when browserApp is in background
    - reloadAlways true|false defines if browserApp content state has to be saved
- referenced in ui config *UiConfig.jxBrowserConfig.appConfigs.**<appId>*** (see next chapter)
- used in JXBrowser implementation to configure the browserApp behavior

Add to the file.

```
JxBrowserAppConfigs.customApp1=include:JxBrowserAppConfigs.Default
JxBrowserAppConfigs.customApp1.keepListenersEnabled=true
JxBrowserAppConfigs.customApp1.reloadAlways=true
```

3.1.2 ui.properties

File in *config/standard/parameter/client*.

Configuration of ***customApp1*** BrowserApp.

Purpose of configuration file:

- used for jxBrowser implementation to get app configs (*UiConfig.jxBrowserConfig.appConfigs.**<appId>***)

Add to the file.

```
UiConfig.jxBrowserConfig.appConfigs.customApp1=include:JxBrowserAppConfigs.customApp1
```

*UiConfig.jxBrowserConfig.appConfigs.**customApp1*** has relation to *JxBrowserAppConfigs.**customApp1*** from previous configuration file.

3.1.3 browserConfigs.properties

File in *config/standard/parameter/client/flow*.

Configuration of ***customApp1*** BrowserApp.

Purpose of the configuration file:

- config for browser process
- basic config to specify appId and browserUrl
    - url defines the app url to load
    - appId defines app id by which to retrieve app config (*UiConfig.jxBrowserConfig.appConfigs.**<appId>*** in ui.properties)
    - fullScreen true|false defines if show browser toolbar or not
    - setSSOToken true|false defines if GKSSOToken has to be set in browser cookies

Add to the file.

```
BrowserConfigs.customApp1=include:BrowserConfigs.Default
BrowserConfigs.customApp1.url=http://localhost:8081/app-debug/build/index.html
BrowserConfigs.customApp1.appId=customApp1
BrowserConfigs.customApp1.fullScreen=true
BrowserConfigs.customApp1.setSSOToken=true
```

Because of compatibility, if client app is using AppEnablement before 2.5, url is http://localhost:8081/app-debug/build/index.html. if client app is using AppEnablement from 2.5, url is iframe:http://localhost:8081/app-debug/build/index.html

3.1.4 launchAppConfigs.properties

File in *config/standard/parameter/client/flow*.

Configuration of ***customApp1*** BrowserApp.

Purpose of the configuration file:

- config for launchApp process
- it specifies some process specific configs + reference to a browser config (*BrowserConfigs.**<appId>***)
- the process is a wrapper around our browser process
- it was implemented for dealing with launchpad apps
- it includes CentralAppLogin and CentralAppLogout (depends on config)

Add to the file.

```
LaunchAppConfigs.customApp1=include:LaunchAppConfigs.Default

#  Allowed in logged in but requires POS Logout.
#           So POS Logout will be part of this process.
#           After logout it will open the app.
#           Session for the app will be kept.
LaunchAppConfigs.customApp1.requiresWorkstationLoggedOut=false

# browser config for custom app 1
LaunchAppConfigs.customApp1.browserConfig=include:BrowserConfigs.customApp1
```

3.1.5 Tableau button

File in *config/standard/parameter/client/tableau-data*.

TableauPosition configuration to trigger process (*browser* → BrowserConfigs.**<appId>** or *launchApp* → LaunchAppConfigs.**<appId>**.browserConfig) which open browser, which can show app with AppEnablement support.

Add to the file.

3.1.5.1 Browser Process

For browser process.

```
{
  "com.gk_software.gkr.pos.tableau.api.model.TableauPosition": {
    "id": "3c430f2a-b12a-4e02-8adf-9067b801d250",
    "page": 0,
    "row": 0,
    "col": 0,
    "rowSpan": 1,
    "colSpan": 1,
    "templateId": "buttonVariantNonary1",
    "tableauPositionTranslationKey": "app1",
    "tableauPositionTranslationList": null,
    "tableauPositionImageList": [
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "de_DE",
          "source": null
        }
      },
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "en_US",
          "source": null
        }
      }
    ],
    "tableauPositionAction": {
      "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
        "processReactionName": "browser",
        "processConfigId": "BrowserConfigs.customApp1",
        "processConfigParameters": [],
        "destinationTableauLevelId": null,
        "destinationTableauAssignmentList": null
      }
    }
  }
}
```


3.1.5.2 LaunchApp Process

For launchApp process.

```
{
  "com.gk_software.gkr.pos.tableau.api.model.TableauPosition": {
    "id": "3c430f2a-b12a-4e02-8adf-9067b801d250",
    "page": 0,
    "row": 0,
    "col": 0,
    "rowSpan": 1,
    "colSpan": 1,
    "templateId": "buttonVariantNonary1",
    "tableauPositionTranslationKey": "app1",
    "tableauPositionTranslationList": null,
    "tableauPositionImageList": [
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "de_DE",
          "source": null
        }
      },
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "en_US",
          "source": null
        }
      }
    ],
    "tableauPositionAction": {
      "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
        "processReactionName": "processLaunchApp",
        "processConfigId": "LaunchAppConfigs.customApp1",
        "processConfigParameters": [],
        "destinationTableauLevelId": null,
        "destinationTableauAssignmentList": null
      }
    }
  }
}
```


Config overrides

It is possible to override browser config in TableauPosition. It has to be added in TableauPosition.tableauPositionAction.processConfigParameters.

See an example

```
      "tableauPositionAction": {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
          "processReactionName": "processLaunchApp",
          "processConfigId": "BrowserConfigs.customApp1",
          "processConfigParameters": [
              {
                  "com.gk_software.gkr.pos.tableau.api.model.TableauProcessConfigParameter": {
                      "key": "url",
                      "value": "http://localhost:3033/build"
                  }
              }
          ],
          "destinationTableauLevelId": null,
          "destinationTableauAssignmentList": null
        }
      }
```

3.1.6 Embedded browserApp

Next code snipped will add embedded browserApp into the POS client view (ItemRegistration, Payment, etc.).

id="customApp1" will be used to retrieve appConfig (*UiConfig.jxBrowserConfig.appConfigs.**<appId>*** in ui.properties).

If for device appId there is no appConfig defined, default appConfig is returned (*JxBrowserAppConfigs.**Default*** in jxBrowserAppConfigs.properties).

It is implemented in generic way, so url from ui.properties and appConfig from jxBrowserAppConfigs.properties will be retrieved automatically.

```
<div id="appAreaComponent" class="appArea" constraints="Center" slot="true" layout="wrapLayout">
    <div id="appAreaSubElement1" class="appContainer">
        <app id="customApp1" url="<app url>" class="app" />
    </div>
</div>
```

Url can be bind from ***UiConfig.customApp1Url***, when specified. So code snipped will look like

```
<div id="appAreaComponent" class="appArea" constraints="Center" slot="true" layout="wrapLayout">
    <div id="appAreaSubElement1" class="appContainer">
        <app id="customApp1" url="UiConfig.customApp1Url" class="app" />
    </div>
</div>
```

3.2 MobilePOS

In this section we will check how to configure BrowserApp for MobilePOS Client.

Standard OPOS has to be started with enabled FlowService.

Configuration and start of FlowService is not covered by this chapter. Installation, configuration and start of MobilePOS client is not covered by this chapter.

3.2.1 ui.properties

File in *config/standard/parameter/client*.

Configuration of ***customApp2*** BrowserApp.

Purpose of configuration file:

- used to provide app configs to mobile client (*UiConfig.appConfigs.*)

Add to the file.

```
UiConfig.appConfigs.0.id=customApp2
UiConfig.appConfigs.0.reloadAlways=false
UiConfig.appConfigs.0.keepListenersEnabled=false
```

3.2.2 browserConfigs.properties

File in config/standard/parameter/flowService/flow.

Configuration of ***customApp2*** BrowserApp.

Purpose of the configuration file:

- config for browser process
- basic config to specify appId and browserUrl
    - url defines the app url to load
    - appId defines app id by which to retrieve app config (*UiConfig.appConfigs.<number>.id=**<appId>*** in ui.properties)
    - fullScreen true|false defines if show browser toolbar or not
    - setSSOToken true|false defines if GKSSOToken has to be set in iFrame cookies

Add to the file.

```
BrowserConfigs.customApp2=include:BrowserConfigs.Default
BrowserConfigs.customApp2.url=http://localhost:8082/app-debug/build/index.html
BrowserConfigs.customApp2.appId=customApp2
BrowserConfigs.customApp2.fullScreen=true
BrowserConfigs.customApp2.setSSOToken=false
```

iframe prefix is url is not the case for MobilePOS. If client app is using AppEnablement before/from 2.5, url is http://localhost:8082/app-debug/build/index.html.

3.2.3 launchAppConfigs.properties

File in config/standard/parameter/flowService/flow.

Configuration of ***customApp2*** BrowserApp.

Purpose of the configuration file:

- config for launchApp process
- it specifies some process specific configs + reference to a browser config (*BrowserConfigs.**<appId>***)
- the process is a wrapper around our browser process
- it was implemented for dealing with launchpad apps
- it includes CentralAppLogin and CentralAppLogout (depends on config)

Add to the file.

```
LaunchAppConfigs.customApp2=include:LaunchAppConfigs.Default

#  Allowed in logged in but requires POS Logout.
#           So POS Logout will be part of this process.
#           After logout it will open the app.
#           Session for the app will be kept.
LaunchAppConfigs.customApp1.requiresWorkstationLoggedOut=false

# browser config for custom app 2
LaunchAppConfigs.customApp2.browserConfig=include:BrowserConfigs.customApp2
```

3.2.4 Tableau button

File in *config/standard/parameter/client/tableau-data*.

TableauPosition configuration to trigger process (*browser* → BrowserConfigs.**<appId>** or *launchApp* → LaunchAppConfigs.**<appId>**.browserConfig) which open browser, which can show app with AppEnablement support.

Add to the file.

3.2.4.1 Browser Process

For browser process.

```
{
  "com.gk_software.gkr.pos.tableau.api.model.TableauPosition": {
    "id": "3c430f2a-b12a-4e02-8adf-9067b801d250",
    "page": 0,
    "row": 0,
    "col": 0,
    "rowSpan": 1,
    "colSpan": 1,
    "templateId": "buttonVariantNonary1",
    "tableauPositionTranslationKey": "app1",
    "tableauPositionTranslationList": null,
    "tableauPositionImageList": [
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "de_DE",
          "source": null
        }
      },
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "en_US",
          "source": null
        }
      }
    ],
    "tableauPositionAction": {
      "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
        "processReactionName": "browser",
        "processConfigId": "BrowserConfigs.customApp2",
        "processConfigParameters": [],
        "destinationTableauLevelId": null,
        "destinationTableauAssignmentList": null
      }
    }
  }
}
```


3.2.4.2 LaunchApp Process

For launchApp process.

```
{
  "com.gk_software.gkr.pos.tableau.api.model.TableauPosition": {
    "id": "3c430f2a-b12a-4e02-8adf-9067b801d250",
    "page": 0,
    "row": 0,
    "col": 0,
    "rowSpan": 1,
    "colSpan": 1,
    "templateId": "buttonVariantNonary1",
    "tableauPositionTranslationKey": "app1",
    "tableauPositionTranslationList": null,
    "tableauPositionImageList": [
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "de_DE",
          "source": null
        }
      },
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "en_US",
          "source": null
        }
      }
    ],
    "tableauPositionAction": {
      "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
        "processReactionName": "processLaunchApp",
        "processConfigId": "LaunchAppConfigs.customApp2",
        "processConfigParameters": [],
        "destinationTableauLevelId": null,
        "destinationTableauAssignmentList": null
      }
    }
  }
}
```


Config overrides

It is possible to override browser config in TableauPosition. It has to be added in TableauPosition.tableauPositionAction.processConfigParameters.

See an example

```
      "tableauPositionAction": {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
          "processReactionName": "processLaunchApp",
          "processConfigId": "BrowserConfigs.customApp2",
          "processConfigParameters": [
              {
                  "com.gk_software.gkr.pos.tableau.api.model.TableauProcessConfigParameter": {
                      "key": "url",
                      "value": "http://localhost:3055/build"
                  }
              }
          ],
          "destinationTableauLevelId": null,
          "destinationTableauAssignmentList": null
        }
      }
```

3.2.5 Embedded browserApp

Next code snipped will add embedded browserApp into the MobilePOS client view (ItemRegistration, Payment, etc.).

appConfig is retrieved from *(UiConfig.appConfigs.<number>* in ui.properties where id is wanted appId).

If for appId there is no appConfig defined, default appConfig is returned (*UiConfig.appConfigs.0* in ui.properties).

Function to retrieve appConfig need to be triggered before view with ***gk-browser*** will be rendered. Each value possible to bind.

```
  <gk-browser id="app1Id" :src="url" appEnablementEnabled  appId="embeddedApp1" appToken="waL-JADK1Qp0Fi5R5_NJH1" :posToken="deviceId"
  :reloadAlways="reloadAlways" :keepListeners="keepListenersEnabled" posOrigin="*" ref="embeddedApp1"></gk-browser>
```

3.3 MobilePOS

In this section we will check how to configure BrowserApp for MobilePOS Client.

Standard OPOS has to be started with enabled FlowService.

Configuration and start of FlowService is not covered by this chapter. Installation, configuration and start of MobilePOS client is not covered by this chapter.

3.3.1 ui.properties

File in *config/standard/parameter/client*.

Configuration of ***customApp2*** BrowserApp.

Purpose of configuration file:

- used to provide app configs to mobile client (*UiConfig.appConfigs.*)

Add to the file.

```
UiConfig.appConfigs.0.id=customApp2
UiConfig.appConfigs.0.reloadAlways=false
UiConfig.appConfigs.0.keepListenersEnabled=false
```

3.3.2 browserConfigs.properties

File in config/standard/parameter/flowService/flow.

Configuration of ***customApp2*** BrowserApp.

Purpose of the configuration file:

- config for browser process
- basic config to specify appId and browserUrl
    - url defines the app url to load
    - appId defines app id by which to retrieve app config (*UiConfig.appConfigs.<number>.id=**<appId>*** in ui.properties)
    - fullScreen true|false defines if show browser toolbar or not
    - setSSOToken true|false defines if GKSSOToken has to be set in iFrame cookies

Add to the file.

```
BrowserConfigs.customApp2=include:BrowserConfigs.Default
BrowserConfigs.customApp2.url=http://localhost:8082/app-debug/build/index.html
BrowserConfigs.customApp2.appId=customApp2
BrowserConfigs.customApp2.fullScreen=true
BrowserConfigs.customApp2.setSSOToken=false
```

iframe prefix is url is not the case for MobilePOS. If client app is using AppEnablement before/from 2.5, url is http://localhost:8082/app-debug/build/index.html.

3.3.3 launchAppConfigs.properties

File in config/standard/parameter/flowService/flow.

Configuration of ***customApp2*** BrowserApp.

Purpose of the configuration file:

- config for launchApp process

- it specifies some process specific configs + reference to a browser config (*BrowserConfigs.**<appId>***)
- the process is a wrapper around our browser process
- it was implemented for dealing with launchpad apps
- it includes CentralAppLogin and CentralAppLogout (depends on config)

Add to the file.

```
LaunchAppConfigs.customApp2=include:LaunchAppConfigs.Default

#  Allowed in logged in but requires POS Logout.
#           So POS Logout will be part of this process.
#           After logout it will open the app.
#           Session for the app will be kept.
LaunchAppConfigs.customApp1.requiresWorkstationLoggedOut=false

# browser config for custom app 2
LaunchAppConfigs.customApp2.browserConfig=include:BrowserConfigs.customApp2
```

3.3.4 Tableau button

File in *config/standard/parameter/client/tableau-data*.

TableauPosition configuration to trigger process (*browser* → BrowserConfigs.**<appId>** or *launchApp* → LaunchAppConfigs.**<appId>**.browserConfig) which open browser, which can show app with AppEnablement support.

Add to the file.

3.3.4.1 Browser Process

For browser process.

```
{
  "com.gk_software.gkr.pos.tableau.api.model.TableauPosition": {
    "id": "3c430f2a-b12a-4e02-8adf-9067b801d250",
    "page": 0,
    "row": 0,
    "col": 0,
    "rowSpan": 1,
    "colSpan": 1,
    "templateId": "buttonVariantNonary1",
    "tableauPositionTranslationKey": "app1",
    "tableauPositionTranslationList": null,
    "tableauPositionImageList": [
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "de_DE",
          "source": null
        }
      },
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "en_US",
          "source": null
        }
      }
    ],
    "tableauPositionAction": {
      "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
        "processReactionName": "browser",
        "processConfigId": "BrowserConfigs.customApp2",
        "processConfigParameters": [],
        "destinationTableauLevelId": null,
        "destinationTableauAssignmentList": null
      }
    }
  }
}
```


3.3.4.2 LaunchApp Process

For launchApp process.

```
{
  "com.gk_software.gkr.pos.tableau.api.model.TableauPosition": {
    "id": "3c430f2a-b12a-4e02-8adf-9067b801d250",
    "page": 0,
    "row": 0,
    "col": 0,
    "rowSpan": 1,
    "colSpan": 1,
    "templateId": "buttonVariantNonary1",
    "tableauPositionTranslationKey": "app1",
    "tableauPositionTranslationList": null,
    "tableauPositionImageList": [
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "de_DE",
          "source": null
        }
      },
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "en_US",
          "source": null
        }
      }
    ],
    "tableauPositionAction": {
      "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
        "processReactionName": "processLaunchApp",
        "processConfigId": "LaunchAppConfigs.customApp2",
        "processConfigParameters": [],
        "destinationTableauLevelId": null,
        "destinationTableauAssignmentList": null
      }
    }
  }
}
```


Config overrides

It is possible to override browser config in TableauPosition. It has to be added in TableauPosition.tableauPositionAction.processConfigParameters.

See an example

```
"tableauPositionAction": {
  "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
    "processReactionName": "processLaunchApp",
    "processConfigId": "BrowserConfigs.customApp2",
    "processConfigParameters": [
        {
            "com.gk_software.gkr.pos.tableau.api.model.TableauProcessConfigParameter": {
                "key": "url",
                "value": "http://localhost:3055/build"
            }
        }
    ],
    "destinationTableauLevelId": null,
    "destinationTableauAssignmentList": null
  }
}
```

3.3.5 Embedded browserApp

Next code snipped will add embedded browserApp into the MobilePOS client view (ItemRegistration, Payment, etc.).

appConfig is retrieved from *(UiConfig.appConfigs.<number>* in ui.properties where id is wanted appId).

If for appId there is no appConfig defined, default appConfig is returned (*UiConfig.appConfigs.0* in ui.properties).

Function to retrieve appConfig need to be triggered before view with ***gk-browser*** will be rendered. Each value possible to bind.

```
<gk-browser id="app1Id" :src="url" appEnablementEnabled  appId="embeddedApp1" appToken="waL-JADK1Qp0Fi5R5_NJH1" :posToken="deviceId"
:reloadAlways="reloadAlways" :keepListeners="keepListenersEnabled" posOrigin="*" ref="embeddedApp1"></gk-browser>
```

3.4 POS Client (Full Client/Thin Client)

In this section we will check how to configure BrowserApp in POS Client (Full Client/Thin Client)

3.4.1 jxBrowserAppConfigs.properties

File in *config/standard/parameter/client*.

Configuration of ***customApp1*** BrowserApp.

Purpose of the configuration file:

- it defines browserApp specific configurations
    - keepListenersEnabled true|false defines if registered AppEnablement listeners are enabled|disabled when browserApp is in background

    - reloadAlways true|false defines if browserApp content state has to be saved
- referenced in ui config *UiConfig.jxBrowserConfig.appConfigs.**<appId>*** (see next chapter)
- used in JXBrowser implementation to configure the browserApp behavior

Add to the file.

```
JxBrowserAppConfigs.customApp1=include:JxBrowserAppConfigs.Default
JxBrowserAppConfigs.customApp1.keepListenersEnabled=true
JxBrowserAppConfigs.customApp1.reloadAlways=true
```

3.4.2 ui.properties

File in *config/standard/parameter/client*.

Configuration of ***customApp1*** BrowserApp.

Purpose of configuration file:

- used for jxBrowser implementation to get app configs (*UiConfig.jxBrowserConfig.appConfigs.**<appId>***)

Add to the file.

```
UiConfig.jxBrowserConfig.appConfigs.customApp1=include:JxBrowserAppConfigs.customApp1
```

*UiConfig.jxBrowserConfig.appConfigs.**customApp1*** has relation to *JxBrowserAppConfigs.**customApp1*** from previous configuration file.

3.4.3 browserConfigs.properties

File in *config/standard/parameter/client/flow*.

Configuration of ***customApp1*** BrowserApp.

Purpose of the configuration file:

- config for browser process
- basic config to specify appId and browserUrl
    - url defines the app url to load
    - appId defines app id by which to retrieve app config (*UiConfig.jxBrowserConfig.appConfigs.**<appId>*** in ui.properties)
    - fullScreen true|false defines if show browser toolbar or not
    - setSSOToken true|false defines if GKSSOToken has to be set in browser cookies

Add to the file.

```
BrowserConfigs.customApp1=include:BrowserConfigs.Default
BrowserConfigs.customApp1.url=http://localhost:8081/app-debug/build/index.html
BrowserConfigs.customApp1.appId=customApp1
BrowserConfigs.customApp1.fullScreen=true
BrowserConfigs.customApp1.setSSOToken=true
```

Because of compatibility, if client app is using AppEnablement before 2.5, url is http://localhost:8081/app-debug/build/index.html. if client app is using AppEnablement from 2.5, url is iframe:http://localhost:8081/app-debug/build/index.html

3.4.4 launchAppConfigs.properties

File in *config/standard/parameter/client/flow*.

Configuration of ***customApp1*** BrowserApp.

Purpose of the configuration file:

- config for launchApp process
- it specifies some process specific configs + reference to a browser config (*BrowserConfigs.**<appId>***)
- the process is a wrapper around our browser process
- it was implemented for dealing with launchpad apps
- it includes CentralAppLogin and CentralAppLogout (depends on config)

Add to the file.

```
LaunchAppConfigs.customApp1=include:LaunchAppConfigs.Default

#  Allowed in logged in but requires POS Logout.
#           So POS Logout will be part of this process.
#           After logout it will open the app.
#           Session for the app will be kept.
LaunchAppConfigs.customApp1.requiresWorkstationLoggedOut=false

# browser config for custom app 1
LaunchAppConfigs.customApp1.browserConfig=include:BrowserConfigs.customApp1
```

3.4.5 Tableau button

File in *config/standard/parameter/client/tableau-data*.

TableauPosition configuration to trigger process (*browser* → BrowserConfigs.**<appId>** or *launchApp* → LaunchAppConfigs.**<appId>**.browserConfig) which open browser, which can show app with AppEnablement support.

Add to the file.

3.4.5.1 Browser Process

For browser process.

```
{
  "com.gk_software.gkr.pos.tableau.api.model.TableauPosition": {
    "id": "3c430f2a-b12a-4e02-8adf-9067b801d250",
    "page": 0,
    "row": 0,
    "col": 0,
    "rowSpan": 1,
    "colSpan": 1,
    "templateId": "buttonVariantNonary1",
    "tableauPositionTranslationKey": "app1",
    "tableauPositionTranslationList": null,
    "tableauPositionImageList": [
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "de_DE",
          "source": null
        }
      },
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "en_US",
          "source": null
        }
      }
    ],
    "tableauPositionAction": {
      "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
        "processReactionName": "browser",
        "processConfigId": "BrowserConfigs.customApp1",
        "processConfigParameters": [],
        "destinationTableauLevelId": null,
        "destinationTableauAssignmentList": null
      }
    }
  }
}
```

3.4.5.2 LaunchApp Process

For launchApp process.

```
{
  "com.gk_software.gkr.pos.tableau.api.model.TableauPosition": {
    "id": "3c430f2a-b12a-4e02-8adf-9067b801d250",
    "page": 0,
    "row": 0,
    "col": 0,
    "rowSpan": 1,
    "colSpan": 1,
    "templateId": "buttonVariantNonary1",
    "tableauPositionTranslationKey": "app1",
    "tableauPositionTranslationList": null,
    "tableauPositionImageList": [
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "de_DE",
          "source": null
        }
      },
      {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionImage": {
          "languageId": "en_US",
          "source": null
        }
      }
    ],
    "tableauPositionAction": {
      "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
        "processReactionName": "processLaunchApp",
        "processConfigId": "LaunchAppConfigs.customApp1",
        "processConfigParameters": [],
        "destinationTableauLevelId": null,
        "destinationTableauAssignmentList": null
      }
    }
  }
}
```


Config overrides

It is possible to override browser config in TableauPosition. It has to be added in TableauPosition.tableauPositionAction.processConfigParameters.

See an example

```
      "tableauPositionAction": {
        "com.gk_software.gkr.pos.tableau.api.model.TableauPositionAction": {
          "processReactionName": "processLaunchApp",
          "processConfigId": "BrowserConfigs.customApp1",
          "processConfigParameters": [
              {
                  "com.gk_software.gkr.pos.tableau.api.model.TableauProcessConfigParameter": {
                      "key": "url",
                      "value": "http://localhost:3033/build"
                  }
              }
          ],
          "destinationTableauLevelId": null,
          "destinationTableauAssignmentList": null
        }
      }
```

3.4.6 Embedded browserApp

Next code snipped will add embedded browserApp into the POS client view (ItemRegistration, Payment, etc.).

id="customApp1" will be used to retrieve appConfig (*UiConfig.jxBrowserConfig.appConfigs.**<appId>*** in ui.properties).

If for device appId there is no appConfig defined, default appConfig is returned (*JxBrowserAppConfigs.**Default*** in jxBrowserAppConfigs.properties).

It is implemented in generic way, so url from ui.properties and appConfig from jxBrowserAppConfigs.properties will be retrieved automatically.

```
<div id="appAreaComponent" class="appArea" constraints="Center" slot="true" layout="wrapLayout">
    <div id="appAreaSubElement1" class="appContainer">
        <app id="customApp1" url="<app url>" class="app" />
    </div>
</div>
```

Url can be bind from ***UiConfig.customApp1Url***, when specified. So code snipped will look like

```
<div id="appAreaComponent" class="appArea" constraints="Center" slot="true" layout="wrapLayout">
    <div id="appAreaSubElement1" class="appContainer">
        <app id="customApp1" url="UiConfig.customApp1Url" class="app" />
    </div>
</div>
```

CONTACT

GK Software SE
Waldstraße 7
08261 Schöneck
Germany

T +49 (0) 3 74 64 84 – 0
F +49 (0) 3 74 64 84 – 15
documentation@gk-software.com
www.gk-software.com