



CAIRO-soft sp. z o.o.
NIP 739-360-94-27
ul. Lubelska 36
10-408 Olsztyn
tel. 089 533 95 07
cairo@cairo.pl www.cairo.pl

KRS 0000275085, kapitał: 100 000 zł, Sąd Rejonowy w Olsztynie, VIII Wydział Gospodarczy

FALCON5 WEBSERVICE DOCUMENTATION v 1.23 (EN)

TABLE OF CONTENTS

CHANGELOG	4
RULES OF WORKING WITH SESSIONS	11
COMMUNICATION SCHEMA	12
COMMUNICATION EXAMPLE	13
ERROR CODES	17
LEGEND	18
AVAILABLE METHODS	19
GENERAL	22
getVersion.....	22
getMyCustomerInfo.....	23
doLogin.....	27
getMyBonuses.....	28
doSetRodoStatus.....	30
getTree.....	31
PRODUCTS	32
getProductsInfo.....	32
getProductFileUrl.....	37
getProductStockChanges.....	38
doSearchProducts.....	39
getReplacements.....	40
doBuyNow.....	41
COMPLAINTS	43
doCreateComplaint.....	43
getMyComplaints.....	45
getComplaintQuantityAvailable.....	47
getInvoicesComplaintProduct.....	48
doComplaintPrint.....	49
RESERVATION	50
doOrderProducts.....	50
getMyOrders.....	54
getOrderStockInfo.....	57
doOrderClose.....	59
doOrderItemDelete.....	60
doOrderItemEdit.....	61
doOrderEdit.....	62
doOrderDelete.....	64
doOrderItemsMove.....	65
getOrderStatus.....	66
DOCUMENTS	67
getMyInvoices.....	67
getEDocument.....	72
doDocumentPrint.....	73
doSetDeliveryConfirmation.....	73
getDocumentWDTFFileUrl.....	74
ROUTES	75
getMyRoutes.....	75

LOGISTICS	76
getMyPackages.....	76
PAYMENTS	78
getMyPayments.....	78
RETURNS	80
getMyReturns.....	80
doReturnCreatePackage.....	81
doDeleteReturn.....	82
doEditReturn.....	83
doCreateReturn.....	84
getReturnQuantityAvailable.....	85
getReturnNowProductInfo.....	86
doReturnClearPackage.....	87
doReturnGetPackageLabel.....	88
doReturnSendPackage.....	89
doReturnNow.....	90
DROPSHIPPING	92

CHANGELOG

1.23 (16.12.2024)	<ul style="list-style-type: none"> • In method <code>getMyOrders</code> added response parameters: <code>description</code> (#69792) • In method <code>doOrderItemEdit</code> added response parameters: <code>finalDocumentType</code>, <code>description</code> (#69792) • added support for method names in url e.g. http://adres.webservice.fl5.eu:5555/ws/getProductsInfo (#69897) • In method <code>getMyBonuses</code> added response parameters: <code>targetLevelPeriodList</code> (#50881) • New method <code>getOrderStatus</code> (#69419, #67770) • In method <code>doSetDeliveryConfirmation</code> added request parameters <code>clientIP</code> , <code>clientLogin</code> (#68616, #68409) • In method <code>getMyInvoices</code> added request parameters <code>getEWDtIdList</code>(#53762) . Added response parameters: <code>tmsDeliveryConfirmed</code> (#66904), <code>isEwdtBlocked</code>(#64695), <code>finalDocumentInfo</code> (#56619), <code>eWDTidList</code> (#56619), <code>srcDocumentId</code> (#36196), <code>srcDocumentNumber</code> (#55987), <code>externalItemId</code> (#55396) • In method <code>getEDocument</code> added new type of error <code>ERR_DELIVERY_NOT_CONFIRMED</code> (#64295) • In method <code>getMyCustomerInfo</code> added response parameters <code>paymentDocsStat</code>, <code>isWDTActive</code>(#64701), <code>minOrderValue</code> (#61374), <code>accountManager</code> (#53521) • In method <code>getMyReturns</code> added additional user description (#63384) • In method <code>doReturnNow</code> added request parameters <code>collectivePackageId</code> (#47080). Added response parameters <code>itemDepositInfo</code> (#51650) • New method <code>getDocumentWDTFileUrl</code> • In method <code>getProductsInfo</code> added request parameters <code>showBarcodes</code>, <code>showDeliveryDetails</code>. Added response parameters <code>barCodeList</code> (#54353), <code>estArrivalTime</code>, <code>estDeliveryTime</code>, <code>deliveryDetailsList</code> (#61971) • In method <code>getMyRoutes</code> added response parameters <code>transportType</code> (#64293)
1.22 (01.10.2024)	<ul style="list-style-type: none"> • XML support is now deprecated and will be removed in the near future. We recommend using JSON instead, as it is much faster and easier to work with. The end of support date will be established soon, and after that, only JSON will be supported.
1.21 (25.04.2023)	<ul style="list-style-type: none"> • In method <code>getProductsInfo</code> added request parameters: <code>gtuCode</code> (#34527) • In method <code>doOrderProducts</code> removed request parameters: <code>routeInfo</code> (#38317) • In method <code>getMyReturns</code> added response parameters: <code>packageBarcode</code> (#40153) • In method <code>getMyBonuses</code> added response parameters: <code>type_bonus</code>, <code>is_ranking</code> and <code>rankingPromoList</code> (#48802) • In method <code>getReturnNowProductInfo</code> added response parameters: <code>returnSectorInfo</code> (#47453) • In method <code>doLogmCreatePackage</code> added request parameters: <code>x</code>, <code>y</code>, <code>z</code>, <code>weight</code> and response parameters: <code>courierLabelList</code> (#44972) • In method <code>doLogmCreatePackage</code> added request parameters: <code>collectivePackageInfo</code> (#47020) and response parameters: <code>dropshippingLabelList</code> (#45984) • New method <code>getTree</code> (#33833)
1.20 (15.09.2021)	<ul style="list-style-type: none"> • In method <code>getMyOrders</code> added response parameters: <code>priceCatExclTax</code>, <code>priceCatInclTax</code>. In method <code>doOrderItemEdit</code> added response parameters: <code>priceCatExclTax</code>, <code>priceCatInclTax</code>. In method <code>getMyInvoices</code> added response parameters: <code>priceCatExclTax</code>, <code>priceCatInclTax</code> (#10665) • In method <code>getMyPackages</code> added request parameters: <code>getTrackingUrl</code> (#10553) • In method <code>getMyCustomerInfo</code> added response parameters: <code>deliveryConfirmationDocTypes</code> (#10499) • In method <code>getMyRoutes</code> response parameter <code>isDefault</code> was marked as „deprecated“

and is always returning value "false" (#10477)

- In method `getMyReturns` added response parameters: `documentNumber` (#10441)
- In method `getMyOrders` added response parameters: `isFromDeficiencies` (#10437)
- In method `getMyOrders` added response parameters: `isExternalOrderInProgress` (#10400)
- In method `doOrderProducts` added request parameters: `customerDestinationWarehouseId`, `isCustomerSupply` (#10378)
- In method `getMyInvoices` the range of values of request parameter `closeStatus` was changed (#10339)
- In method `getMyInvoices` added request parameters: `productIdList` (#10128)
- In method `doOrderProducts` added request parameters: `superSupplier` (#10000)
- In method `getProductsInfo` added response parameters: `motonet`. In method `doOrderProducts` added response parameters: `motonet` (#9906)
- In method `getMyOrders` added response parameters: `motonet` (for items). In method `getProductsInfo` added request parameters: `motonet`. In method `doOrderProducts` added request parameters: `motonet`. In method `doBuyNow` added request parameters: `motonet`. In method `getReturnNowProductInfo` added request parameters: `motonet`. In method `doReturnNow` added request parameters: `motonet` (#9896)
- In method `getMyCustomerInfo` added response parameters: `isErpSaleLocked` (#9890)
- In method `doCreateReturn` added request parameters: `srcDocumentId`. Added response errors: `ERR_WRONG_SOURCE_DOCUMENT`, `ERR_WRONG_SOURCE_DOCUMENT_WAREHOUSE`. In method `getReturnQuantityAvailable` added request parameters: `detailedQuantity`. Added response parameters: `documentList` (#9834)
- In method `getProductsInfo` removed response parameters: `deliveryTimeMax` (#9772)
- In method `doOrderClose` added response errors: `ERR_BLOCKED_CONFIRM_RECEIPT_DELIVERY` (#9764)
- In method `getMyPackages` added response parameters: `urlTracking` (#9758)
- In method `getMyCustomerInfo` added response parameters: `creditLimit` (#9730)
- In method `getMyBonuses` added request parameters: `getGroupPromoList`, `getProductPromoList`. Added response parameters: `groupPromoList`, `productPromoList` (#9704)
- Added method `getProductStockChanges` (#9582)
- In method `doReturnNow` added response parameters: `isSplitPayment` (#9570)
- In method `doOrderClose` added request parameters: `routeId`. Added response errors: `ERR_CIP_ROUTE`, `ERR_ROUTE_UNAVAILABLE`, `ERR_WRONG_ROUTE_ID` (#9464)
- In method `getMyCustomerInfo` added response parameters: `pointsExpired` (#9450)
- In method `getMyOrders` added response parameters: `routeId`. In method `getMyInvoices` added response parameters: `routeId` (#9400)
- In method `doLogmCreatePackage` added request parameters: `hasDeposit`, `depositBuyPrice`, `depositIlkod`. Added response errors: `ERR_INVALID_DEPOSIT_BUY_PRICE` (#8967)
- In method `getProductsInfo` added response parameters: `hasProductDeposit` (#8934)
- In method `doLogmCreatePackage` added response parameters: `specificationPdfData` (#8924)
- Added method `getReturnNowProductInfo` (#8874)
- In method `doOrderItemEdit` added response errors: `ERR_ITEM_HAS_DEPOSIT` (#8868)
- In method `doOrderProducts` added request parameters: `handoverToDD`. Added response errors: `ERR_INVALID_HANDOVERTODD` (#8820)

	<ul style="list-style-type: none"> • In method <code>getMyCustomerInfo</code> added response parameters: <code>isDD</code> (#8758) • In method <code>getMyCustomerInfo</code> added response parameters: <code>isInvoice</code>, <code>isReceipt</code> (#8709) • Added method <code>doLogmCreatePackage</code> (#8685) • In method <code>doOrderProducts</code> added request parameters: <code>packMergeKey</code>, <code>srcDocId</code>, <code>retShelfMerge</code> (#8408) • In method <code>getMyInvoices</code> removed request parameters: <code>webCatUserId</code>. In method <code>doOrderProducts</code> removed request parameters: <code>webCatUserId</code> (#8167) • In method <code>doOrderProducts</code> added request parameters: <code>onlyFoundItems</code> (#8082) • In method <code>getMyPackages</code> added response parameters: <code>isClosed</code> (for package) and <code>Ilkod</code> (for package item) (#7587)
<p>1.19 (05.11.2019)</p>	<ul style="list-style-type: none"> • New method <code>getProductFileUrl</code> (#7145) • In method <code>doOrderProducts</code> added parameters <code>routeInfo</code>, <code>dhlde_postfilialePostfilialNumber</code>, <code>dhlde_postfilialepostNumber</code>, <code>dhlde_postfilialeZip</code>, <code>dhlde_postfilialeCity</code>, <code>dhlde_preferredLocation</code>, <code>dhlde_preferredNeighbour</code>, <code>dhlde_recipientEmailAddress</code>, <code>dhlde_namedPersonOnly</code>, <code>dhlde_identCheckSurname</code>, <code>dhlde_identCheckGivenName</code>, <code>dhlde_identCheckDateOfBirth</code>, <code>dhlde_identCheckMinimumAge</code>, <code>dhlde_visualCheckOfAge</code>, <code>dhlde_preferredDay</code>, <code>dhlde_preferredTime</code>, <code>dhlde_noNeighbourDelivery</code>, <code>dhlde_bulkyGoods</code>, <code>dhlde_additionalInsurance</code> i <code>dhlde_noticeOfNonDeliverability</code> (#6836) • In method <code>getMyCustomerInfo</code> added return fields <code>saleBlockInfo</code>, <code>status</code>, <code>code</code> and <code>msg</code> (#6720) • In method <code>getMyOrders</code> added parameter <code>getLoyaltyInfo</code> and return fields <code>isReward</code> and <code>loyaltyRewardPoints</code>. In method <code>getMyInvoices</code> added return fields <code>loyaltyRewardPoints</code> and <code>isReward</code> (#6690) • In method <code>doOrderProducts</code> added parameter <code>externalItemId</code> (#6673) • In method <code>getMyOrders</code> and <code>getMyInvoices</code> added parameter and return field <code>isReward</code> (#6577) • In method <code>doOrderProducts</code> added parameter <code>isReward</code> (#6572) • In method <code>doOrderItemsMove</code> added return field <code>destOrderId</code> (#6511) • In method <code>getMyInvoices</code> added parameters <code>wmsStatus</code>, <code>getTrackingNrList</code> and added return fields <code>externalId</code> i <code>trackingNrList</code> (#6459) • In method <code>doCreateComplaint</code> added parameters and return fields <code>compensationValue</code> i <code>compensationCurrency</code> (#6418) • In method <code>getMyCustomerInfo</code> parameters <code>dateFrom</code>, <code>dateTo</code> are deprecated (#6151)
<p>1.18 (17.04.2019)</p>	<ul style="list-style-type: none"> • In method <code>getMyInvoices</code> added parameter <code>deliveryConfirmationStatus</code>, and added return field <code>isDeliveryConfirmed</code>. In method <code>getMyCustomerInfo</code> added return fields <code>isDeliveryConfirmationRequired</code>, <code>deliveryConfirmationPeriod</code>. New method <code>doSetDeliveryConfirmation</code> (#6148) • In method <code>doOrderProducts</code> added parameter <code>shipmentType</code> (#6024) • In method <code>doOrderProducts</code> dodano nowy rodzaj błędu <code>ERR_INVALID_CUSTOMER</code> (#5938) • In method <code>getMyInvoices</code> added return field <code>country</code> (#5925) • In method <code>getMyCustomerInfo</code> added return field <code>nextRouteDt</code> (#5813) • In method <code>getMyCustomerInfo</code> deleted a field <code>isColorGroupPromo</code> (#5804) • In method <code>getMyInvoices</code> added return fields: <code>weight_net</code>, <code>weight_gross</code>, <code>customs_code</code> (#5799) • New method <code>doSetRodoStatus</code> (#5600) • In method <code>getMyCustomerInfo</code> added return field <code>rodoStatus</code> (#5592)

	<ul style="list-style-type: none"> • In method <code>getMyInvoices</code> added return field and parameter <code>isDoReturnNow</code> (#5550) • New method <code>doReturnNow</code> (#5358) • In method <code>getProductsInfo</code> change extended functionality of field <code>isGroupPromo</code>, changed value of <code>ERR_PRODUCTLIST_OVERSIZED</code> (#5300) • In method <code>getMyCustomerInfo</code> added return field <code>isActive</code> (#5277) • In method <code>getMyBonuses</code> extended functionality of field <code>type</code> (#5233) • In method <code>doOrderProducts</code> changed error value <code>ERR_PRODUCTORDERLIST_OVERSIZED</code> (#5231) • In method <code>doCreateComplaint</code> added parameters: <code>powerValue</code>, <code>powerUnit</code>, <code>installedBy</code>, <code>uninstalledBy</code>, <code>faultFoundBy</code>, <code>faultWhen</code>, <code>refuseAddCosts</code>. (#5174)
<p>1.17 (19.09.2017)</p>	<ul style="list-style-type: none"> • In method <code>getMyCustomerInfo</code> added return field <code>isColorGroupPromo</code> (#5016) • In method <code>getMyBonuses</code> added return field <code>targetIncrease</code> (#4987) • In method <code>getMyInvoices</code> and <code>getMyOrders</code> added return field <code>justNumber</code> (#4907) • In method <code>getProductsInfo</code> added return field <code>isGroupPromo</code> (#4823) • In method <code>getMyBonuses</code> added struct <code>targetLevelList</code> (#4880) • In method <code>getMyBonuses</code> extended functionality of field <code>type</code> (#4804) • In method <code>getMyInvoices</code> added: parameter <code>getLoyaltyInfo</code> and return field <code>loyaltyPoints</code> (#4786) • In method <code>getMyCustomerInfo</code> added return struct <code>groupPromoList</code> (#4777)
<p>1.16 (04.04.2017)</p>	<ul style="list-style-type: none"> • In method <code>getMyInvoices</code> added return field <code>loyaltyPoints</code> (#4738) • In method <code>getMyCustomerInfo</code> added struct <code>loyaltyInfo</code> (#4734) • New method <code>doDocumentPrint</code> (#4687) • New method <code>doComplaintPrint</code> (#4672) • In method <code>doCreateComplaint</code> added new error code <code>ERR_CAR_DATA_INACTIVE</code> (#4669) • In method <code>getMyCustomerInfo</code> added return fields: <code>isComplaintEditCar</code>, <code>isComplaintEditDesc</code>, <code>isComplaintDescRequired</code> (#4664) • In method <code>getMyReturns</code> added return field <code>canEdit</code> (#4655) • In method <code>getMyCustomerInfo</code> added return field <code>isReturnActive</code> (#4646) • New method <code>getReplacements</code> (#4645)
<p>1.15 (14.02.2017)</p>	<ul style="list-style-type: none"> • In method <code>getProductsInfo</code> always return fields: <code>quantityRealShow</code>, <code>quantityReal</code> (#4620) • In method <code>getInvoicesComplaintProduct</code> added return field <code>quantityAvailable</code> (#4615) • In method <code>getMyCustomerInfo</code> added return field <code>isComplaintActive</code> (#4613) • In method <code>getMyOrders</code> added parameters: <code>additionalInfo</code>, <code>dateAddInfo</code> (#4581) • In method <code>doOrderEdit</code> added parameters: <code>additionalInfo</code>, <code>dateDelivery</code> and new error code <code>ERR_ADDITIONAL_INFO_INACTIVE</code> (#4580) • In method <code>getProductsInfo</code> added return fields: <code>reference2</code>, <code>tecidd</code>, <code>tecnum</code>, <code>group2</code>, <code>isReplacement</code>, <code>isPhoto</code>, <code>producer</code> (#4569) • In method <code>getVersion</code> added return fields: <code>ownum</code>, <code>verFull</code> (#4548) • In method <code>getMyCustomerInfo</code> added return fields: <code>isGoodsIssueActive</code>, <code>isExternalStockAvailable</code>, <code>stockQuantityFormat</code>, <code>isEdocumentActive</code>. In method <code>getMyInvoices</code> added return field <code>isPrinted</code> (#4540) • In method <code>getMyCustomerInfo</code> added return field <code>warehouseList</code> (#4529) • New method <code>doBuyNow</code> (#4525) • In method <code>getMyInvoices</code> added return fields: <code>recipientId</code>, <code>recipientName</code>, <code>recipientShortName</code> (#4521)
<p>1.14</p>	<ul style="list-style-type: none"> • New method <code>getMyBonuses</code> (#4504)

<p>(21.12.2016)</p>	<ul style="list-style-type: none"> • In method <code>getMyComplaints</code> added return field <code>decisionComment</code> (#4469) • In method <code>getMyCustomerInfo</code> added return fields: <code>isPromoActive</code>, <code>subTerminalId</code>, <code>showPaymentsOnStart</code>, <code>isBonusesActive</code>, <code>introText</code> (#4464) • New method <code>doReturnSendPackage</code> (#4451) • New method <code>doReturnGetPackageLabel</code> and in method <code>doReturnCreatePackage</code> added new error codes: <code>ERR_INVALID_RETURN_ID</code>, <code>ERR_INVALID_RETURN_STATE</code>, <code>ERR_INVALID_RETURN_STATUS</code>, <code>ERR_INTERNAL</code> (#4441) • In method <code>getMyInvoices</code> added return fields: <code>type</code>, <code>eDocumentStatus</code> (#4435) • In method <code>getMyComplaints</code> added return fields: <code>storeExt</code>, <code>currency</code>, <code>productionYear</code>, <code>brandVehicle</code>, <code>registerNumber</code>, <code>engineNumber</code>, <code>vinNumber</code>, <code>engineCapacity</code>, <code>installDate</code>, <code>counterStateInstall</code>, <code>workshopInstallName</code>, <code>workshopInstallAddress</code>, <code>uninstallDate</code>, <code>counterStateUninstall</code>, <code>workshopUninstallName</code>, <code>workshopUninstallAddress</code> (#4417) • New method <code>doCreateReturn</code> (#4415) • In method <code>getMyReturns</code> in struct <code>filterOptions</code> added fields: <code>dateFrom</code>, <code>dateTo</code> and new error codes: <code>ERR_INVALID_DATERANGE</code>, <code>ERR_INVALID_RETURN_ID</code>, <code>ERR_INVALID_RETURN_STATE</code>, <code>ERR_INVALID_RETURN_STATUS</code> (#4413) • New method <code>doReturnClearPackage</code> (#4409) • New method <code>doEditReturne</code> (#4408) • New method <code>getReturnQuantityAvailable</code> (#4407) • New method <code>doDeleteReturn</code> (#4394) • New method <code>doReturnCreatePackage</code> (#4370) • In method <code>doCreateComplaint</code> added parameters: <code>productionYear</code>, <code>brandVehicle</code> (#4365) • New method <code>getMyReturns</code> (#4354) • In method <code>getInvoicesComplaintProduct</code> added struct <code>itemList</code> (#4306) • New method <code>getInvoicesComplaintProduct</code> (#4280) • In method <code>doCreateComplaint</code> added parameters: <code>registerNumber</code>, <code>engineNumber</code>, <code>vinNumber</code>, <code>engineCapacity</code>, <code>installDate</code>, <code>counterStateInstall</code>, <code>workshopInstallName</code>, <code>workshopInstallAddress</code>, <code>uninstallDate</code>, <code>counterStateUninstall</code>, <code>workshopUninstallName</code>, <code>workshopUninstallAddress</code> (#4238) • New method <code>getComplaintQuantityAvailable</code> (#4232) • In method <code>getMyComplaints</code> added return fields: <code>dateClose</code>, <code>productionYear</code>, <code>brandVehicle</code> (#4218)
<p>1.13 (22.07.2016)</p>	<ul style="list-style-type: none"> • In method <code>getMyComplaints</code> added return fields <code>dateCreate</code>, <code>dateClose</code>, <code>decision</code>, <code>decisionText</code>, <code>decisionType</code> (#4032) • In method <code>getMyInvoices</code> added struct <code>pieceList</code> (#4001) • In method <code>doOrderProducts</code> added struct <code>pickupPointAddress</code> (#3921)
<p>1.12 (10.03.2016)</p>	<ul style="list-style-type: none"> • In method <code>doOrderProducts</code> added parameter <code>sendOnSaturday</code> (#3913) • In method <code>doOrderProducts</code> added parameter <code>deliveryOnSaturday</code> and new error code <code>ERR_COURIER_INFO_EDIT_INACTIVE</code> (#3848) • In method <code>getMyCustomerInfo</code> added struct <code>balance</code> (#3833)
<p>1.11 (22.12.2015)</p>	<ul style="list-style-type: none"> • In method <code>getMyPayments</code> added parameters <code>isRelDocument</code>, <code>relDocumentId</code>, <code>relDocumentType</code>, <code>payItemList</code> (#3758) • In method <code>getMyInvoices</code> and <code>getMyOrders</code> added an option to filter the field <code>externalId</code> (#3731) • New method <code>getMyPackages</code> (#3728) • In method <code>doOrderProducts</code> added struct <code>additionalDeliveryAddress</code> containing the parameters: <code>name</code>, <code>company</code>, <code>street</code>, <code>postcode</code>, <code>city</code>, <code>country</code>,

	<p>phone, email; and new error code ERR_ADDITIONAL_DELIVERY_ADDRESS_EMPTY_NAME_AND_COMPANY (#3683)</p> <ul style="list-style-type: none"> • In method <code>getMyInvoices</code> added parameter <code>srcDocId</code> (#3665) • In method <code>doLogin</code> added new error code <code>ERR_LOGIN_BLOCKED</code> (#3663)
<p>1.10 (14.09.2015)</p>	<ul style="list-style-type: none"> • New method <code>getMyComplaints</code> (#3571) • New method <code>doCreateComplaint</code> (#3566) • In method <code>getMyInvoices</code> added parameters <code>closeStatus</code> and <code>isClosed</code> (#3560) • In method <code>doOrderProducts</code> added parameters <code>packageProductId</code>, <code>dataExpire</code> and new error codes (#3544): <ul style="list-style-type: none"> ◦ <code>ERR_PACKAGE_PRODUCT_INACTIVE</code> ◦ <code>ERR_PACKAGE_PRODUCT_ID_INVALID</code>
<p>1.09 (17.08.2015)</p>	<ul style="list-style-type: none"> • English version of documentation. • In method <code>doOrderProducts</code> added parameter <code>routeId</code> (#3483)
<p>1.08 (07.07.2015)</p>	<ul style="list-style-type: none"> • In method <code>doOrderProducts</code> added parameter <code>payment</code> and new error codes (#3415 and #3442): <ul style="list-style-type: none"> ◦ <code>ERR_PAYMENT_EDIT_NOT_AVAILABLE</code> ◦ <code>ERR_PAYMENT_INVALID_TYPE</code> ◦ <code>ERR_PAYMENT_TYPE_NOT_AVAILABLE</code>
<p>1.07 (30.06.2015)</p>	<ul style="list-style-type: none"> • In method <code>getMyInvoices</code> added filter <code>paymentStatus</code> (#3402) • New method <code>getMyCustomerInfo</code> (#3402) • In method <code>doOrderProducts</code> added parameters: <code>incotermsCode</code>, <code>additionalInfo</code> and new error codes (#3374): <ul style="list-style-type: none"> ◦ <code>ERR_INCOTERMS_INACTIVE</code> ◦ <code>ERR_INCOTERMS_INVALID_CODE</code> ◦ <code>ERR_ADDITIONAL_INFO_INACTIVE</code>
<p>1.06 (11.06.2015)</p>	<ul style="list-style-type: none"> • In method <code>doSearchProducts</code> added <code>treeid</code> <code>search</code> (#3364) • In method <code>getProductsInfo</code> added return fields: <code>groupSymbol</code>, <code>groupName</code>, <code>expectedDeliveryDate</code> (#3364) • In method <code>getMyInvoices</code> added return struct <code>correctedItem</code> (possibility to get sales invoice corrections) (#3364) • In method <code>doOrderProducts</code> added struct <code>deliveryAddress</code> and new error codes (#3284): <ul style="list-style-type: none"> ◦ <code>ERR_DELIVERY_ADDRESS_INACTIVE</code> ◦ <code>ERR_DELIVERY_ADDRESS_EMPTY_NAME_AND_COMPANY</code> • In method <code>getMyInvoices</code> added return field <code>wmsStatus</code> (#3283) • In methods <code>getProductsInfo</code>, <code>doOrderProducts</code> added parameter <code>ILkod</code> (#3276) • In method <code>doOrderItemsMove</code> added new error code: <code>ERR_WRONG_DEST_ORDER_ID</code> (wrong target order id) (#3252) • In methods <code>doOrderProducts</code> and <code>doOrderItemEdit</code> added parameter <code>description</code> (#3203)
<p>1.05 (27.02.2015)</p>	<ul style="list-style-type: none"> • In method <code>doLogin</code> added a limit of opened session for a single user (#3126) • In method <code>getInvoices</code> added new error code: <code>ERR_NOT_ALLOWED_FINALDOCUMENTS_TYPE</code> (#3069) • Added new error codes in methods used for managing reservations (#3066): <ul style="list-style-type: none"> ◦ <code>ERR_SALE_BLOCK</code> (Order cannot be realized. Sale blocked.) ◦ <code>ERR_ISSUE_BLOCK</code> (Order cannot be realized. Issuing vouchers blocked.) ◦ <code>ERR_ISSUES_OVERDUE</code> (Order cannot be realized. Issuing vouchers term exceeded) ◦ <code>ERR_PAYMENT_OVERDUE</code> (Order cannot be realized. Payment term exceeded) ◦ <code>ERR_CREDIT_LIMIT</code> (Order cannot be realized. Granted credit limit exceeded)
<p>1.04 (29.12.2014)</p>	<ul style="list-style-type: none"> • Added method <code>doOrderItemsMove</code> for changing order items positions (#2957) • In method <code>getProductsInfo</code> added return struct <code>subProductList</code> that contains list of subproducts if a new parameter <code>showSubProductsInfo = 1</code> is passed (#2931)

	<ul style="list-style-type: none"> • In method <code>getMyInvoices</code> added return field <code>isEDocument</code> that contains information about availability of digital version of document (#2929) • Added method <code>getEDocument</code>, that returns digital version of document (#2928) • In method <code>getMyInvoices</code> added parameter <code>finalDocuments</code> that allow to define type of returned documents (#2908, #2926) • In method <code>getProductsInfo</code> added return struct <code>photoList</code> that contains information about product photos if a new parameter <code>showPhotos = 1</code> is passed (#2850)
1.03 <i>(18.08.2014)</i>	<ul style="list-style-type: none"> • In method <code>getProductsInfo</code> added return fields: <code>priceBeforePromotionExclTax</code>, <code>priceBeforePromotionInclTax</code> • In method <code>getProductsInfo</code> added return struct <code>additionalInfo</code> with field <code>sale</code>; additional information of product is returned if a new parameter <code>showAdditionalInfo = 1</code> is passed • Added method <code>getMyRoutes</code> that returns list of available routes for logged user
1.02 <i>(12.08.2014)</i>	<ul style="list-style-type: none"> • “Change Log” changed to “Changelog” • Added dates to changelog • In method <code>doOrderProducts</code> added parameter <code>externalID</code> in struct <code>newOrderInfo</code>
1.01	<ul style="list-style-type: none"> • Added changelog • Added pages numeration • Added links in table of content
1.00	<ul style="list-style-type: none"> • First version of documentation

RULES OF WORKING WITH SESSIONS

Using the methods provided by Webservice involves going through an authentication process. Authentication is performed by calling the doLogin method, which requires the username and MD5-encoded password received from the service provider. The doLogin method returns a session key that we add to each query.

```
REQUEST-STR:
POST / HTTP/1.1
Host: 127.0.0.1:7888
Content-Type: application/json
Accept: application/json
Content-Length: 83

{"doLogin":{"userLogin":"test","userPassword":"289dff07669d7a23de0ef88d2f7129e7"}}

RESPONSE-STR:
HTTP/1.0 200 OK
Date: Tue, 17 Dec 2024 11:05:07 +0100
Server: dbfd.webServer
Connection: close
Content-Length: 76
Cache-Control: private
Content-Type: application/json; charset=utf-8

{"doLoginResponse":{"sessionId":"eP3cFozcI3pnyq9w03Fa7vwg0H7CthI0029736_D"}}

REQUEST-STR:
POST / HTTP/1.1
Host: 127.0.0.1:7888
Content-Type: application/json
Accept: application/json
Content-Length: 144

{"getProductsInfo":
{"sessionId":"eP3cFozcI3pnyq9w03Fa7vwg0H7CthI0029736_D","productList":{"product":
[{"id":"0006VI"}, {"reference":"144 666"}]}}
```

The session is closed after a specified period of inactivity. Passing an inactive session identifier to other methods will result in the error code ERR_SESSION being returned. In such a situation, you must log in again using the doLogin method to obtain a new session key. The webservice does not require logging out after each request. Limitations to obtaining a new session may include exceeding the maximum number of sessions (default 30).

Session life cycle:

- 10 minutes of activity
- 3 hours from logging in regardless of whether the session is still in use.

COMMUNICATION SCHEMA

Request:

```
{
  "getVersion": {
    "parameters"
  }
}
```

Response:

```
HTTP/1.0 200 OK
(...)
{
  "getVersionResponse": {
    "response"
  }
}
```

Response in case of error:

```
HTTP/1.0 500 Internal Server Error
(...)
{
  "error" : {
    "code": "error_code",
    "msg": "error_message"
  }
}
```

COMMUNICATION EXAMPLE

Request:

```
POST /ws HTTP/1.1
Host: 127.0.0.1
Content-Type: text/xml; charset=utf-8
Content-Length: 81
User-Agent: FL
SOAPAction: "getVersion"

{ "getVersion":{} }
```

Response:

```
HTTP/1.0 200 OK
Date: Thu, 14 Feb 2013 14:15:03 GMT
Server: dbfd.webServer
Connection: close
Content-Length: 85
Cache-Control: private
Content-Type: text/json; charset=utf-8

{
  "getVersionResponse": {
    "ownum": "526",
    "verFull": "5.645.15127 2024-09-27 LC",
    "version": "dbfd2 5.645.15127 2024-09-27 LC ENC:SXE JSON"
  }
}
```

Sample PHP code:

```
<?php

$first_call=true;
function dbfd_call( $host, $method, $params )
{
    global $first_call;

    if( !$first_call )
        echo "=====\n\n";
    $first_call = false;

    $request = Array($method=>$params);
    $request_str = json_encode($request);

    $ch = curl_init();

    curl_setopt($ch, CURLOPT_URL, $host);
    curl_setopt($ch, CURLOPT_POST, 1);
    curl_setopt($ch, CURLOPT_POSTFIELDS, $request_str);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_HEADER, true );
    curl_setopt($ch, CURLINFO_HEADER_OUT, true);
    curl_setopt($ch, CURLOPT_HTTPHEADER, Array('Content-Type: application/json', 'Accept:
application/json'));

    $response_str = curl_exec($ch);
    $error = curl_error($ch);
    $request_headers = curl_getinfo($ch,CURLINFO_HEADER_OUT);
    curl_close($ch);

    $response_header = "";
    $response_body = "";
    $response = "";

    if( !empty($response_str) )
```

```

    {
        list($response_header, $response_body) = explode("\r\n\r\n", $response_str, 2);
        $response_header .= "\r\n\r\n";
        $response = json_decode($response_body,true);
    }

    echo "REQUEST-ARRAY:\n".print_r($request,true)."\n\n";
    echo "-----\nREQUEST-STR:\n".$request_headers.$request_str."\n\n";
    echo "-----\nERROR:\n".$error."\n\n";
    echo "-----\nRESPONSE-STR:\n".$response_header.$response_body."\n\n";
    echo "-----\nRESPONSE-ARRAY:\n".print_r($response,true)."\n\n";

    return $response;
}

// =====

$host = "http://127.0.0.1:7888/";

$method = 'doLogin';
$params = Array('userLogin' => 'test',
                'userPassword' => '289dff07669d7a23de0ef88d2f7129e7' //md5 z hasła
                );

//logowanie z pobraniem sesji
$result = dbfd_call( $host, $method, $params );
if( isset($result['doLoginResponse']['sessionId']) )
{
    $sessionId = $result['doLoginResponse']['sessionId'];

    //metoda pobierająca dane o towarach
    $method = 'getProductsInfo';

    //pytamy o 2 towary: 1 - identyfikowany wg id FL5, 2 - identyfikowany wg symbolu
    $productList = Array( 'product' => [ Array('id' => '0006VI'),
                                        Array('reference' => '144 666') ] );

    $params = Array( 'sessionId' => $sessionId, 'productList' => $productList);
    dbfd_call( $host, $method, $params );
}

```

Effect of the example code in PHP:

```

REQUEST-ARRAY:
Array
(
    [doLogin] => Array
        (
            [userLogin] => test
            [userPassword] => 289dff07669d7a23de0ef88d2f7129e7
        )
)

-----
REQUEST-STR:
POST / HTTP/1.1
Host: 127.0.0.1:7888
Content-Type: application/json
Accept: application/json
Content-Length: 83

{"doLogin":{"userLogin":"test","userPassword":"289dff07669d7a23de0ef88d2f7129e7"}}

-----
ERROR:
-----
RESPONSE-STR:
HTTP/1.0 200 OK
Date: Tue, 17 Dec 2024 11:05:07 +0100
Server: dbfd.WebServer
Connection: close

```

```

Content-Length: 76
Cache-Control: private
Content-Type: application/json; charset=utf-8

{"doLoginResponse":{"sessionId":"eP3cFozcI3pnyq9w03Fa7vWg0H7CthI0029736_D"}}

-----
RESPONSE-ARRAY:
Array
(
    [doLoginResponse] => Array
        (
            [sessionId] => eP3cFozcI3pnyq9w03Fa7vWg0H7CthI0029736_D
        )
)

=====

REQUEST-ARRAY:
Array
(
    [getProductsInfo] => Array
        (
            [sessionId] => eP3cFozcI3pnyq9w03Fa7vWg0H7CthI0029736_D
            [productList] => Array
                (
                    [product] => Array
                        (
                            [0] => Array
                                (
                                    [id] => 0006VI
                                )
                            [1] => Array
                                (
                                    [reference] => 144 666
                                )
                        )
                )
        )
)

-----
REQUEST-STR:
POST / HTTP/1.1
Host: 127.0.0.1:7888
Content-Type: application/json
Accept: application/json
Content-Length: 144

{"getProductsInfo":{"sessionId":"eP3cFozcI3pnyq9w03Fa7vWg0H7CthI0029736_D","productList":
{"product":[{"id":"0006VI"},{"reference":"144 666"}]}}

-----
ERROR:

-----
RESPONSE-STR:
HTTP/1.0 200 OK
Date: Tue, 17 Dec 2024 11:05:07 +0100
Server: dbfd.WebServer
Connection: close
Content-Length: 1670
Cache-Control: private
Content-Type: application/json; charset=utf-8

{"getProductsInfoResponse":{"productInfoList":{"productInfo":
[{"ILkod":"","blocked":"","currency":"z\u0142","decimals":"2","expectedDeliveryDate":"","file

```

```
s": "0", "group2": "BUM", "groupName": "TOYOTA", "groupSymbol": "BAZA", "gtuCode": "", "hasDeposit": "0",
"hasProductDeposit": "0", "id": "0006VI", "isGroupPromo": "0", "isPhoto": "1", "isReplacement": "1", "mo
tonet": "", "name": "SUPER CZESC DO TOYOTY
9876", "priceCatExclTax": "55.56", "priceCatInclTax": "68.34", "reference": "96714", "reference2": "LA
LALALALALA", "stockInfoList": {"stockInfo": [{"externalStockInfo":
{"deliveryTimeMin": "24", "deliveryTimeShow": "1d", "quantity": "75.00", "quantityShow": "75.00", "war
ehouseShow": ""}, {"inventory": "0", "priceExclTax": "344.30", "priceInclTax": "423.49", "quantity": "19
1.00", "quantityMultiplier": "0", "quantityReal": "181.00", "quantityRealShow": "181.00", "quantitySh
ow": "191.00", "viewOnly": "0", "warehouseId": "01"}]}, {"tax": "23", "tecidd": "256", "tecnum": "0P
525/1", "type": "0", "unit": "SZT"}},
{"ILkod": "", "blocked": "0", "currency": "z\u00142", "decimals": "0", "expectedDeliveryDate": "", "files
": "0", "groupName": "", "groupSymbol": "", "gtuCode": "", "hasDeposit": "0", "hasProductDeposit": "0", "i
d": "0006YW", "isGroupPromo": "0", "isPhoto": "0", "isReplacement": "1", "motonet": "", "name": "KIERUNKO
WSKAZ AUDI 80 91- /L/TYC", "priceCatExclTax": "0.00", "priceCatInclTax": "0.00", "reference": "144
666", "stockInfoList": {"stockInfo": [{"externalStockInfo":
{"quantity": "0", "quantityShow": "", "warehouseShow": ""}, {"inventory": "0", "priceExclTax": "0.00", "p
riceInclTax": "0.00", "quantity": "0", "quantityMultiplier": "0", "quantityReal": "0", "quantityRealSh
ow": "", "quantityShow": "", "viewOnly": "0", "warehouseId": "01"}]}, {"tax": "23", "type": "0", "unit": ""}
]}}}
```

RESPONSE-ARRAY:

```
Array
(
    [getProductsInfoResponse] => Array
        (
            [productInfoList] => Array
                (
                    [productInfo] => Array
                        (
                            [0] => Array
                                (
                                    [ILkod] =>
                                    [blocked] => 0
                                    [currency] => z\u0142
                                    [decimals] => 2
                                    [expectedDeliveryDate] =>
                                    [files] => 0
                                    [group2] => BUM
                                    [groupName] => TOYOTA
                                    [groupSymbol] => BAZA
                                    [gtuCode] =>
                                    [hasDeposit] => 0
                                    [hasProductDeposit] => 0
                                    [id] => 0006VI
                                    [isGroupPromo] => 0
                                    [isPhoto] => 1
                                    [isReplacement] => 1
                                    [motonet] =>
                                    [name] => SUPER CZESC DO TOYOTY 9876
                                    [priceCatExclTax] => 55.56
                                    [priceCatInclTax] => 68.34
                                    [reference] => 96714
                                    [reference2] => LALALALALALA
                                    [stockInfoList] => Array
                                        (
                                            [stockInfo] => Array
                                                (
                                                    [0] => Array
                                                        (
                                                            [externalStockInfo] => Array
                                                                (
                                                                    [deliveryTimeMin] => 24
                                                                    [deliveryTimeShow] => 1d
                                                                    [quantity] => 75.00
                                                                    [quantityShow] => 75.00
                                                                    [warehouseShow] =>
                                                                )
                                                            [inventory] => 0
                                                            [priceExclTax] => 344.30
                                                            [priceInclTax] => 423.49
                                                            [quantity] => 191.00
                                                            [quantityMultiplier] => 0
                                                            [quantityReal] => 181.00
                                                        )
                                                    )
                                                )
                                        )
                                    )
                                )
                            )
                        )
                    )
                )
            )
        )
    )
```


ERROR CODES

ERR_UNKNOWN

Unknown error

ERR_INTERNAL

Internal server error

ERR_KEY_WRONG

Can't read message (decoding message failure) / Wrong transmission key

ERR_REQUEST

Wrong request syntax

ERR_METHOD_UNAVAILABLE

Method unavailable (user-type connects from wrong address / method does not exist)

ERR_LOGIN

Wrong login or password

ERR_SESSION

Session expired or does not exist

LEGEND

- (?) Optional parameter
- * Details in comments

AVAILABLE METHODS

GENERAL

getVersion

Get system version number

getMyCustomerInfo

Get information about logged user

doLogin

Login user to system

getMyBonuses

Get information about promotions / bonuses

doSetRodoStatus

Toggle agreement of the processing personal data

getTree

Get products categories tree

PRODUCTS

getProductsInfo

Get information about products and services

getProductFileUrl

Put file to download and return an URL adress

getProductStockChanges

Get paged information about product stock changes

doSearchProducts

Search products

getReplacements

Get product replacements

doBuyNow

Purchase method Buy Now by goods issue note.

COMPLATINS

doCreateComplaint

Make a complaint

getMyComplaints

Get list of complaint of logged user

getComplaintQuantityAvailable

Get quantity can be complained for document item..

getInvoicesComplaintProduct

Get list of sales documents by the specified product and complaint date range.

doComplaintPrint

Generate PDF printout of complaint

RESERVATIONS

doOrderProducts

Make reservation for products

getMyOrders

Get order list for logged user

getOrderStockInfo

Get information about availability of order items

doOrderClose

Close order

doOrderItemDelete

Delete order item

doOrderItemEdit

Edit order item

doOrderEdit

Edit order parameters

doOrderDelete

Delete order

doOrderItemsMove

Move items between orders

getOrderStatus

Get order status

DOCUMENTS

getMyInvoices

Get list of sale documents of logged user

getEDocument

Get digital version of document (PDF)

doDocumentPrint

Generate PDF printout of document

doSetDeliveryConfirmation

Delivery confirmation for WZ document

getDocumentWDTFileUrl

Displaying the WDT printout for downloading and returning the URL address

ROUTES

getMyRoutes

Get list of users routes

LOGISTICS

getMyPackages

Get list of users packages

PAYMENTS

getMyPayments

Get list of payments

RETURNS

getMyReturns

Get list of users returns

doReturnCreatePackage

Create package number for list of returns

doDeleteReturn

Delete registered return

doEditReturn

Edit Return

doCreateReturn

Create new return

getReturnQuantityAvailable

Get available quantity product possible to return

getReturnNowProductInfo

Additional information related to product quick return

doReturnClearPackage

Delete information about package from list of returns

doReturnGetPackageLabel

Get package label for return

doReturnSendPackage

Mark return from package as sent

doReturnNow

Quick return

GENERAL

getVersion

Get system version number

Parameters:

none

Return:

1 version	(string)	Version number DBFD
2 ownum	(string)	The installation ID
3 verFull	(string)	The full version number

Error codes:

none

Example:

```
Request:
{"getVersion": {}}

Response:
{"getVersionResponse": {
  "version": "dbfd2 5.36.712 2013-02-13 ENC:SXE JSON"
}}
```

getMyCustomerInfo

Get information about logged user

Parameters:

1 sesionId (string) Session id obtained with doLogin method

Return:

1	customerId	(string)	User id
2	name	(string)	Name
3	shortName	(string)	Short name
4	date	(date)	[format: yyyy-mm-dd] Register date
5	FKId	(string)	Number fk
6	FKId2	(string)	Number fk2
7	taxId	(string)	Tax id
8	payer	(struct)	information about payer
	1 name	(string)	Name
	2 shortName	(string)	Short name
	3 taxId	(string)	Tax id
	4 address	(struct)	Address
	1 name	(string)	Name and surname
	2 companyName	(string)	Company name
	3 street	(string)	Street
	4 zipCode	(string)	Post code
	5 city	(string)	City
	6 country	(string)	Country
	7 phone	(string)	Phone
	8 email	(string)	Email
	9 fax	(string)	Fax
9	accountManager	(string)	Manager (name + surname of operator)
	1 name	(string)	Name + surname of operator
	2 mail	(string)	mail
	3 phone	(string)	phone
	4 supervisor	(struct)	supervisor
	1 name	(string)	Name + surname of operator
	2 mail	(string)	mail
	3 phone	(string)	phone
10	address	(struct)	Address
	1 name	(string)	Name and surname
	2 companyName	(string)	Company name
	3 street	(string)	Street
	4 zipCode	(string)	Post code
	5 city	(string)	City
	6 country	(string)	Country
	7 phone	(string)	Phone
	8 email	(string)	Email
	9 fax	(string)	Fax
11	mailingAddress	(struct)	Mailing address
	1 name	(string)	Name and surname
	2 companyName	(string)	Company name
	3 street	(string)	Street
	4 zipCode	(string)	Post code
	5 city	(string)	City
	6 country	(string)	Country
	7 phone	(string)	Phone
	8 email	(string)	Email
	9 fax	(string)	Fax
12	deliveryAddress	(struct)	Delivery address

1	name	(string)	Name and surname
2	companyName	(string)	Company name
3	street	(string)	Street
4	zipCode	(string)	Post code
5	city	(string)	City
6	country	(string)	Country
7	phone	(string)	Phone
8	email	(string)	Email
9	fax	(string)	Fax
13	comment	(string)	Comment
14	balance	(struct[])	Balance(lista z podziałem na waluty)
	1 item	(struct)	Balance
	1 currency	(string)	Currency
	2 currencyRate	(string)	Currency rate
	3 value	(float)	Value
	4 overdue	(float)	Overdue value
15	isPromoActive	(int)	Promotion active (1=yes, 0=no)
16	subTerminalId	(string)	Subterminal id
17	showPaymentsOnStart	(int)	Balances at the start (1=yes, 0=no)
18	isBonusesActive	(int)	Bonuses active (1=yes, 0=no)
19	isComplaintActive	(int)	Complaints active (1=yes, 0=no)
20	isComplaintEditCar	(int)	Complaints – edition of vehicle data (1- yes, 0-no)
21	isComplaintEditDesc	(int)	Complaints – edition of defect description (1=yes, 0-no)
22	isComplaintDescRequired	(int)	Complaints - is description defect required (1=yes, 0-no)
23	isReturnActive	(int)	Active returns (1=yes, 0=no)
24	isGoodsIssueActive	(int)	Preview goods Issue note (1=yes, 0=no)
25	isExternalStockAvailable	(int)	External stock available
26	stockQuantityFormat	(int)	Stock quantity format (1 - none, 2 - '*' is anything in stock , 3 - 0-none, 1-one, 2-more, 4 – actual state, 7 – to 10 state, later >10, 8 - to 5 state, later > 5)
27	isEdocumentActive	(int)	Electronic version of invoices active (1=yes, 0=no)
28	rodoStatus	(int)	Is consent to the processing of personal data (1=yes, 0=no)
29	nextRouteDt	(string)	[yyyy-mm-dd hh:ii:ss] The closest time of departure
30	loyaltyInfo	(struct)	Information about loyalty program
	1 isActive	(int)	Client active in loyalty system (1=yes, 0=no)
	2 points	(int)	Available points
	3 pointsExpired	(int)	Expired points
31	introText	(CDATA)	Welcome text
32	warehouseList	(struct[])	List of available warehouses
	1 warehouse	(struct)	Information about warehouse
	1 id	(string)	Warehouse id
	2 name	(string)	Warehouse name
	3 isDefault	(int)	Default warehouse (1=yes, 0=no)
	4 canOrder	(int)	Can order in this warehouse (1=yes, 0=no)
	5 canViewStock	(int)	Can check state stock in this warehouse (1=yes, 0=no)
	2 warehouse ...		
33	groupPromoList	(struct[])	List of promoted product groups
	1 groupPromo	(struct)	Promoted product group
	1 groupSymbol	(string)	Product group Id
	2 groupName	(string)	Product group name
	3 (*) dateFrom	(date)	[yyyy-mm-dd] Date of promotion start
	4 (*) dateTo	(date)	[yyyy-mm-dd] Date of promotion end
	2 groupPromo ...		
34	isDeliveryConfirmationRequired	(int)	Confirmation of receipt of delivery (1 - yes 0 - no)
35	deliveryConfirmationPeriod	(int)	Number of days to confirm receipt of delivery

36	deliveryConfirmationDocTypes	(string)	Delivery confirmation – document types
37	saleBlockInfo	(struct[])	Information about lock or warning
	1 status	(int)	Sale lock (0 – no, 1 – yes, 2 – no, but with statement)
	2 isErpSaleBlocked	(int)	Sale lock in cairo.ERP (0 – no, 1 – yes)
	3 code	(string)	Error id
	4 msg	(string)	Error description
38	isInvoice	(int)	Can customer use invoices (0 – no, 1 – yes)
39	isReceipt	(int)	Can customer use receipts (0 – no, 1 – yes)
40	isDD	(int)	Can customer use goods issue notes (0 – no, 1 – yes)
41	creditLimit	(struct)	Credit limit
	1 value	(float)	Value of the credit limit
	2 valueLeft	(float)	Value of the credit limit left
	3 currency	(float)	Currency of the credit limit
42	paymentDocsStat	(struct)	Payment Statistics
	1 overdueCount	(int)	number of overdue payments
43	isWDTActive	(int)	whether WDT is active (1 - yes 0 - no)
44	minOrderValue	(float)	minimum net order value

Error codes:

ERR_METHOD_UNAVAILABLE

Method unavailable (user-type connects from wrong address / method does not exist)

Comments:

* deprecated (saved for compatibility with old versions)

Example:

```
Request:
{
  "getMyCustomerInfo": {
    "sessionId": "2BZY8zSx0D00UmM0y1FKSZjMsgDG564YxTMvj_u_D"
  }
}

Response:
{
  "getMyCustomerInfoResponse": {
    "name": "KOWAL Jan Kowalski",
    "shortName": "Jan Kowalski",
    "date": "2011-02-21",
    "fkId": "123456",
    "fkId2": "234567",
    "taxId": "12112112",
    "payerInfo": "nip nazwa adres",
    "accountManager": "Jan Operator",
    "comment": "dopisany komentarz",
    "address": {
      "name": "Jan Kowalski",
      "companyName": "KOWAL Jan Kowalski",
      "street": "Uliczna",
      "zipCode": "12-345",
      "city": "Miasto",
      "country": "Polska",
      "phone": "123456789",
    }
  }
}
```

```
"email": "jk@mail.pl",
"fax": "123456789"
},
"mailingAddress": {
  "name": "Jan Kowalski",
  "companyName": "KOWAL Jan Kowalski",
  "street": "Uliczna",
  "zipCode": "12-345",
  "city": "Miasto",
  "country": "Polska",
  "phone": "123456789",
  "email": "jk@mail.pl",
  "fax": "123456789"
},
"deliveryAddress": {
  "name": "Jan Kowalski",
  "companyName": "KOWAL Jan Kowalski",
  "street": "Uliczna",
  "zipCode": "12-345",
  "city": "Miasto",
  "country": "Polska",
  "phone": "123456789",
  "email": "jk@mail.pl",
  "fax": "123456789"
},
"payer": {
  "name": "interland platnik",
  "shortName": "platnik",
  "taxId": "12112112",
  "address": {
    "name": "Mama Jana Kowalskiego",
    "companyName": "mamusia",
    "street": "Uliczna",
    "zipCode": "12-345",
    "city": "Miasto",
    "country": "Polska",
    "phone": "123456789",
    "email": "mama@mail.pl",
    "fax": "123456789"
  }
}
}
}
```

doLogin

Login user to system

Parameters:

1	userLogin	(string)	User login
2	userPassword	(string)	Md5 of user password
3(?)	languageId	(string)	Language id (default: user language)

Return:

1.	sessionId	(string)	Session id
----	-----------	----------	------------

Erroe codes:

ERR_WRONG_LANGUAGE	Wrong user language
ERR_USERTYPE_DENIED	Access for specified user-type denied
ERR_TOO_MANY_SESSIONS	Too many opened sessions. Use session id from previous login or wait until one of sessions expire
ERR_LOGIN_BLOCKED	Access blocked. You have exceeded the maximum time without purchasing. Please contact your provider.

Comments:

Session is closed after specified time of inactivity. Sending an inactive session id to others methods will result in error code `ERR_SESSION`. In that case it is needed to log in again using `doLogin` method and get a new session key. Number of sessions for a single user is limited to 10. If this limit is exceeded, this method will return error code `ERR_TOO_MANY_SESSIONS`.

Example:

```
Request:
{"doLogin": {
  "userLogin": "CAIRO",
  "userPassword": "86484e9f2ff98544f98b5fe89c9c6"
}}

Response:
{"doLoginResponse": {
  "sessionId": "Gq0HVex1eHg9qVTeI5hCVXuGGMnFaH7GYYeJ2B_D"
}}
```

getMyBonuses

Get information about promotions / bonuses

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	(?) getGroupPromoList	(int)	Should the method return the list of promotions for product groups (1 – yes, 0 – no)
3	(?) getProductPromoList	(int)	Should the method return the list of promotions for products (1 – yes, 0 – no)

Return:

1	bonusList	(struct[])	List of promotions / bonuses
1	bonus	(struct)	Information about promotions / bonuses
1	id	(string)	Promotions / bonuses id
2	name	(string)	Promotion name
3	type_bonus	(int)	Promotion type: 0 – standard, 1 - description
4	dateFrom	(date)	[format: yyyy-mm-dd] Date of promotion start
5	dateTo	(date)	[format: yyyy-mm-dd] Date of promotion end
6	dateVisibleFrom	(date)	[format: yyyy-mm-dd] Date of visible promotion start
7	dateVisibleTo	(date)	[format: yyyy-mm-dd] Date of visible promotion end
8	type	(int)	Target type: 1-point, 2-quantitative, 3-value, 4-increase, 5-value increase
9	valueCompare	(float)	The value of the compare
10	currentValue	(float)	The current value obtained
11	currentShow	(string)	The current value to display
12	targetValue	(string)	Target Value
13	targetIncrease	(float)	Target increase
14	progress	(float)	Target progress
15	isAvailable	(int)	Is available (achieved) 0-no, 1-yes
16	desc	(string)	Description promotion
17	currency	(string)	Currency promotion
18	taxMode	(int)	Tax mode: 0-exclude, 1- include
19	targetLevelList	(struct[])	List of target levels
1	targetLevel	(struct)	Target level
1	level	(int)	Level
2	value	(float)	Value to achieve
3	description	(string)	Description
20	groupPromoList*	(struct[])	The list of promotions for product groups
1	groupPromo	(struct)	Promotion for a product group
1	symbol	(string)	Product group symbol
2	name	(string)	Product group name
2	groupPromo ...		
21	productPromoList*	(struct[])	The list of promotions for products
1	productPromo	(struct)	Promotion for a product
1	id	(string)	Product id
2	reference	(string)	Product symbol
2	productPromo ...		
22	isRanking	(int)	Customer ranking: 0-no, 1-yes
23	rankingPromoList	(struct[])	Customers ranking + 2 additional ranking places
1	rankingPromo	(struct)	Information about ranking
1	customerId	(string)	Customer Id
2	positionNumber	(int)	Position in ranking
3	currentValue	(int)	current score
2	rankingPromo ...		
24	targetLevelPeriodList	(struct[])	List of periods
1	targetLevelPeriod	(struct)	Period

1	achieved	(float)	Number of units achieved
2	dateFrom	(date)	[yyyy-mm-dd] Beginning of period
3	dateTo	(date)	[yyyy-mm-dd] End of period
4	isAvailable	(int)	Bonus Availability (Goal Achieved) 0-No, 1-Yes
5	left	(float)	Number of units left to achieve
6	level	(int)	Period number
7	percentage	(float)	Percentage of period goal achievement
8	value	(float)	Goal to be achieved in the period

2 targetLevelPeriod ...

2 bonus ...

Error codes:

none

Remarks:

* returns at most 500 items

doSetRodoStatus

Toggle agreement of the processing personal data

Parameters:

1	sessionId	(string)	Session Id
2	newStatus	(int)	Agreement status (1-agreement, 2-withdrawal)
3	clientIP	(string)	Client IP (*)

Return:

1	status	(int)	1-success
---	--------	-------	-----------

Error codes:

ERR_INVALID_STATUS	Invalid agreement status
ERR_INTERNAL	Internal server error
ERR_CLIENT_IP_NOT_AVAILABLE	Invalid client IP

Comments:

* Parameter clientIP is available only in selected connection types.

getTree

Get products categories tree

Parameters:

1	sessionId	(string)	Session id
2	parId	(string)	Parent category id (if empty, function returns main categories)

Return:

1	nodeList	(struct[])	Node List
1	node	(struct)	Node
1	catId	(string)	Category id
2	parId	(string)	Parent category id
3	name	(string)	Category name
14	hasChild	(int)	Has child categories (0 = NO, 1 = YES)

Error codes:

TREE_KATID_LEN	Invalid category id
----------------	---------------------

Request example:

```
{
  "getTree": {
    "parid": "000001",
    "sessionId": "....."
  }
}
```

Response example:

```
{
  "getTreeResponse": {
    "nodeList": {
      "node": [
        {
          "catid": "000002",
          "name": "podkategoria 1",
          "parid": "000001",
          "hasChild": 1
        },
        {
          "catid": "000003",
          "name": "podkategoria 2",
          "parid": "000001",
          "hasChild": 0
        },
        {
          "catid": "000010",
          "name": "podkategoria 3",
          "parid": "000001",
          "hasChild": 1
        }
      ]
    }
  }
}
```


PRODUCTS

getProductsInfo

Get information about products

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	productList	(struct[])	List of products (max. 50)
	1 product	(struct)	Information about product
	1*(?) id	(string)	Product unique id
	2*(?) reference	(string)	Product reference
	3*(?) tecidd	(string)	TecDoc supplier id
	4*(?) tecnum	(string)	TecDoc product id
	5*(?) ILkod	(string)	ILKod product id
	6*(?) motonet	(string)	Motonet number
	2 product ...		
3	showExternalStockInfo	(int)	Return information about external stock (0-no, 1-only local, 2-local and remote).
	4 showAdditionalInfo	(int)	Return additional information about product (0-no, 1-yes)
5	showPhotos	(int)	Return product photos (0-no, 1-yes)
6	showSubProductsInfo	(int)	Return secondary products (0-no, 1-yes)
7	showBarcodes	(int)	Should I return inferior goods?
8	showDeliveryDetails *	(int)	Should I return delivery information to the warehouse?

Return:

1	productInfoList	(struct[])	Product list
	1 productInfo	(struct)	Information about product
	1 id	(string)	Product unique id
	2 type	(int)	Product type (0-article, 1-service)
	3 reference	(string)	Product reference
	4(?) reference2	(string)	Product reference 2
	5(?) ILkod	(string)	ILkod product id
	6(?) motonet	(string)	Motonet number
	7(?) tecidd	(string)	TecDoc supplier id
	8(?) tecnum	(string)	TecDoc product id
	9 name	(string)	Product name in user language
	10 groupSymbol	(string)	Product group symbol
	11 isGroupPromo	(int)	Is group promoted for client (always 0)
	12 groupName	(string)	Product group name
	13(?) group2	(string)	Group 2 symbol
	14 decimals	(int)	Decimal places (0-3)
	15 tax	(float)	Tax rate
	16(?) priceCatExclTax	(float)	Catalog price without tax
	17(?) priceCatInclTax	(float)	Catalog price with tax
	18 currency	(string)	Currency
	19 unit	(string)	Metric unit
	20 blocked	(int)	Sale lock (1-on, 0-off)
	21 files	(int)	Does product have attach any files?(1-yes, 0-no)
	22(?) hasDeposit	(int)	Does product have deposit? (1-yes, 0-no)
	23(?) depositPriceExclTax	(float)	Deposit value without tax
	24(?) depositPriceInclTax	(float)	Deposit value with tax
	25(?) hasProductDeposit	(int)	Does product have a material deposit (1 - yes, 0 - no)
	26(?) expectedDeliveryDate	(date)	[format: yyyy-mm-dd] Expected delivery date
	27 isReplacement	(int)	Has replacement (1 - yes, 0 - no)
	28 isPhoto	(int)	Has photo(1 - yes, 0 - no)
	29(?) producer	(string)	Producer

30	gtuCode	(string)	GTU code
31	stockInfoList	(struct[])	Information about stock
1	stockInfo	(struct)	Information about stock
1	warehouseId	(string)	Warehouse id
2	viewOnly	(int)	Is ordering disabled (1=yes, 0=no)
3	inventory	(int)	Inventory in progress (1=yes, 0=no)
4	quantityShow	(string)	Stock available [include price list] (string, e.g. "> 5")
5	quantity	(float)	Stock available [include price list] (number)
6	quantityRealShow	(string)	Stock available [in warehouse] (string, e.g. "> 5")
7	quantityReal	(float)	Stock available [in warehouse] (number)
8(?)	priceExclTax	(float)	Price without tax
9(?)	priceInclTax	(float)	Price with tax
10(?)	priceBeforePromotionExclTax	(float)	Price before promotion without tax
11(?)	priceBeforePromotionInclTax	(float)	Price before promotion with tax
12(?)	promotionPriceExclTax	(float)	Promotion price without tax
13(?)	promotionPriceInclTax	(float)	Promotion price with tax
14(?)	promotionDiscount	(float)	Discount promotion
15	quantityMultiplier	(int)	Multiplier (0-off)
16(?)	externalStockInfo	(struct)	Information about external stock
1	warehouseShow	(string)	Name of warehouse
2	quantityShow	(string)	Stock available (string, e.g. "> 5")
3	quantity	(float)	Stock available (number)
4(?)	deliveryTimeMin	(int)	Minimal delivery time in hours
5(?)	deliveryTimeShow	(string)	Delivery time (string, e.g. "1 - 2 days")
6(?)	estArrivalTime	(string)	shortest delivery time to this warehouse
7(?)	estDeliveryTime	(string)	Closing date of the nearest route after the shortest delivery date from a given warehouse
17(?)	additionalStockInfo	(struct[])	Additional information about stock
1(?)	sale	(int)	Sale active (1=yes, 0=no)
18.	estDeliveryTime	(string)	The closing date of the nearest route in a given warehouse
19.	deliveryDetailsList	(struct[])	Warehouse delivery information
1	deliveryDetail	(struct)	Information about product availability
1	estArrivalTime	(string)	delivery date to this warehouse
2	estDeliveryTime	(string)	the closing date of the nearest route after the delivery date from a given warehouse
3	quantity	(float)	Quantity available [taking into account price lists] (in numbers)
4	quantityShow	(string)	Quantity available [including price lists] to display (e.g. "> 5")
5	quantityReal	(float)	Quantity available [in stock] (in numbers)
6	quantityRealShow	(string)	Quantity available [in stock] to display (e.g. "> 5")
2.	deliveryDetail	...	
2	stockInfo	...	
32(?)	photoList	(struct[])	List of product photos
1	photo	(struct)	Information about single photo
1	url	(string)	Photo URL
2	description	(string)	Photo description
3	isDefault	(int)	Default photo (1=yes, 0=no)
2	photo	...	
33	subProductList	(struct[])	List of secondary products
1	subProductInfo	(struct)	Information about secondary product
1	id	(string)	Product id
2	reference	(string)	Product reference
3	name	(string)	Product name in user language
4	tax	(float)	Tax rate

5 unit	(string)	Unit
6 decimals	(int)	Decimal places (0-3)
7 quantityMultiplier	(int)	Multiplier (0-off)
8 main	(int)	1-main product 0-other
9 currency	(string)	Currency
10 size	(string)	Size
11 color	(string)	Color
12 subBarCodeList	(struct)	List of barcodes
1 subBarCode	(struct)	Information about single barcode
1 code	(string)	Barcode
2 quantity	(float)	Product quantity
3 default	(string)	Default barcode (1-yes, 0-no)
13 subBarCodeList ...		
2 subProductInfo ...		
34. barCodeList	(struct[])	Barcode List
1. barCode	(struct)	Barcode
1. code	(string)	Barcode
2. quantity	(float)	Quantity
3. default	(string)	Default barcode
2. barCode ...		
2 productInfo ...		

Error codes:

ERR_PRODUCTLIST_OVERSIZED Product list size exceeded (max: 200)

Comments:

- * One of these is required: "id" or "reference", or both "tecidd" and "tecnum", or "motonet".
- * In case of sending TecDoc data or Motonet number - information about first fitting product is returned.
- * Informations about products are returned in the same order as in request.
- * If product is not found, not available for specified user-type, not found in any warehouse available for client or client doesn't have access to any warehouse, a blank struct <productInfo /> will be returned.
- * **The showDeliveryDetails parameter is responsible for displaying the deliveryDetailsList structure and the estArrivalTime and estDeliveryTime fields**

Example:

```
Request:
{
  "getProductsInfo": {
    "sessionId": "Gq0HVex1eHg9qVTeI5hCVXuGGMnFaH7GYyeJ2B_D",
    "productList": {
      "product": [
        { "reference": "0-001-231-030" },
        { "reference": "SYMBOL123" }
      ]
    },
    "showExternalStockInfo": "2"
  }
}
```

```
Response:
{
  "getProductsInfoResponse": {
    "productInfoList": {
      "productInfo": {
        "id": "003LFT",
        "type": "0",
        "reference": "0-001-231-030",
        "name": "Osłona przegubu /BOSCH/",
        "decimals": "2",
        "tax": "22",
        "priceCatExclTax": "52.49",
        "priceCatInclTax": "64.04",
        "currency": "zł",
        "unit": "szt",
        "blocked": "0",
        "files": "0",
        "hasDeposit": "0",
        "color": "pink",
        "size": "xxL",
        "barcodeList": {
          "barcode": [
            {
              "code": "1234567",
              "quantity": "2",
              "default": "0"
            },
            {
              "code": "2345678",
              "quantity": "1",
              "default": "1"
            }
          ]
        },
        "stockInfoList": {
```

```

"stockInfo": [
  {
    "warehouseId": "XX",
    "inventory": "0",
    "quantityShow": "> 5.00",
    "quantity": "6.00",
    "priceExclTax": "55.57",
    "priceInclTax": "67.79",
    "priceBeforePromotionExclTax": "75.35",
    "priceBeforePromotionInclTax": "96.22",
    "quantityMultiplier": "0",
    "externalStock": {
      "warehouseShow": "xxx",
      "quantityShow": "2.00",
      "quantity": "2.00",
      "deliveryTimeMin": "24",
      "deliveryTimeShow": "1 day"
    }
  },
  {
    "warehouseId": "CE",
    "inventory": "0",
    "quantityShow": "2.00",
    "quantity": "2.00",
    "priceExclTax": "5000.00",
    "priceInclTax": "6100.00",
    "quantityMultiplier": "0"
  },
  {
    "warehouseId": "OL",
    "inventory": "0",
    "quantityShow": "0.00",
    "quantity": "0.00",
    "priceExclTax": "1294.61",
    "priceInclTax": "1579.42",
    "quantityMultiplier": "0"
  }
],
"additionalInfo": {
  "sale": "1"
}
}
}
}
}
}

```

getProductFileUrl

Put file to download and return an URL adress

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	fileId	(string)	File id
3	productId	(string)	Product id

Return:

1	url	(string)	File url adress
---	-----	----------	-----------------

Error codes:

ERR_WRONG_FILE_ID	Invalid file id
ERR_SERVER_NOT_CONFIGURED	Server is not configured correctly

Example:

```
Request:
{
  "getProductFileUrl": {
    "sessionId": "YhQ4owGibJeiz2LByY2Im7yvik3F27y0017943_D",
    "fileId": "00000013",
    "productId": "0003B4"
  }
}

Response:
{
  "getProductFileUrlResponse": {
    "url": http://produkty.com/katalog/1/produkt1.txt
  }
}
```

getProductStockChanges

Get paged information about product stock changes

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	warehouseId	(string)	Warehouse id
3	fromDt	(string)	[yyyy-mm-dd hh:ii:ss] Starting date for stock changes analysis (*)
4	(?) pageSize	(int)	Number of results on a single page (max. 1000, def. 100) (*)
5	(?) pageNumber	(int)	Page number (def. 1) (*)

Return:

1	productStockList	(struct[])	Product stock list
1	productStock	(struct)	Information about product stock status
1	productId	(string)	Product id
2	changeDt	(string)	[yyyy-mm-dd hh:ii:ss] The most recent stock change date
3	quantity	(float)	Product quantity
4	quantityExt	(float)	External product quantity
2	productStock ...		

Error codes:

ERR_INVALID_WAREHOUSE	Invalid warehouse
ERR_INVALID_FROM_DATE	Invalid starting date
ERR_WAREHOUSE_ACCESS_DENIED	Warehouse access denied
ERR_RESULT_LIMIT	Result limit exceeded

Request example:

```
{
  "getProductStockChanges": {
    "sessionId": "vG1mNEXIAPL7LXUs5d0vzyGeYex7ra80024259_D",
    "warehouseId": "01",
    "fromDt": "2021-09-01 16:17:25"
  }
}
```

Response example:

```
{
  "getProductStockChangesResponse": {
    "productStockList": {
      "productStock": [
        {
          "changeDt": "2021-09-21 14:07:28",
          "productId": "0003AZ",
          "quantity": "1253",
          "quantityExt": "0"
        },
        {
          "changeDt": "2021-09-21 14:07:28",
          "productId": "0003B0",
          "quantity": "8468",
          "quantityExt": "0"
        }
      ]
    }
  }
}
```

doSearchProducts

Search products

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	searchString	(string)	Searched phrase or multiple phrases separated with space
3	searchMode	(struct[])	Search option
	1 modeName	(string)	Available options: id, reference, name, tecdoc, treeid
	2 modeName ...		

Return:

1	productInfoList	(struct[])	List of found products
	1 productInfo	(struct)	Informations about found product
	1 id	(string)	Product id
	2 reference	(string)	Product reference
	3 name	(string)	Product name in user language
	4 source	(int)	Source: 1-id, 2-reference, 3-reference2, 4-original number, 5-supplier reference, 6-name, 7-tecdoc, 8-treiid
	2 productInfo ...		

Error codes:

ERR_INVALID_WAREHOUSE	Invalid warehouse
ERR_WAREHOUSE_ACCESS_DENIED	No access to warehouse
ERR_INVALID_PRODUCT	Invalid product

Comments:

Function returns max. 50 products containing searching phrase.

Example:

```
Request:
{
  "doSearchProducts": {
    "sessionId": "Gq0HVex1eHg9qVTeI5hCVXuGGmFah7GYyeJ2B_D",
    "searchString": "96575",
    "searchMode": {
      "modeName": [
        "name",
        "reference"
      ]
    }
  }
}

Response:
{
  "doSearchProductsResponse": {
    "productInfoList": {
      "productInfo": {
        "id": "0AX451",
        "reference": "96575",
        "name": "Osłona przegubu /BOSCH/",
        "source": "2"
      }
    }
  }
}
```


getReplacements

Get product replacements

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	productId	(string)	Product id
3	(?) warehouseId	(string)	Warehouse id

Return:

1	replacementList	(struct[])	List of replacements
1	replacementInfo	(struct)	Information about replacement (product)
1	id	(string)	Product id
2	reference	(string)	Product reference
3	name	(string)	Product name
2	replacementInfo ...		

Error codes:

ERR_INVALID_WAREHOUSE	Invalid warehouse
ERR_WAREHOUSE_ACCESS_DENIED	No access to warehouse
ERR_INVALID_PRODUCT	Invalid product

Comments:

Replacements are sorted according to program settings

doBuyNow

Purchase method Buy Now by goods issue note.

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2(?)	warehouseId	(string)	Warehouse id, which products will be ordered. (default: The default warehouse for the client)
3(?)	documentId	(string)	Document id, which order products. (default: Today unprintable document client)
4(?)	forceNewOrder	(int)	Force making a new order (1=yes, 0=no, default 0)
5	productOrderList	(struct[])	List of products to order
1	productOrder	(struct)	Information about product
1(?)	* id	(string)	Product id
2(?)	* reference	(string)	Product reference
3(?)	* tecidd	(string)	TecDoc supplier id
4(?)	* tecnum	(string)	TecDoc product id
5(?)	* ILkod	(string)	ILkod product id
6(?)	* motonet	(string)	Motonet number
7	quantity	(float)	Quantity to order
2	productOrder ...		

Return:

1	documentInfo	(struct)	Information about document, which ordered products.
1	id	(string)	Document id
2	number	(string)	Document number
2	productOrderInfoList	(struct[])	Information about ordered products
1	productOrderInfo	(struct)	Information about ordered product (if success)
1	id	(string)	Product id
2	reference	(string)	Product reference
3	ILkod	(string)	ILkod product id
4	name	(string)	Product name in user language
5	decimals	(int)	Decimal places (0-3)
6	tax	(float)	Tax rate in % ("-1" - exempt, "-2" - untaxed)
7	priceExclTax	(float)	Price without tax
8	priceInclTax	(float)	Price with tax
9	currency	(string)	Currency
10	unit	(string)	Unit
11	warehouseId	(string)	Warehouse id
12	quantity	(float)	Quantity ordered
2	productOrderInfo	(struct)	Information about ordered product (if error)
1	error	(string)	Error message
3	productOrderInfo ...		

Error codes:

ERR_METHOD_UNAVAILABLE	Method unavailable (user-type connects from wrong address / method does not exist)
ERR_NO_WAREHOUSE	No available warehouses to order
ERR_WAREHOUSE_UNAVAILABLE	In warehouse %% user can't make orders
ERR_METHOD_BLOCKED	Method is blocked
ERR_PRODUCTORDERLIST_OVERSIZED	Products list oversized (max: 50)
ERR_CANT_FORCE_NEW_ORDER	Can't force new order (orderId passed)
ERR_INVALID_DOCUMENT_ID	Invalid document id
ERR_ORDER_LOCKED	Document is locked by other user

Comments:

* One of these is required: "id" or "reference" or both "tecidd" and "tecnum" or "motonet"

- * In case of sending TecDoc or Motonet data - information about first fitting product is returned.
- * Informations about products are returned in the same order as in request.

COMPLAINTS

doCreateComplaint

Make a complaint

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	complaintInfo	(struct)	Information about complaint
1	srcDocumentId	(string)	Source document id
2	srcItemId	(string)	Document item id
3	productId	(string)	Product id
4	(?) dateProductDelivery	(date)	[format: yyyy-mm-dd] Date of product delivery
5	(?) serialNumber	(string)	Date of product delivery
6	(?)* defectDescription	(string)	Defect description
7	quantity	(float)	Quantity
8	(?) decisionIfYes	(string)	Decision if accepted: "K"-money return, "W"-product replacement
9	(?) decisionIfNo	(string)	Decision if rejected: "Z"-recycling, "O"-product return
10	(?)* confirmReComplaint	(int)	Confirmation of recomplaining the same product (1-yes, 0-no)
11	(?) registerNumber	(string)	Vehicle registration number
12	(?) engineNumber	(string)	Engine number
13	(?) vinNumber	(string)	VIN number
14	(?) engineCapacity	(string)	Engine Capacity
15	(?) installDate	(date)	[format: yyyy-mm-dd] Date of assembly parts
16	(?) counterStateInstall	(int)	Mileage counter before assembly
17	(?) workshopInstallName	(string)	Assembly - workshop name
18	(?) workshopInstallAddress	(string)	Assembly - workshop address
19	(?) uninstallDate	(date)	[format: yyyy-mm-dd] Date of disassembly parts
20	(?) counterStateUninstall	(int)	Mileage counter before disassembly
21	(?) workshopUninstallName	(string)	Disassembly - workshop name
22	(?) workshopUninstallAddress	(string)	Disassembly - workshop address
23	(?) productionYear	(string)	Year production vehicle
24	(?) brandVehicle	(string)	Mark vehicle
25	(?) **powerValue	(int)	Power - value
26	(?) **powerUnit	(int)	Power - unit
27	(?) **installedBy	(int)	Installed by (0-workshop, 1-user)
28	(?) **uninstalledBy	(int)	Uninstalled by (0-workshop, 1-user)
29	(?) **faultFoundBy	(int)	Fault found by (0-workshop, 1-user)
30	(?) **faultWhen	(int)	When a fault was found (0-before assembly, 1-during assembly, 2-during exploitation, 3-after the collision)
31	(?) **refuseAddCosts	(int)	The complainant resigns from additional costs (0-no, 1-yes)
32	(?) compensationValue	(int)	Compensation value
33	(?) compensationCurrency	(string)	Compensation currency

Return value:

1	complaintId	(string)	Complaint id
---	-------------	----------	--------------

Error codes:

ERR_COMPLAINTS_INACTIVE	Complaints are not active
ERR_WRONG_DOCUMENT_ID	Invalid document id
ERR_WRONG_ITEM_ID	Invalid document item id
ERR_WRONG_PRODUCT_ID	Invalid product id
ERR_INVALID_QUANTITY	Invalid quantity

ERR_QUANTITY_TOO_MUCH	Too much quantity to complain (max: %%)
ERR_PRODUCT_ALREADY_COMPLAINED	Product was already complained
ERR_DEFECT_DESCRIPTION_INACTIVE	Defect description is not available
ERR_DEFECT_DESCRIPTION_REQUIRED	Defect description is required
ERR_INVALID_DECISION_IF_YES	Illegal decision if accepted
ERR_INVALID_DECISION_IF_NO	Illegal decision if rejected
ERR_CAR_DATA_INACTIVE	Edition of vehicle data is not available

Comments:

** This functionality can be limited

*** If the product was already complained, an error code `ERR_PRODUCT_ALREADY_COMPLAINED` will be returned, unless a parameter `confirmReComplaint = 1` is sent, then a re-complaint will be accepted.

getMyComplaints

Get list of complaints of logged user

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	(?)* documentIdList	(string[])	List of documents to return
	1 id	(string)	Document id
	2 id ...		
3	(?)* state	(string)	Complaint state (0-closed, 1-opened, 2-opened by customer), default 1
4	(?) filterOptions	(struct)	Document filters
	1 (?) dateFrom	(date)	[format: yyyy-mm-dd] Begin of date range. By default, documents from last 7 days are returned (if parameter documentIdList was NOT given) or from any range (if parameter documentIdList was given)
	2 (?) dateTo	(date)	[format: yyyy-mm-dd] End of date range (default: today)
5	(?) sortOptions	(struct)	Sorting options
	1 (?) sortType	(int)	Sort type (1-by date of create (default))
	2 (?) sortOrder	(int)	Sort order (1-ascending, 2-descending (default))
6	(?) pageSize	(int)	Number of documents on page (min: 1, max: 50, default: 10)
7	(?) pageNumber	(int)	Page number (default: 1)

Return:

1	complaintList	(struct[])	List of complaints
	1 complaint	(struct)	Information about complaint
	1 id	(string)	Complaint id
	2 number	(string)	Complaint number
	3 state	(string)	Complaint state (0-closed, 1-opened, 2-opened by customer)
	4 dateCreate	(date)	[format: yyyy-mm-dd] Date of create
	5 dateClose	(date)	[format: yyyy-mm-dd] Date of decision
	6 decision	(int)	Decision id
	7 decisionText	(string)	Description of the decision
	8 decisionType	(int)	Decision type: 0-none, 1-accepted, 2- rejected
	9 status	(string)	Complaint status
	10 statusText	(string)	Complaint status description
	11 statusDate	(date)	[format: yyyy-mm-dd] Date of status change
	12 productId	(string)	Product id
	13 productName	(string)	Product name
	14 productReference	(string)	Product reference
	15 quantity	(int)	Quantity complained
	16 priceNet	(int)	Price net
	17 valueNet	(int)	Value net
	18 srcDocumentType	(string)	Source document type
	19 srcDocumentNumber	(string)	Source document number
	20 srcDocumentDate	(date)	[format: yyyy-mm-dd] Date of sale
	21 srcDocumentId	(string)	Source document id
	22 defectDescription	(string)	Defect description
	23 serialNumber	(string)	Serial number
	24 decisionIfYes	(string)	Decision if accepted (id)
	25 decisionIfYesText	(string)	Decision if accepted (description)
	26 decisionIfNo	(string)	Decision if rejected (id)
	27 decisionIfNoText	(string)	Decision if rejected (description)
	28 decisionComment	(string)	Description of the decision complaint
	29 comment	(string)	Comments
	30 storeExt	(string)	Warehouse id complaint

31	currency	(string)	Currency
32	productionYear	(string)	Year production vehicle
33	brandVehicle	(string)	Mark vehicle
34	registerNumber	(string)	Vehicle registration numbe
35	engineNumber	(string)	Engine number
36	vinNumber	(string)	VIN number
37	engineCapacity	(string)	Engine Capacity
38	installDate	(date)	[format: yyyy-mm-dd] Date of assembly parts
39	counterStateInstall	(int)	Mileage counter before assembly
40	workshopInstallName	(string)	Assembly - workshop name
41	workshopInstallAddress	(string)	Assembly - workshop address
42	uninstallDate	(date)	[format: yyyy-mm-dd] Date of disassembly parts
43	counterStateUninstall	(int)	Mileage counter before disassembly
44	workshopUninstallName	(string)	Disassembly - workshop name
45	workshopUninstallAddress	(string)	Disassembly - workshop address
2	complaint	...	

Error codes:

ERR_COMPLAINTS_INACTIVE	Complaints are not active
ERR_WRONG_COMPLAINTS_STATE	Invalid complaint state

getComplaintQuantityAvailable

Get quantity can be complained for document item.

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	complaintInfo	(struct)	Information about complaint
	1 srcDocumentId	(string)	Complaint id
	2 srcItemId	(string)	Complaint number
	3 productId	(string)	Product id

Return:

1	quantity	(float)	Quantity can be complained
---	----------	---------	----------------------------

Error codes:

ERR_COMPLAINTS_INACTIVE	Complaints are not active
ERR_WRONG_DOCUMENT_ID	Wrong document id
ERR_WRONG_ITEM_ID	Wrong item id
ERR_WRONG_PRODUCT_ID	Wrong product id

getInvoicesComplaintProduct

Get list of sales documents by the specified product and complaint date range.

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	productId	(string)	Product Id
3	dateFrom	(date)	[format: yyyy-mm-dd] Begin of date range of documents returned.
4(?)	dateTo	(date)	[format: yyyy-mm-dd] End of date range of documents returned, (default: today)
5(?)	warehouseId	(string)	Warehouse id, which documents will be returned. (default: all)

Return value:

1	invoiceList	(struct[])	List of documents
1	invoice	(struct)	Information about document
1	id	(string)	Document id
2	number	(string)	Document number
3	date	(date)	[format: yyyy-mm-dd] Document date
4	warehouseId	(string)	Warehouse id
5	quantity	(float)	Quantity
6	quantityAvailable	(float)	Quantity possible to return (all)
7	decimals	(int)	Decimal places (0-3)
8	itemList	(struct[])	List of items
1	item	(struct)	Item with product
1	itemId	(string)	Item id
2	quantity	(float)	Quantity
3	quantityAvailable	(float)	Quantity possible to return
2	item ...		
2	invoice ...		

Error codes:

ERR_INVALID_PRODUCT	Invalid product id
ERR_INVALID_DATEFROM	Invalid begin of date range
ERR_INVALID_DATETO	Invalid end of date range
ERR_INVALID_WAREHOUSE	Invalid warehouse id

doComplaintPrint

Generate PDF printout of complaint

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	complaintId	(string)	Complaint Id

Return:

1	pdfData	(string)	Complaint printout (PDF coded Base64)
---	---------	----------	---------------------------------------

Error codes:

ERR_COMPLAINTS_INACTIVE	Complaint module is not available
ERR_WRONG_COMPLAINT_ID	Invalid complaint id

RESERVATION

doOrderProducts

Make reservation for products

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2(?)	warehouseId	(string)	Warehouse id in which reservation will be made (if parameter is not given, then reservation will be made in default warehouse)
3(?)	orderId	(string)	Order id in which products will be added (if parameter is not given, products will be added to newest opened order or (if not found) a new order will be created)
4(?)	dateDelivery	(date)	Date of order realization [format: yyyy-mm-dd]
5(?)	forceNewOrder	(int)	Force making a new order (1=yes, 0=no, default 0)
6(?)	onlyFoundItems	(int)	Should the error be returned if any of the items is not found (1=yes, 0=no, default 0) (only if forceNewOrder = 1)
7(?)	newOrderInfo	(struct)	Information about new order (only if forceNewOrder = 1)
1(?)	externalId	(string)	Order external id
2(?)	comment	(string)	Comment to order
3(?)	srcDocId	(string)	Source document id
4** (?)	incotermsCode	(string)	Incoterms code
5** (?)	additionalInfo	(string)	Additional information
6(?)	isReward	(int)	Bring out reward marker (1 – yes, 0 – no (default))
7(?)	packMergeKey	(int)	The key used to merge orders into a single package (if empty, the order will be packed separately)
8(?)	retShelfMerge	(int)	Specifies the order in which products are taken out of return shelves (0 – normal, 1 – first, 2 – last)
9(?)	deliveryAddress**	(struct)	Delivery address
1	name	(string)	Name and surname
2	company	(string)	Company name
3	street	(string)	Street, street number
4	postcode	(string)	Post code
5	city	(string)	City
6	country	(string)	Country
7	phone	(string)	Phone
8	email	(string)	E-mail
10(?)	additionalDeliveryAddress**	(struct)	Delivery address 2
1	name	(string)	Name and surname
2	company	(string)	Company name
3	street	(string)	Street, street number
4	postcode	(string)	Post code
5	city	(string)	City
6	country	(string)	Country
7	phone	(string)	Phone
8	email	(string)	E-mail
11(?)	pickupPointAddress	(struct)	Pick up point address
1	name	(string)	Name and surname
2	company	(string)	Company name
3	street	(string)	Street, street number
4	postcode	(string)	Post code
5	city	(string)	City
6	country	(string)	Country
7	phone	(string)	Phone
8	email	(string)	E-mail

9	pickupPointId	(string)	DHLD - Number of the packstation
10	packstationPostNumber	(string)	DHLD - Post Nummer of the postfiliale
12(?)	payment	(string)	Payment information
1	type	(int)	Payment type (2-cash, 3-cash in term, 5-bank wire, 6-cash/bank wire, 7-COD, 8-credit card)
2(?)	days	(int)	Payment term in days (if not given, default value will be inserted)
13(?)	routeId	(string)	Route id
15**(?)	packageProductId	(string)	Product Id- package
16**(?)	deliveryOnSaturday	(int)	Delivery on Saturday (1=yes, 0=no, default 0)
17**(?)	sendOnSaturday	(int)	Send on Saturday (1=yes, 0=no, default 0)
18	(?) shipmentType	(string)	Shipment type
19	(?) handoverToDD	(int)	Hand over to a goods issue note (1 – yes, 0 – no, by default the parameter is taken from the customer settings)
20	(?) superSupplier	(int)	Is the order to the super supplier (1 – yes, 0 – no (default))
21	(?) customerDestinationWarehouseId	(string)	For warehouse orders, the symbol of the warehouse where the order was created
22	(?) isCustomerSupply	(int)	For warehouse orders, is is the supply order
8	productOrderList	(struct[])	List of products to order
1	productOrder	(struct)	Information about product
1*	(?) id	(string)	Product id
2*	(?) reference	(string)	Product reference
3*	(?) tecidd	(string)	TecDoc supplier id
4*	(?) tecnum	(string)	TecDoc product id
5*	(?) ILkod	(string)	ILkod product id
6*	(?) motonet	(string)	Motonet number
7	quantity	(float)	Quantity to order
8(?)	requireStockBlock	(int)	Information whether product should be ordered only if user can block quantity in warehouse (1=yes, 0=no, default 0)
9(?)	finalDocumentType	(int)	Type of final document, "conversion direction" (0-default, 1-invoice, 2-bill, default 0)
10(?)	customerReference	(string)	Customer reference
11(?)	packageType	(int)	Package type (0-self, 1-client, 2-none, default 0)
12(?)	description	(string)	Item description
13(?)	dataExpire	(date)	[format: yyyy-mm-dd] Minimum expiration date
14(?)	externalItemId	(string)	External item id
2	productOrder ...		

Return:

1	orderInfo	(struct)	Information about order
1	id	(string)	Order id
2	number	(string)	Order number
2	productOrderInfoList	(struct[])	Information about ordered products
1	productOrderInfo	(struct)	Information about ordered product (if success)
1	id	(string)	Product id
2	reference	(string)	Product reference
3	ILkod	(string)	ILkod product id
4	motonet	(string)	Motonet number
5	name	(string)	Product name in user language
6	decimals	(int)	Decimal places (0-3)
7	tax	(float)	Tax rate
8	priceExclTax	(float)	Price without tax
9	priceInclTax	(float)	Price with tax
10	currency	(string)	Currency
11	unit	(string)	Unit
12	warehouseId	(string)	Warehouse id

13 quantity	(float)	Quantity ordered
14 quantityBlocked	(float)	Quantity blocked in warehouse
15 quantityNotBlocked	(float)	Quantity not blocked in warehouse
2 productOrderInfo	(struct)	Information about ordered product (if error)
1 error	(string)	Error message
3 productOrderInfo ...		

Error codes:

ERR_NO_WAREHOUSE	No available warehouses
ERR_WAREHOUSE_UNAVAILABLE	In warehouse %% user can't make orders
ERR_METHOD_BLOCKED	Method is blocked
ERR_PRODUCTORDERLIST_OVERSIZED	Products list oversized (max: 200)
ERR_CANT_FORCE_NEW_ORDER	Can't force new order (orderId passed)
ERR_COMMENT_TOO_LONG	Comment too long (max 30 chars)
ERR_COURIER_INFO_EDIT_INACTIVE	Editing parameters courier unavailable
ERR_EXTERNALID_TOO_LONG	External order number is too long (max 47 chars)
ERR_EXTERNALID_ALREADY_EXISTS	External order number already exists
ERR_DATEDELIVERY_WRONG_FORMAT	Wrong order delivery date type
ERR_DELIVERY_ADDRESS_INACTIVE	Delivery address support unavailable
ERR_DELIVERY_ADDRESS_EMPTY_NAME_AND_COMPANY	Empty delivery address
ERR_ADDITIONAL_DELIVERY_ADDRESS_EMPTY_NAME_AND_COMPANY	Empty additional delivery address
ERR_INCOTERMS_INACTIVE	INCOTERMS code support unavailable
ERR_INCOTERMS_INVALID_CODE	Wrong INCOTERMS code
ERR_ADDITIONAL_INFO_INACTIVE	Additional information about product unavailable
ERR_PACKAGE_PRODUCT_INACTIVE	Product package unavailable
ERR_PACKAGE_PRODUCT_ID_INVALID	Wrong package product id
ERR_PAYMENT_EDIT_NOT_AVAILABLE	Editing payment type unavailable
ERR_PAYMENT_INVALID_TYPE	Wrong payment type
ERR_PAYMENT_TYPE_NOT_AVAILABLE	Selected payment type unavailable
ERR_ROUTE_INVALID	Selected route is invalid or isn't available
ERR_INVALID_CUSTOMER	Invalid customer
ERR_ITEM_NOT_FOUND	At least one product has not been found
ERR_INVALID_HANDOVERTODD	Handing over to goods issue note is not available
ERR_LOGM_SERVICE_TYPE_INVALID	Invalid logistic customer type
ERR_INVALID_FINAL_DOCUMENT_TYPE	Invalid final document type

Comments:

- * One of these is required: "id", or "reference", or both "tecidd" and "tecnum", or "motonet".
- * In case of sending TecDoc or Motonet data - information about first fitting product is returned.
- * Informations about products are returned in the same order as in request.
- ** **This functionality can be limited**

Example:

```
Request:
{
  "doOrderProducts": {
    "sessionId": "2BZY8zSx0D00UmM0ylFKSZjMsgDG564YxTMvju_D",
    "productOrderList": {
      "productOrder": [
        {
          "reference": "0-001-231-030",
          "quantity": "2"
        },
        {
          "reference": "0-001-268-320",
          "quantity": "1"
        },
        {
          "reference": "SYMBOL123",
          "quantity": "2"
        }
      ]
    }
  }
}
```

```
Response:
{ "doOrderProductsResponse": {
  "orderInfo": {
    "id": "00001269",
    "number": "63"
  },
  "productOrderInfoList": {
    "productOrderInfo": [
      {
        "id": "003LFT",
        "reference": "0-001-231-030",
        "name": "Osłona przegubu /BOŚCH/",
        "decimals": "2",
        "tax": "22",
        "warehouseId": "XX",
        "unit": "szt",
        "priceExclTax": "55.57",
        "priceInclTax": "67.79",
        "currency": "zł",
        "quantity": "2",
        "quantityBlocked": "2",
        "quantityNotBlocked": "0"
      },
      {
        "id": "003LGF",
        "reference": "0-001-268-320",
        "name": "Osłona przegubu wew. /BOSCH/",
        "decimals": "0",
        "tax": "22",
        "warehouseId": "XX",
        "unit": "szt",
        "priceExclTax": "6111.11",
        "priceInclTax": "7455.56",
        "currency": "zł",
        "quantity": "1",
        "quantityBlocked": "0",
        "quantityNotBlocked": "1"
      }
    ],
    { "error": "brak kartoteki w magazynie" }
  ]
}
}
```

getMyOrders

Get order list for logged user

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2(?)	orderIdList	(string[])	List of orders
	1 id	(string)	Order id
	2 id ...		
3(?)	filterOptions	(struct)	Orders filters
	1(?) externalId	(string)	The customer order number
	2(?) isReward	(int)	Filtering orders by tags "bring out reward" (1 – tag is set to yes, 0 – tag is set to no, no default filter)
4(?)	getItems	(int)	Return information about items (1=yes, 0=no; default 0)
5(?)	getLoyaltyInfo	(int)	Return data of loyalty system (1 – yes, 0 – no (default))

Return:

1	orderList	(struct[])	Order list
	1 order	(struct)	Information about order
	1 id	(string)	Document id
	2 number	(string)	Document number
	3 justNumber	(string)	Document number without postscript
	4 externalId	(string)	External order number
	5 type	(string)	Document type: WZ – invoice, DD – goods issue note, ZW – invoice correction
	6 warehouseId	(string)	Warehouse id
	7 date	(date)	[format: yyyy-mm-dd] Document date
	8 dateExpire	(date)	[format: yyyy-mm-dd] Date of expire
	9 dateRealization	(date)	[format: yyyy-mm-dd] Date of realization
	10 isClosed	(int)	Is order closed (1=yes, 0=no)
	11 additionalInfo	(string)	Additional information for the document
	12 dateAddInfo	(date)	Date of editing additional information to the document
	13 recipientId	(string)	Receiver id
	14 recipientName	(string)	Receiver name
	15 recipientShortName	(string)	Abbreviated receiver name
	16 isDoReturnNow	(int)	Is document created with the doReturnNow method (1 – yes, 0 - no)
	17 comment	(string)	Comment
	18 currency	(string)	Currency
	19 localCurrency	(string)	Local currency
	20 currencyRate	(float)	Currency rate
	21 exportType	(int)	Type of bill (0-country, 1-export, 2-country in currency)
	22 isTaxLC	(int)	Is tax in local currency (1) or document currency (0)
	23 isValueLC	(int)	Is value in local currency (1) or document currency (0)
	24 valueExclTaxLC	(float)	Price without tax in local currency
	25 valueInclTaxLC	(float)	Price with tax in local currency
	26 valueTaxLC	(float)	Tax amount in local currency
	27 valueExclTax	(float)	Value without tax
	28 valueInclTax	(float)	Value with tax
	29 valueTax	(float)	Tax value
	30 isEU	(int)	EU document (1=yes, 0=no)
	31 isReExport	(int)	Is document re-exported (1=yes, 0=no)
	32 paymentType	(int)	Payment type: 1-none, 2-cash, 3-cash in term, 4-check, 5-bank wire, 6-cash/bank wire, 7-COD, 8-credit card, 9-installment
	33 paymentDays	(float)	Payment term (days)
	34 isPrinted	(int)	Was the document printed (1=yes, 0=no)
	35 wmsStatus	(int)	Issue status of the document (0 – not issued, 1 – issue due, 2 – issue complete)

36	paymentStatus	(int)	Payment status (0 – unknown, 1 – paid, 2 – not paid, 3 – not paid and overdue)
37	loyaltyPoints	(int)	Number of points earned in loyalty system (only if getLoyaltyInfo = 1)
38	loyaltyRewardPoints	(int)	Number of points that will be deducted from client account after “bring out reward” (return only if getLoyaltyInfo = 1)
39	isDeliveryConfirmed	(int)	Is delivery confirmed (1 – yes, 0 – no)
40	isReward	(int)	Content of tag “bring out reward” (0 – no, 1 – yes)
41	routeId	(string)	Route id
42	isFromDeficiencies	(int)	Is order created because of deficiencies (0 – no, 1 – yes)
43	isExternalOrderInProgress	(int)	Is external order currently in progress (0 – no, 1 – yes)
44	itemsCount	(int)	Number of items in order
45	items	(struct[])	List of order items (only if getItems = 1)
	1 item	(struct)	Information about document item
	1 id	(string)	Item id
	2 ordinal	(int)	Item position in document
	3 productId	(string)	Product id
	4 productType	(int)	Product type (0-article, 1-service)
	5 productReference	(string)	Product reference
	6 productName	(string)	Name of product in user language
	7 (?) motonet	(string)	Motonet number
	8 decimals	(int)	Decimal places (0-3)
	9 tax	(float)	Tax rate
	10 quantity	(float)	Quantity ordered
	11 quantityBlocked	(float)	Quantity blocked in warehouse
	12 isExternal	(int)	Is item ordered in external warehouse (1/0)
	13 externalInfo	(string)	Information about delivery time
	14 isDeposit	(int)	Is item a deposit (1/0)
	15 unit	(string)	Unit
	16 priceExclTax	(float)	Price without tax
	17 valueExclTax	(float)	Value without tax
	18 priceInclTax	(float)	Price with tax
	19 valueInclTax	(float)	Value with tax
	20 valueTax	(float)	Tax value
	21 priceExclTaxLC	(float)	Price without tax in local currency
	22 valueExclTaxLC	(float)	Value without tax in local currency
	23 priceInclTaxLC	(float)	Price with tax in local currency
	24 valueInclTaxLC	(float)	Value with tax in local currency
	25 valueTaxLC	(float)	Tax value in local currency
	26 currency	(string)	Currency
	27 discountPercent	(float)	Discount %
	28 startingPriceExclTax	(float)	Price without tax before discount
	29 startingPriceInclTax	(float)	Price with tax before discount
	30 (?) priceCatExclTax	(float)	Catalogue price without tax in document currency
	31 (?) priceCatInclTax	(float)	Catalogue price with tax in document currency
	32 packageType	(int)	Package type: 0 – original package, 1 – customer’s package, 2 – none (only if package types are enabled in the system)
	33 customerReference	(string)	Customer reference (only if enabled in the system)
	34 finalDocumentType	(int)	The final type of the document (0 – default, 1 – invoice, 2 – receipt)
	35 loyaltyRewardPoints	(int)	Number of points that will be deducted from client account after “bring out reward” (return only if getLoyaltyInfo = 1)
	36 description	(string)	Description for the item
	2 item ...		
	2 order ...		

2 ordersCount

(int)

Number of orders (can be approximate)

Error codes:

none

Example:

```
Request:
{
  "getMyOrders": {
    "sessionId": "v5etQUyinDnHosbSzjy7iLdj3Us4GaEr5ILMt9_D",
    "getItems": "1"
  }
}

Response:
{
  "getMyOrdersResponse": {
    "orderList": {
      "order": {
        "id": "01520531",
        "number": "RE 11426",
        "warehouseId": "CE",
        "date": "2013-01-01",
        "dateExpire": "2013-01-01",
        "dateRealization": "2013-01-07",
        "isClosed": "0",
        "comment": "testowe zamówienie",
        "currency": "zł",
        "localCurrency": "zł",
        "currencyRate": "0",
        "exportType": "0",
        "valueExclTax": "100.00",
        "valueInclTax": "123.00",
        "valueTax": "23.00",
        "itemsCount": "1",
        "items": {
          "item": {
            "id": "R-3975485",
            "ordinal": "1",
            "productId": "105275",
            "productType": "0",
            "productReference": "123-456-789",
            "productName": "Filtr kabiny VAG",
            "decimals": "0",
            "tax": "23",
            "quantity": "2.000",
            "quantityBlocked": "1.000",
            "isExternal": "0",
            "isDeposit": "0",
            "unit": "szt",
            "priceExclTax": "50.00",
            "priceInclTax": "61.50",
            "valueExclTax": "100.00",
            "valueInclTax": "123.00",
            "valueTax": "23.00",
            "currency": "zł",
            "discountPercent": "50.0",
            "startingPriceExclTax": "100.00",
            "startingPriceInclTax": "123.00"
          }
        }
      }
    },
    "ordersCount": "1"
  }
}
```

getOrderStockInfo

Get information about availability of order items

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	orderId	(string)	Order id

Return value:

1	orderId	(string)	Order id
2	itemsCount	(int)	Number of items
3	items	(struct[])	List of items
1	item	(struct)	Information about item
1	id	(string)	Item id
2	ordinal	(int)	Item position in document
3	productId	(string)	Product id
4	productType	(int)	Product type (0-article, 1-service)
5	productReference	(string)	Product reference
6	productName	(string)	Product name in user language
7	decimals	(int)	Decimal places (0-3)
8	quantity	(float)	Quantity ordered
9	quantityInStock	(float)	Quantity available
10	stockStatus	(int)	Availability status (0-in warehouse, 1-partial external availability, 2-full external availability, "-1" product external unavailable)
11	stockInfo	(string)	Text info about availability
12	unit	(string)	Unit
2	item ...		

Error codes:

ERR_WRONG_ORDER_ID	Wrong order id
ERR_ORDER_ALREADY_CLOSED	Order is already closed
ERR_ORDER_LOCKED	Order is locked by other user

Example:

```
Request:
{
  "getOrderStockInfo": {
    "sessionId": "15LADxWUF9w4RFxoj7rmsOdiGVTApT3AoE1S10_D",
    "orderId": "01512768"
  }
}

Response:
{
  "getOrderStockInfoResponse": {
    "orderId": "01512768",
    "itemsCount": "10",
    "items": {
      "item": [
        {
          "id": "R-3899087",
          "ordinal": "1",
          "productId": "40458",
          "productType": "0",
          "productReference": "11.5855",
          "productName": "Linka ham.OPEL ASTRA /COFLE/ kombi",
          "decimals": "0",
          "quantity": "3",
          "quantityInStock": "3",
          "stockStatus": "0",
          "stockInfo": "w magazynie",
          "unit": "szt"
        },
        {
          "id": "R-3899089",
          "ordinal": "2",
          "productId": "000UAN",
          "productType": "0",
          "productReference": "MD9044",
```

```

    "productName": "Zawór zwrotny paliwa 6mm - uniwersalny",
    "decimals": "0",
    "quantity": "3",
    "quantityInStock": "0",
    "stockStatus": "2",
    "stockInfo": "Pełna (od 2 dni)",
    "unit": "szt"
  },
  {
    "id": "R-3899090",
    "ordinal": "3",
    "productId": "0009VJ",
    "productType": "0",
    "productReference": "54624.GL",
    "productName": "Uszcz.miski olej.VAG 1.6-2.0 93- /GLASER/",
    "decimals": "0",
    "quantity": "4",
    "quantityInStock": "0",
    "stockStatus": "1",
    "stockInfo": "1 szt (1 - 2 dni)",
    "unit": "szt"
  },
  {
    "id": "R-3899091",
    "ordinal": "4",
    "productId": "141129",
    "productType": "0",
    "productReference": "11.721.800",
    "productName": "Świeca żarowa HONDA,RENAULT,ROVER 2.2DT /ISKRA/",
    "decimals": "0",
    "quantity": "4",
    "quantityInStock": "0",
    "stockStatus": "-1",
    "stockInfo": "towar niedostępny",
    "unit": "szt"
  }
]
}
}
}
}

```

doOrderClose

Close order

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	orderId	(string)	Order id
3	(?) collectionInPerson	(int)	Set delivery method to "personal collection"
4	(?) routeId	(string)	Identyfikator trasy

Return value:

1	orderId	(string)	Order id
---	---------	----------	----------

Error codes:

ERR_WRONG_ORDER_ID	Wrong order id
ERR_ORDER_ALREADY_CLOSED	Order is already closed
ERR_ORDER_LOCKED	Order is locked by other user
ERR_CANT_CLOSE_ORDER	Closing failed
ERR_COLLECTION_IN_PERSON_UNAVAILABLE	Collection in person is not available
ERR_CIP_ROUTE	Only one of the two fields may be used at the same time: collectionInPerson, routeId.
ERR_ROUTE_UNAVAILABLE	Choosing route is unavailable.
ERR_WRONG_ROUTE_ID	Incorrect route.
ERR_BLOCKED_CONFIRM_RECEIPT_DELIVERY	Delivery confirmation is required.
ERR_SALE_BLOCK	Order can't be closed. Sale locked
ERR_ISSUE_BLOCK	Order can't be closed. Vouchers locked
ERR_ISSUES_OVERDUE	Order can't be closed. Vouchers term exceeded
ERR_PAYMENT_OVERDUE	Order can't be closed. Payment term exceeded
ERR_CREDIT_LIMIT	Order can't be closed. Credit limit exceeded

Example:

```
Request:
{
  "doOrderClose": {
    "sessionId": "V5etQuyInDnHosbSzjy7iLdj3Us4GaEr5ILMt9_D",
    "orderId": "01520531",
    "collectionInPerson": "1"
  }
}

Response:
{
  "doOrderCloseResponse": { "orderId": "01520531" }
}
```

doOrderItemDelete

Delete order item

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	orderId	(string)	Order id
3	itemId	(string)	Order item id

Return:

1	itemId	(string)	Deleted item id
2	orderDeleted	(int)	Blank order deleted (1=yes, 0=no)

Error code:

ERR_WRONG_ORDER_ID	Wrong order id
ERR_ORDER_CLOSED	Order is closed
ERR_ORDER_LOCKED	Order is locked by other user
ERR_WRONG_ITEM_ID	Wrong item id
ERR_ITEM_IS_EXTERNAL	Can't delete external ordered item
ERR_ITEM_IS_DEPOSIT	Can't delete deposit
ERR_CANT_DELETE_ITEM	Deleting failed
ERR_SALE_BLOCK	Order can't be closed. Sale locked
ERR_ISSUE_BLOCK	Order can't be closed. Vouchers locked
ERR_ISSUES_OVERDUE	Order can't be closed. Vouchers term exceeded
ERR_PAYMENT_OVERDUE	Order can't be closed. Payment term exceeded
ERR_CREDIT_LIMIT	Order can't be closed. Credit limit exceeded

Example:

```
Request:
{
  "doOrderItemDelete": {
    "sessionId": "V5etQuyInDnHosbSzjy7iLdj3Us4GaEr5ILMt9_D",
    "orderId": "01520531",
    "itemId": "R-3975485"
  }
}

Response:
{
  "doOrderItemDeleteResponse": {
    "itemId": "R-3975485",
    "orderDeleted": "0"
  }
}
```

doOrderItemEdit

Edit order item

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	orderId	(string)	Order id
3	itemId	(string)	Order item id
4	itemInfo	(struct)	Item information
	1(?) quantity	(float)	Quantity ordered
	2(?) finalDocumentType	(int)	Final document type, "convert direction" (0-default, 1-invoice, 2-bill)
	3(?) customerReference	(string)	Customer reference
	4(?) packageType	(int)	Package type (0-own, 1-client, 2-none, default 0)
	5(?) description	(string)	Item description

Return:

1	item	(struct)	Item information
	1 id	(string)	Item id
	2 ordinal	(int)	Item position in document
	3 productId	(string)	Product id
	4 productType	(int)	Product type (0-article, 1-service)
	5 productReference	(string)	Product reference
	6 productName	(string)	Product name in user language
	7 decimals	(int)	Decimal places (0-3)
	8 tax	(float)	Tax rate
	9 quantity	(float)	Quantity ordered
	10 quantityBlocked	(float)	Quantity blocked in warehouse
	11 isExternal	(int)	Is item ordered in external warehouse (1/0)
	12 externalInfo	(string)	Information about delivery time
	13 isDeposit	(int)	Is item a deposit (1/0)
	14 unit	(string)	Unit
	15 priceExclTax	(float)	Price without tax
	16 valueExclTax	(float)	Value without tax
	17 priceInclTax	(float)	Price with tax
	18 valueInclTax	(float)	Value with tax
	19 valueTax	(float)	Tax value
	20 priceExclTaxLC	(float)	Price without tax in local currency
	21 valueExclTaxLC	(float)	Value without tax in local currency
	22 priceInclTaxLC	(float)	Price with tax in local currency
	23 valueInclTaxLC	(float)	Value with tax in local currency
	24 valueTaxLC	(float)	Tax value in local currency
	25 currency	(string)	Currency
	26 discountPercent	(float)	Discount %
	27 startingPriceExclTax	(float)	Price without tax before discount
	28 startingPriceInclTax	(float)	Price with tax before discount
	29 (?) priceCatExclTax	(float)	Catalogue price without tax in document currency
	30 (?) priceCatInclTax	(float)	Catalogue price with tax in document currency
	31 (?) finalDocumentType	(int)	Final document type, "conversion direction" (0-default, 1-invoice, 2-receipt)
	32 (?) description	(string)	Description for the item

Error codes:

ERR_WRONG_ORDER_ID	Wrong order id
ERR_ORDER_CLOSED	Order is already closed
ERR_ORDER_LOCKED	Order is locked by other user
ERR_WRONG_ITEM_ID	Wrong item id

ERR_ITEM_IS_EXTERNAL	Can't edit external ordered item
ERR_ITEM_HAS_DEPOSIT	Can't edit item with deposit
ERR_ITEM_IS_DEPOSIT	Can't edit deposit
ERR_WRONG_QUANTITY	Wrong quantity of ordered items
ERR_CANT_EDIT_ITEM	Edit failed
ERR_SALE_BLOCK	Order can't be closed. Sale locked
ERR_ISSUE_BLOCK	Order can't be closed. Vouchers locked
ERR_ISSUES_OVERDUE	Order can't be closed. Vouchers term exceeded
ERR_PAYMENT_OVERDUE	Order can't be closed. Payment term exceeded
ERR_CREDIT_LIMIT	Order can't be closed. Credit limit exceeded

Example:

```
Request:
{
  "doOrderItemEdit": {
    "sessionId": "v5etquyinDnHosbszjy7iLdj3Us4GaEr5ILMt9_D",
    "orderId": "01520531",
    "itemId": "R-3975485",
    "itemInfo": { "quantity": "2" }
  }
}

Response:
{
  "doOrderItemEditResponse": {
    "item": {
      "id": "R-3975485",
      "ordinal": "1",
      "productId": "105275",
      "productType": "0",
      "productReference": "123-456-789",
      "productName": "Filtr kabiny VAG",
      "decimals": "0",
      "tax": "23",
      "quantity": "2.000",
      "quantityBlocked": "1.000",
      "isExternal": "0",
      "isDeposit": "0",
      "unit": "szt",
      "priceExclTax": "50.00",
      "priceInclTax": "61.50",
      "valueExclTax": "100.00",
      "valueInclTax": "123.00",
      "valueTax": "23.00",
      "currency": "zł",
      "discountPercent": "50.0",
      "startingPriceExclTax": "100.00",
      "startingPriceInclTax": "123.00"
    }
  }
}
```

doOrderEdit

Edit order parameters

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	orderId	(string)	Order id
3	orderInfo	(struct)	Information about order
	1(?) comment	(string)	Comment
	2(?) dateDelivery	(date)	Date of realization [format: yyyy-mm-dd]
	3(?) additionalInfo	(string)	Additional information for the documentu

Return:

1	orderId	(string)	Order id
---	---------	----------	----------

Error codes:

ERR_WRONG_ORDER_ID	Wrong order id
--------------------	----------------

ERR_ORDER_ALREADY_CLOSED	Order is already closed
ERR_ORDER_LOCKED	Order is locked by other user
ERR_COMMENT_TOO_LONG	Comment is too long (max 30 chars)
ERR_CANT_CHANGE_ORDER	Order save failed
ERR_SALE_BLOCK	Order can't be closed. Sale locked
ERR_ISSUE_BLOCK	Order can't be closed. Vouchers locked
ERR_ISSUES_OVERDUE	Order can't be closed. Vouchers term exceeded
ERR_PAYMENT_OVERDUE	Order can't be closed. Payment term exceeded
ERR_CREDIT_LIMIT	Order can't be closed. Credit limit exceeded
ERR_ADDITIONAL_INFO_INACTIVE	Additional information about product unavailable

Example:

```

Request:
{
  "doOrderEdit": {
    "sessionId": "V5etQuyInDnHosbszjy7iLdj3Us4GaEr5ILMt9_D",
    "orderId": "01520531",
    "orderInfo": { "comment": "komentarz" }
  }
}

Response:
{
  "doOrderEditResponse": { "orderId": "01520531" }
}

```


doOrderDelete

Delete order

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	orderId	(string)	Order id
3(?)	deleteItems	(int)	Delete all items from odrder (1=yes, 0=no, default 0)

Return:

1	orderId	(string)	Order id
---	---------	----------	----------

Error code:

ERR_WRONG_ORDER_ID	Wrong order id
ERR_METHOD_BLOCKED	Method blocked
ERR_ORDER_ALREADY_CLOSED	Order is already closed
ERR_ORDER_LOCKED	Order is locked by other user
ERR_ORDER_STILL_HAS_ITEMS	Order still has some items
ERR_CANT_DELETE_ORDER	Order deleting error

Example:

```
Request:
{
  "doOrderDelete": {
    "sessionId": "v5etquyinDnHosbszjy7iLdj3Us4GaEr5ILMt9_D",
    "orderId": "01520531"
  }
}

Response:
{
  "doOrderDeleteResponse": { "orderId": "01520531" }
}
```

doOrderItemsMove

Move items between orders

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	srcOrderId	(string)	Source order id
3	srcItemList	(struct[])	List of items to move
	1 srcItem	(struct)	Item to move
	1 itemId	(string)	Item id
	2 quantity	(float)	Quantity to move
	2 srcItem ...		
4(?)	destOrderId	(string)	Destination order id
5	destOrderInfo	(struct)	Information about destination order
	1(?) dateDelivery	(date)	Date of realization [format: yyyy-mm-dd]

Return value:

1	status	(int)	1 - success, 0 – failed
2(*)	destOrderId	(string)	Destination order id

Error codes:

ERR_WRONG_ORDER_ID	Wrong source order id
ERR_WRONG_ORDERS	Wrong orders
ERR_WRONG_DEST_ORDER_ID	Wrong destination order id
ERR_WRONG_ITEM_ID	Wrong item id
ERR_ORDER_ALREADY_CLOSED	Order is already closed
ERR_ORDER_LOCKED	Order is locked by other user
ERR_DATEDELIVERY_WRONG_FORMAT	Wrong delivery date format
ERR_ORDER_CREATE	Error during creating new order

Uwagi:

* Information about destination order id will be returned if moving position will end succesfull.

Example:

```
request:
{
  "doOrderItemsMove": {
    "sessionId": "V5etQuyInDnHosbSzy7iLdj3Us4GaEr5ILMt9_D",
    "srcOrderId": "01520531",
    "destOrderId": "01520532"
  }
}

response:
{
  "doOrderItemsMoveResponse": {
    "status": "1"
  }
}
```

getOrderStatus

Get order status

Parameters:

1 sessionId	(string)	Session ID obtained via doLogin method
2 externalId	(string)	Order number

Return value:

1 status	(int)	0-unconfirmed, 1-in progress, 2-released, 3-sent, 4-cancelled, 5-status
2.statusDescription	(string)	Status description
3.packagesList	(struct[])	Package List
1. package	(struct)	Package Information
1. id	(string)	Package ID
2. tracking	(string)	Tracking number of the package
3. kep	(string)	Courier name
4. trackingUrl	(string)	Link to track your package
5. items	(struct[])	Package Item List
1. item	(struct)	Package position information
2. reference	(string)	Product symbol
3. quantity	(float)	Quantity of goods

Error codes:

ERR_UNKNOWN_EXTERNALID	Unknown order number
------------------------	----------------------

DOCUMENTS

getMyInvoices

Get list of sale documents of logged user

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	finalDocuments	(int)	Type of returned documents (0-vouchers, 1-invoices, 2-vouchers and invoices, 3-invoices corrections; default 2) ignored if given documentIdList
3(?)	documentIdList	(string[])	List of documents to return
1	id	(string)	Document id
2	id ...		
4(?)	filterOptions	(struct)	Document filters
1(?)	dateFrom	(date)	[format: yyyy-mm-dd] Begin of date range. By default, documents from last 7 days are returned (if parameter documentIdList was NOT given) or from any range (if parameter documentIdList was given)
2(?)	dateTo	(date)	[format: yyyy-mm-dd] End of date range (default: today)
3(?)	paymentStatus	(string)	Payment status (0-all, 1-only payed, 2-only not payed, 3-only not payed, out of date)
4(?)	closeStatus	(string)	Document status (0-all, 1-not confirmed, 2-confirmed)
5(?)	srcDocId	(string)	Id source document
6(?)	externalId	(string)	The customer order number
7(?)	isDoReturnNow	(int)	Return only documents which were created by doReturnNow (1-yes, 0-no)
8(?)	deliveryConfirmationStatus	(int)	Delivery confirmation status (1-confirmed, 2-unconfirmed)
9(?)	wmsStatus	(int)	Filter by document (0 – nohow, 1 – pass to delivery, 2 – to deliver)
10(?)	isReward	(int)	Filtering orders by tags “bring out reward” (1 – tag is set to yes, 0 – tag is set to no, no default filter)
11(?)	productIdList	(string[])	List of products of which at least one needs to be in the document
1	id	(string)	Product id
2	id ...		
5(?)	sortOptions	(struct)	Sorting options
1(?)	sortType	(int)	Sort type (1-by date of create (default))
2(?)	sortOrder	(int)	Sort order (1-ascending, 2-descending (default))
6(?)	getItems	(int)	Return information about items (1-yes, 0-no; default 0)
7(?)	getLoyaltyInfo	(int)	Return data of loyalty system (1 - yes, 0 - no; default 0)
8(?)	getTrackingNrList	(int)	Return data about list? (1 – return trackingNrList, 0 – don’t return)
9(?)	pageSize	(int)	Number of documents on page (min: 1, max: 50, default 10)
10(?)	pageNumber	(int)	Page number (default: 1)
11(?)	getEWDIdList	(int)	Whether to return information about electronic IDT attachments (1 - yes, 0 - no; default 0)

Return value:

1	invoiceList	(struct[])	List of documents
1	invoice	(struct)	Information about document
1	id	(string)	Document id
2	number	(string)	Document number

3	justNumber	(string)	Document number without postscript
4	externalId	(string)	Number of external order
5	type	(string)	Document type: WZ-invoice, DD-goods issue note, ZW-invoice correction
6	warehouseId	(string)	Warehouse id
7	date	(date)	[format: yyyy-mm-dd] Document date
8	isClosed	(int)	Document status (1-confirmed, 0- not confirmed)
9	recipientId	(string)	Recipient id
10	recipientName	(string)	Recipient name
11	recipientShortName	(string)	Short name of recipient
12	dateSale	(date)	[format: yyyy-mm-dd] Sale date
13	isFinalDocument	(int)	Type of document: 1-invoice, 0-voucher
14	isDoReturnNow	(int)	Return only documents which were created by doReturnNow (1=yes, 0=no)
15	comment	(string)	Comment
16	currency	(string)	Currency
17	localCurrency	(string)	Local currency
18	currencyRate	(float)	Currency rate
19	exportType	(int)	Invoice type: 0 - country, 1 - export, 2 – country in currency
20	isTaxLC	(int)	Is tax in local currency (1) or document currency (0)
21	IsValueLC	(int)	Is value in local currency (1) or document currency (0)
22	valueExclTaxLC	(float)	Value without tax in local currency
23	valueInclTaxLC	(float)	Value with tax in local currency
24	valueTaxLC	(float)	Tax value in local currency
25	valueExclTax	(float)	Value without tax
26	valueInclTax	(float)	Value with tax
27	valueTax	(float)	Tax value
28	isEU	(int)	EU document (1=yes, 0=no)
29	isReExport	(int)	Is document re-exported (1=yes, 0=no)
30	paymentType	(string)	Payment type: 1-none, 2-cash, 3-cash in term, 4-check, 5-bank wire, 6-cash/bank wire, 7-COD, 8-credit card, 9-installment
31	paymentDays	(int)	Payment term (in days)
32	isEDocument	(int)	Electronic version of document exist (1=yes, 0=no)
33	eDocumentStatus	(int)	Electronic version status: (0-no information, 1-created, 2-sent)
34	isEwdtBlocked	(int)	Is the EWDT e-document blocked for downloading (not created or no delivery confirmation) (0-no, 1=yes)
35	isPrinted	(int)	Printed document (1 – yes, 0 – no)
36	wmsStatus	(int)	Document WMS status (0-none, 1-given to release, 2-released)
37	paymentStatus	(int)	Payment status (0-unknown status, 1-settled, 2-not settled, 3-not settled, out of date)
38	loyaltyPoints	(int)	Number of points earned in loyalty system (only if getLoyaltyInfo = 1)
39	loyaltyRewardPoints	(int)	Number of points that will be decreased after given reward (returned only if getLoyaltyInfo = 1)
40	isDeliveryConfirmed	(int)	Is delivery confirmed (1 - yes, 0 - no)
41	isReward	(int)	Content of tag “bring out reward” (0 – no, 1 - yes)
42	routeId	(string)	Route id
43	trackingNrList	(struct[])	List of tracking numbers
	1 item	(string)	Tracking number
	2 item ...		
44	eWDTIdList	(struct[])	List of file ids with WDT printout
	1 item	(string)	WDT print file id
	2 item ...		
45	finalDocumentInfo	(struct)	Target Document Information

1	type	(string)	Target document type
2	documentNumber	(string)	Target document number
3	date	(string)	Target document date
46	tmsDeliveryConfirmed	(int)	Has the document content been delivered (1 - yes, 0 - no)
47	itemsCount	(int)	Number of items in document
48	items	(struct[])	List of document items (only if <code>getItems = 1</code>)
1	item	(struct)	Information about item
1	id	(string)	Item id
2	ordinal	(int)	Item position in document
3	productId	(string)	Product id
4	productType	(int)	Product type (0-article, 1-service)
5	productReference	(string)	Product reference
6	productName	(string)	Product name in user language
7	country	(string)	Product country
8	decimals	(int)	Decimal places (0-3)
9	tax	(float)	Tax rate
10	quantity	(float)	Quantity (for invoice corrections: quantity after correction)
11	unit	(string)	Unit
12	priceExclTax	(float)	Price without tax
13	valueExclTax	(float)	Value without tax
14	priceInclTax	(float)	Price with tax
15	valueInclTax	(float)	Value with tax
16	valueTax	(float)	Tax value
17	priceExclTaxLC	(float)	Price without tax in local currency
18	valueExclTaxLC	(float)	Value without tax in local currency
19	priceInclTaxLC	(float)	Price with tax in local currency
20	valueInclTaxLC	(float)	Value with tax in local currency
21	valueTaxLC	(float)	Tax value in local currency
22	currency	(string)	Currency
23	discountPercent	(float)	Discount %
24	startingPriceExclTax	(float)	Price without tax before discount
25	startingPriceInclTax	(float)	Price with tax before discount
26	(?) priceCatExclTax	(float)	Catalogue price without tax in document currency
27	(?) priceCatInclTax	(float)	Catalogue price with tax in document currency
28	loyaltyPoints	(int)	Number of points earned in loyalty system (only if <code>getLoyaltyInfo = 1</code>)
29	loyaltyRewardPoints	(int)	Number of points that will be decreased after given reward (returned only if <code>getLoyaltyInfo = 1</code>)
30	weight_net	(float)	Net weight
31	weight_gross	(float)	Gross weight
32	customs_code	(string)	Customs code
33	(?) correctedItem	(struct)	Information about corrected item (for invoice corrections)
1	quantity	(float)	Quantity before correction
2	priceExclTax	(float)	Price without tax
3	valueExclTax	(float)	Value without tax
4	priceInclTax	(float)	Price with tax
5	valueInclTax	(float)	Value with tax
6	valueTax	(float)	Tax value
7	priceExclTaxLC	(float)	Price without tax in local currency
8	valueExclTaxLC	(float)	Value without tax in local currency
9	priceInclTaxLC	(float)	Price with tax in local currency
10	valueInclTaxLC	(float)	Value with tax in local currency
11	valueTaxLC	(float)	Tax value in local currency
12	discountPercent	(float)	Discount %
13	startingPriceExclTax	(float)	Price without tax before discount

14	startingPriceInclTax	(float)	Price with tax before discount
15	srcDocumentId	(string)	Source document id of the corrected item
16	srcDocumentNumber	(string)	Source document number of the corrected item
34**	pieceList	(struct[])	List information about piece
1	piece	(struct)	Piece information
1	imei	(string)	IMEI number
2	sn	(string)	Serial number
2	piece ...		
35.	externalItemId	(string)	External order number
2	item ...		
2	invoice ...		
2	invoicesCount	(int)	Number of documents that meets the criteria value can be approximated

Error codes:

ERR_WRONG_FINALDOCUMENTS_TYPE	Wrong document type
ERR_NOT_ALLOWED_FINALDOCUMENTS_TYPE	Forbidden document type

Comments:

Metod returns documents from 7 days, unless dateFrom, dateTo (dates range) or documentIdList (list of documents IDs) is passed.

** Information about piece returns only for products with specified parameters.

Example:

```

Request:
{
  "getMyInvoices": {
    "sessionId": "2BZY8zSx0D00Umm0y1FKSZjMsgDG564YxTMvju_D",
    "documentIdList": {
      "id": [
        "00001277",
        "23456789"
      ]
    },
    "filterOptions": {
      "dateFrom": "2013-03-01",
      "dateTo": "2013-03-07",
      "webCatUserId": "012932"
    },
    "sortOptions": {
      "sortType": "1",
      "sortOrder": "1"
    },
    "pageSize": "50",
    "pageNumber": "1",
    "getItems": "1"
  }
}

Response:
{
  "getMyInvoicesResponse": {
    "invoiceList": {
      "invoice": [
        {
          "id": "00001277",
          "number": "FV/1/01/CE/2009",
          "warehouseId": "CE",
          "dateIssue": "2013-03-04",
          "dateSale": "2013-03-04",
          "isFinalDocument": "1",
          "currency": "zł",
          "localCurrency": "zł",
          "currencyRate": "0",
          "exportType": "0",
          "valueExclTax": "1040.44",
          "valueInclTax": "1269.34",
          "valueTax": "228.90",
          "isEDocument": "0",
          "itemsCount": "2",
          "items": {
            "item": [
              {
                "id": "R-8239",
                "ordinal": "1",
                "productId": "003WUL",
                "productType": "0",
                "productReference": "0-092-s40-280",
                "productName": "Akumulator 95AH/830A /BOSCH ",
                "decimals": "0",
                "tax": "22",
                "quantity": "1.000",
                "unit": "szt",
                "priceExclTax": "819.67",
                "valueExclTax": "819.67",
                "priceInclTax": "1000.00",
                "valueInclTax": "1000.00",
                "valueTax": "180.33",
                "currency": "zł",
                "discountPercent": "0.0",
                "startingPriceExclTax": "819.67",
                "startingPriceInclTax": "1000.00"
              },
              item2...
            ]
          }
        },
        invoice2...
      ]
    }
  }
}

```


getEDocument

Get digital version of document (PDF)

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	documentId	(string)	Document id

Return:

1	eDocument	(string)	PDF data of document (base64-encoded)
---	-----------	----------	---------------------------------------

Error codes:

ERR_WRONG_DOCUMENT_ID	Wrong document id
ERR_NO_EDOCUMENT	Digital version of document is not available
ERR_DELIVERY_NOT_CONFIRMED	Delivery not confirmed

Example:

```
Request:
{
  "getEDocument": {
    "sessionId": "2BZY8zSx0D00umM0y1FKSZjMsgDG564YxTMvju_D",
    "documentId": "00001277"
  }
}

Response:
{
  "getEDocumentResponse": {
    "eDocument": "PDF zakodowany base64"
  }
}
```

doDocumentPrint

Generate PDF printout of document

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	documentId	(string)	Document id

Return:

1	pdfData	(string)	Printout of document
---	---------	----------	----------------------

Error codes:

ERR_WRONG_DOCUMENT_ID	Wrong document id
ERR_DOCUMENTPRINT_INACTIVE	Printout is unavailable
ERR_EDOCUMENT_EXISTS	This document exists in electronic form

doSetDeliveryConfirmation

Delivery confirmation for WZ document

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	documentId	(string)	Document id
3	clientIP	(string)	Client IP address (*)
4	clientLogin	(string)	Customer login in external system (*)

Return value:

1	status	(int)	1 – confirmed documents
---	--------	-------	-------------------------

Error codes:

ERR_WRONG_DOCUMENT_ID	Wrong document id
ERR_DOCUMENT_ALREADY_CONFIRMED	Document already confirmed
ERR_DOCUMENT_NOT_DELIVERED	No parcels delivered by TMS
ERR_CLIENT_IP_NOT_AVAILABLE	The client IP address cannot be provided in this connection.
ERR_CLIENT_LOGIN_NOT_AVAILABLE	Cannot provide client login on this connection

Additional information:

The clientIP parameter is only available in selected connection types.

The clientLogin parameter is only available in selected connection types.

getDocumentWDTFileUrl

Displaying the WDT printout for downloading and returning the URL address

Parameters:

1. fileId (string) File ID
2. documentId (string) Document ID

Return value:

1. url (string) Adres URL pliku

Error codes:

- | | |
|---------------------------|---------------------------------------|
| ERR_WRONG_FILE_ID | Invalid file id |
| ERR_SERVER_NOT_CONFIGURED | The server is not properly configured |

Additional informations:

The clientIP parameter is only available in selected connection types.

The clientLogin parameter is only available in selected connection types.

ROUTES

getMyRoutes

Get list of users routes

Parameters:

1 sessionId (string) Session id obtained with doLogin method

Return:

1 warehouseList	(struct[])	List of routes for available warehouses
1 warehouse	(struct)	List of routes in a single warehouse
1 warehouseId	(string)	Warehouse id
2 routeList	(struct[])	List of routes
1 route	(struct)	Information about route
1 id	(string)	Route id
2 description	(string)	Route description
3 isDefault	(int)	Default route in warehouse (deprecated, always returns: 0)
4 transportType	(int)	0-default, 1-own, 2-external
2 route ...		
3 warehouseId ...		
2 warehouse ...		

Error codes:

none

Example:

```
Request:
{
  "getMyRoutes": { "sessionId": "V5etQuyInDnHosbszjy7iLdj3Us4GaEr5ILMt9_D" }
}

Response:
{
  "getMyRoutesResponse": {
    "warehouseList": {
      "warehouse": {
        "warehouseId": "01",
        "routeList": {
          "route": [
            {
              "id": "314A",
              "description": "trasa pi ( 12:00 - 13:00 ) Def.",
              "isDefault": "1"
            },
            {
              "id": "111A",
              "description": "Odbiór osobisty",
              "isDefault": "0"
            }
          ]
        }
      }
    }
  }
}
```

LOGISTICS

getMyPackages

Get list of users packages

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2(?)	packageIdList	(string[])	List of packages to return
1	id	(string)	Package Id
2	id ...		
3(?)	filterOptions	(struct)	Packages filter
1(?)	dateFrom	(date)	[format: yyyy-mm-dd] Begin of date range. By default, packages from last 7 days are returned (if parameter packageIdList was not given) or from any range (if parameter packageIdList was given)
2(?)	dateTo	(date)	[format: yyyy-mm-dd] End of date range (default:today)
3(?)	documentIdList	(string[])	Documents Id related with packages
1	id	(string)	Document Id
2	id ...		
4(?)	routeId	(string)	Route Id
4(?)	sortOptions	(struct)	Sorting options
1(?)	sortType	(int)	Sort type (1 - by date of create (default))
2(?)	sortOrder	(int)	Sort order (1-ascending, 2-descending (default))
5(?)	pageSize	(int)	Number of packages on page (min: 1, max: 50, default 10)
6(?)	pageNumber	(int)	Page number(default: 1)
7(?)	getItems	(int)	Return information about items (default: 0 = No, 1 = Yes)
8(?)	getTrackingUrl	(int)	Return tracking urls for packages (default: 0 = No, 1 = Yes)

Return:

1	packageList	(struct[])	List of packages
1	package	(struct)	Package information
1	id	(string)	Package Id
2	date	(date)	[format: yyyy-mm-dd] Data of create
3	routeId	(string)	Route id
4	srcDocId	(string)	Package's number document. If it's empty, package contain items from a many documents
5	weight	(int)	Package weight in kg
6	x	(int)	Size 1
7	y	(int)	Size 2
8	z	(int)	Size 3
9	isClosed	(int)	Is package closed
10	urlTracking	(string)	URL to the package history in TMS
11	items	(struct[])	List of package item (only if getItems = 1)
1	item	(struct)	Item
1	productId	(string)	Product Id
2	ilkod	(string)	Product ILkod
3	itemSrcDocId	(string)	Source document Id
4	quantity	(int)	Quantity
2	item ...		
2	package ...		

Error codes:

ERR_INVALID_WAREHOUSE
ERR_WAREHOUSE_ACCESS_DENIED

Invalid warehouse
No access to the selected warehouse

PAYMENTS

getMyPayments

Get list of payments

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	paymentStatus	(int)	Payment status (0-unpaid, 1-paid, default: 0)
3(?)	filterOptions	(struct)	Payments filter
1(?)	dateFrom	(date)	[format: yyyy-mm-dd] Begin of date range.
2(?)	dateTo	(date)	[format: yyyy-mm-dd] End of date range.
4(?)	sortOptions	(struct)	Informacje o sposobie sortowania
1(?)	sortType	(int)	Sort type (1 - by date of create (default)), 2 – by date of payment)
2(?)	sortOrder	(int)	Sort order (1-ascending, 2-descending (default))
5(?)	pageSize	(int)	Number of payment on page (min: 1, max: 50, default 10)
6(?)	pageNumber	(int)	Page number (default: 1)

Return:

1	paymentList	(struct[])	List of payments
1	payment	(struct)	Payment information
1	id	(string)	Payment Id
2	createDate	(date)	[format: yyyy-mm-dd] Date of create
3	paymentDate	(date)	[format: yyyy-mm-dd] Date of payment
4	status	(int)	Payment status (0-unpaid, 1-paid)
5	documentNumber	(string)	Document Id
6	daysLeft	(int)	Number of days to payment's date (negative means number days after to payment's date)
7	warehouseId	(string)	Warehouse Id
8	valueTotal	(float)	Total value
9	valueToPay	(float)	Unpaid value
10	currency	(string)	Currency
11	isRelDocument	(string)	Payment is related with document (0 = No,1 = Yes)
12	relDocumentId	(string)	Related document id
13	relDocumentType	(string)	Related document type
14	payItemList	(struct[])	Payment's item
1	payItem	(struct[])	Payment's item information
1	amount	(int)	Amount
2	date	(date)	Date
3	description	(string)	Description
2	item ...		
2	payment ...		
2	paymentsCount	(int)	Number of payments that meets the criteria value can be approximated

Error code

none

Example:

```
Request:
{
  "getMyPayments": {
    "sessionId": "2BZY8zSx0D00UmM0y1FKSZjMsgDG564YxTMvju_D",
    "paymentStatus": "0",
    "filterOptions": {
      "dateFrom": "2013-03-01",
      "dateTo": "2013-03-07"
    },
    "sortOptions": {
      "sortType": "1",
      "sortOrder": "1"
    },
    "pageSize": "50",
    "pageNumber": "1"
  }
}
```

```
Response:
{
  "getMyPaymentsResponse": {
    "paymentList": {
      "payment": [
        {
          "id": "00001277",
          "createDate": "2014-08-04",
          "paymentDate": "2014-08-11",
          "status": "0",
          "documentNumber": "F/1024/CE/2014",
          "daysLeft": "-3",
          "warehouseId": "CE",
          "valueTotal": "123.45",
          "valueToPay": "123.45",
          "currency": "PLN",
          "isRelDocument": "1",
          "relDocumentId": "00001234",
          "relDocumentType": "wz"
        },
        "....."
      ]
    }
  }
}
```


RETURNS

getMyReturns

Get list of users returns

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2(?)	returnIdList	(string[])	List of returns
1	id	(string)	Return id
2	id ...		
3(?)	filterOptions	(struct)	Returns filter
1(?)	state	(int)	State return: 0-waiting, 1-completed, (default: 0)
2(?)	packageName	(int)	Package number
3(?)	dateFrom	(date)	[format: yyyy-mm-dd] The starting date of the range. The dateTo parameter must also be specified.
4(?)	dateTo	(date)	[format: yyyy-mm-dd] The end date of the range. The dateFrom parameter must also be specified.

Return:

1	returnsCount	(int)	Count of returns. This value can be approximate.
2	returnList	(struct[])	List of returns
1	return	(struct)	Information about return
1	id	(string)	Return id
2	documentNumber	(string)	Number of the linked sales document
3	dateCreate	(date)	Date of create return
4	state	(int)	Return state: 0-waiting, 1-completed
5	warehouseId	(string)	Warehouse id
6	productId	(string)	Product id
7	productCode	(string)	Product reference
8	productName	(string)	Product name in user language
9	quantity	(float)	Quantity
10	ildec	(int)	Decimal places
11	status	(int)	Return status: 0-added, 1-sent, 2-package received, 3-accepted, 4-rejected
12	packageName	(int)	Package number
13 (?)	packageBarcode	(string)	EAN-13 Barcode
14	dateReceived	(date)	Date of received package
15	info	(text)	Information about return
16	canEdit	(int)	Is edition available
2	return ...		

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_INVALID_DATERANGE	Invalid date range
ERR_INVALID_RETURN_ID	Invalid return id
ERR_INVALID_RETURN_STATE	Incorrect return state
ERR_INVALID_RETURN_STATUS	Incorrect return status

doReturnCreatePackage

Create number package for list of returns.

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	returnIdList	(string[])	List of returns
	1 id	(string)	Return id
	2 id ...		
3	overwrite	(int)	Generate package number for returns again. (1=yes, 0=no), (default:0)

Return:

1	packageNumber	(int)	Package number
---	---------------	-------	----------------

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_MULTI_WAREHOUS	All returns must come from one warehouse
ERR_PACKAGE_ALREADY_EXIST	Package has already generated
ERR_EMPTY_RETURN_ID_LIST	Empty list of returns id
ERR_INVALID_RETURN_ID	Invalid return id
ERR_INVALID_RETURN_STATE	Invalid return state
ERR_INVALID_RETURN_STATUS	Invalid return status
ERR_INTERNAL	Generating labels

doDeleteReturn

Delete registered return

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	returnId	(string)	Return id
3	deleteIfSent	(int)	Delete return, which was sent (1=yes, 0=no), (default:0)

Return:

1	status	(int)	Status: 1-succeed
---	--------	-------	-------------------

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_INVALID_RETURN_ID	Invalid return id
ERR_CANNOT_DELETE	Can't delete return (contact administrator)
ERR_WAREHOUSE_ACCESS_DENIED	No access to chosen warehouse
ERR_RETURN_SENT	The return was already sent

doEditReturn

Edit Return

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	returnId	(string)	Return id
3	returnInfo	(struct)	Information about return
	1(?) info	(text)	Information about return

Return:

1	status	(int)	Status: 1-succeed
---	--------	-------	-------------------

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_INVALID_RETURN_ID	Invalid return id
ERR_RETURN_STATUS	Can't edit return (status)
ERR_INVALID_RETURN_INFO	Invalid information about return

doCreateReturn

Create new return

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	productId	(string)	Product id
3	warehouseId	(string)	Warehouse id
4	quantity	(float)	Quantity to return
5	(?) srcDocumentId	(string)	Source document id

Return:

1	returnId	(string)	Return id
---	----------	----------	-----------

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_INVALID_WAREHOUSE	Invalid warehouse id
ERR_WAREHOUSE_ACCESS_DENIED	No access to chosen warehouse
ERR_WRONG_PRODUCT_ID	Wrong product id
ERR_CANNOT_RETURN	Can't return product <reason>
ERR_QUANTITY_TOO_MUCH	Too much quantity (max: %%)
ERR_QUANTITY	Incorrect quantity (none or <= 0)
ERR_QUANTITY_FORMAT	Invalid format quantity (number of decimal places)
ERR_WRONG_SOURCE_DOCUMENT	Incorrect source document
ERR_WRONG_SOURCE_DOCUMENT_WAREHOUSE	Incorrect warehouse of the source document

getReturnQuantityAvailable

Get available quantity product possible to return

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	productId	(string)	Product id
3(?)	warehouseId	(string)	Warehouse id
4(?)	detailedQuantity	(int)	Detailed quantities grouped by document (1=yes, 0=no). Default: 0

Return:

1	quantity	(float)	Quantity possible to return
2	limitUsed	(float)	Used percent for limit granted returns
3	returnInfo	(string)	Additional information about return
4 (?)	documentList	(struct[])	Detailed quantities grouped by document
1	document	(struct)	
1	id	(string)	Document id
2	numer	(string)	Document number
3	quantity	(float)	Quantity on the document which is possible to be returned
4	quantityAvailable	(float)	Quantity on the document which is available to be returned

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_INVALID_WAREHOUSE	Invalid warehouse id
ERR_WAREHOUSE_ACCESS_DENIED	No access to chosen warehouse
ERR_WRONG_PRODUCT_ID	Wrong product id
ERR_CANNOT_RETURN	Can't return product <reason>

getReturnNowProductInfo

Additional information related to product quick return

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	(?) warehouseId	(string)	Warehouse id (by default it is the customer's default warehouse)
3	product	(struct)	Product information
1	(?)* id	(string)	Product id
2	(?)* reference	(string)	Product symbol
3	(?)* tecidd	(string)	TecDoc supplier id
4	(?)* tecnum	(string)	TecDoc product id
5	(?)* ILkod	(string)	Ilkod product id
6	(?)* motonet	(string)	Motonet number

Return:

1	isReturnBlocked	(int)	Is product return blocked (1 – yes, 0 – no)
2	isMainWarehouseAcceptBlocked	(int)	Is product accept blocked by the main warehouse (1 – yes, 0 – no)
3	returnSectorInfo	(struct)	Sector info
1	id	(string)	sector id
2	name	(string)	sector name

Error codes:

ERR_NO_WAREHOUSE	No available warehouses to order
ERR_WAREHOUSE_UNAVAILABLE	In warehouse %% user can't make orders
ERR_METHOD_BLOCKED	Method is blocked
ERR_WRONG_PRODUCT_ID	Wrong product id

Request example:

```
{
  "getReturnNowProductInfo": {
    "sessionId": "e1cd41ENipXxmN19N1uBasurwOb6FY20010825_D",
    "warehouseId": "01",
    "product": {
      "id": "0003AZ"
    }
  }
}
```

Response example:

```
{
  "getReturnNowProductInfoResponse": {
    "isMainWarehouseAcceptBlocked": "0",
    "isReturnBlocked": "0",
    "returnSectorInfo": {
      "id": "0001",
      "name": "sector 1"
    }
  }
}
```

doReturnClearPackage

Delete information about package from list of returns

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	returnIdList	(string[])	List of returns
1	id	(string)	Return id
2	id ...		

Return:

1	status	(int)	Status: 1-succeed
---	--------	-------	-------------------

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_INVALID_RETURN_ID	Invalid return id
ERR_RETURNS_CLOSED	The returns was already closed
ERR_RETURNS_SENT	The returns was already sent

doReturnGetPackageLabel

Get package label for return

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	packageNumber	(int)	Package number

Return:

1	labelPDFData	(string)	PDF print label (base64-encoded)
---	--------------	----------	----------------------------------

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_WRONG_PARAM	Missing or incorrect parameter (parameter)
ERR_PACKAGE_NOT_FOUND	Package number was not found
ERR_INVALID_RETURN_STATE	Invalid return state
ERR_INVALID_RETURN_STATUS	Invalid return status
ERR_INTERNAL	Generating labels

doReturnSendPackage

Mark return from package as sent

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	packageNumber	(int)	Package Number

Return:

1	status	(int)	Status: 1-succeed
---	--------	-------	-------------------

Error codes:

ERR_RETURNS_INACTIVE	Returns are not active
ERR_PACKAGE_NOT_FOUND	Package number was not found
ERR_INVALID_RETURN_STATE	Invalid return state
ERR_INVALID_RETURN_STATUS	Invalid return status
ERR_INTERNAL	Generating labels

doReturnNow

Quick return

Parameters:

1	sessionId	(string)	Session id obtained with doLogin method
2	(?)warehouseId	(string)	Warehouse id (def.: default client warehouse)
3	product	(struct)	Product information
	1 (?)*id	(string)	Product id
	2 (?)*reference	(string)	Product reference
	3 (?)*tecidd	(string)	TecDoc supplier id
	4 (?)*tecnum	(string)	TecDoc product id
	5 (?)*Ilkod	(string)	ILKod product id
	6 (?)*motonet	(string)	Motonet number
4	quantity	(float)	Return quantity
5	(?)*boxId	(string)	Box id
6	(?)*returnAsManyAsPossible	(int)	Return as many as possible (1=yes, 0=no, def. 0)
7	(?) collectivePackageId	(string)	Id of the original bulk packaging used

Return:

1	quantityReturned	(float)	Quantity returned
2	documentInfo	(struct)	Document info
	1 id	(string)	Document id
	2 dateCreate	(date)	Create date
	3 number	(string)	Document number
	4 isSplitPayment	(int)	Does document have the split payment marker (1=yes, 0=no)
3	correctedDocumentInfo	(struct)	Corrected document info
	1 id	(string)	Corrected document id
	2 dateCreate	(date)	Create date
	3 number	(string)	Corrected document number
4	itemInfo	(struct)	Item info
	1 id	(string)	Item id
	2 productId	(string)	Product id
	3 productReference	(string)	Product reference
	4 Ilkod	(string)	ILKod product id
	5 productName	(string)	Product name
	6 decimals	(int)	Decimal places(0-3)
	7 tax	(float)	Tax rate
	8 quantity	(float)	Quantity
	9 discountPercent	(float)	Discount percent
	10 startingPriceExclTax	(float)	Price without tax in document currency
	11 startingPriceInclTax	(float)	Price with tax in document currency
	12 correctedItem	(struct)	Corrected item info
	1 quantity	(float)	Quantity
	2 discountPercent	(float)	Discount percent
	3 startingPriceExclTax	(float)	Price without tax in document currency
	4 startingPriceInclTax	(float)	Price with tax in document currency
5	itemDepositInfo	(struct)	Information about the deposit of the ZW item (if the product has a deposit)
	1 id	(string)	ZW document item identifier
	2 productId	(string)	Item ID
	3 productReference	(string)	Product symbol
	4 ILkod	(string)	Symbol ILcode
	5 productName	(string)	Product name
	6 decimals	(int)	Number of decimal places
	7 tax	(float)	Tax rate
	8 quantity	(float)	Quantity (MUST BE)
	9 discountPercent	(float)	Discount percentage

10	startingPriceExclTax	(float)	Net price before discount in the document currency
11	startingPriceInclTax	(float)	Gross price before discount in the document currency
12	correctedItem	(struct)	Information about old position values
1	quantity	(float)	Quantity (WAS)
2	discountPercent	(float)	Discount percentage
3	startingPriceExclTax	(float)	Net price before discount in the document currency
4	startingPriceInclTax	(float)	Gross price before discount in the document currency

Error codes:

ERR_NO_WAREHOUSE	No available warehouses to order
ERR_WAREHOUSE_UNAVAILABLE	In warehouse %% user can't make orders
ERR_METHOD_BLOCKED	Method blocked
ERR_WRONG_PRODUCT_ID	Invalid product id
ERR_CANNOT_RETURN	Can't return product <reason>
ERR_QUANTITY_TOO_MUCH	Too much quantity (max: %%)
ERR_QUANTITY	Incorrect quantity (none or <= 0)
ERR_QUANTITY_FORMAT	Invalid format quantity (number of decimal places)
ERR_INVALID_BOX_ID	Invalid box id
ERR_BOX_STATUS	Invalid box status
ERR_WRONG_RETURN_SECTOR	Invalid sector
ERR_WRONG_BOXID	Nieprawidłowe id skrzynki
ERR_WRONG_COLLECTIVEPACKAGEID	Nieprawidłowe id opakowania zbiorczego

Comments:

- * One of these is required: "id" or "reference" or both "tecidd" and "tecnum" or "motonet".
- * In case of sending TecDoc or Motonet data - information about first fitting product is returned.

DROPSHIPPING

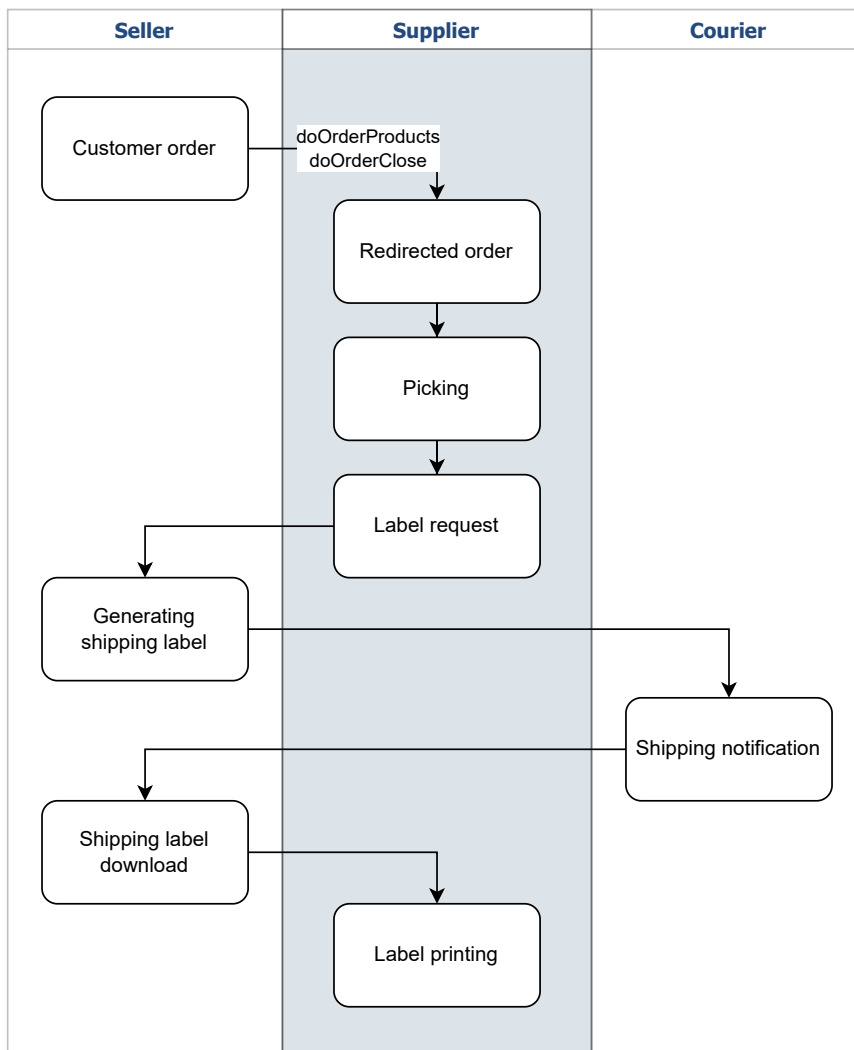
Dropshipping is a logistics model in which the seller accepts orders but does not store the products or ship them directly. Instead, the orders are forwarded to the supplier, who prepares the shipment and sends it to the final recipient. Depending on the chosen scenario of this process, the seller may be responsible for reporting the shipment to the courier company (then the supplier communicates with the seller to download the label for the package - scenario A) or directly the supplier (scenario B).

Scenario A

In this arrangement, the vendor must provide the option of downloading labels for packages prepared by the supplier.

In the query, the supplier can provide the following information about the package: order number, dimensions, weight, content. In response, the vendor must provide a label in a PDF file, which will be placed on the package. Depending on technical conditions, there may be technical restrictions regarding the maximum size of the label. For details, please contact the supplier directly.

Note! This form of process requires each time the supplier's system to be adapted to the vendor's IT solutions, so its implementation may be more time-consuming.



Example of a doOrderProducts query:

```
{  "doOrderProducts": {
    "forceNewOrder": 1,
    "onlyFoundItems": 1,
    "newOrderInfo": {
      "externalId": "*****",
      "routeId": "*****"
    },
    "productOrderList": {
      "productOrder": [{
        "tecidd": "350",
        "tecnum": "ADV184326",
        "quantity": 2
      }
    ]
  },
  "sessionId": "*****"
}
```

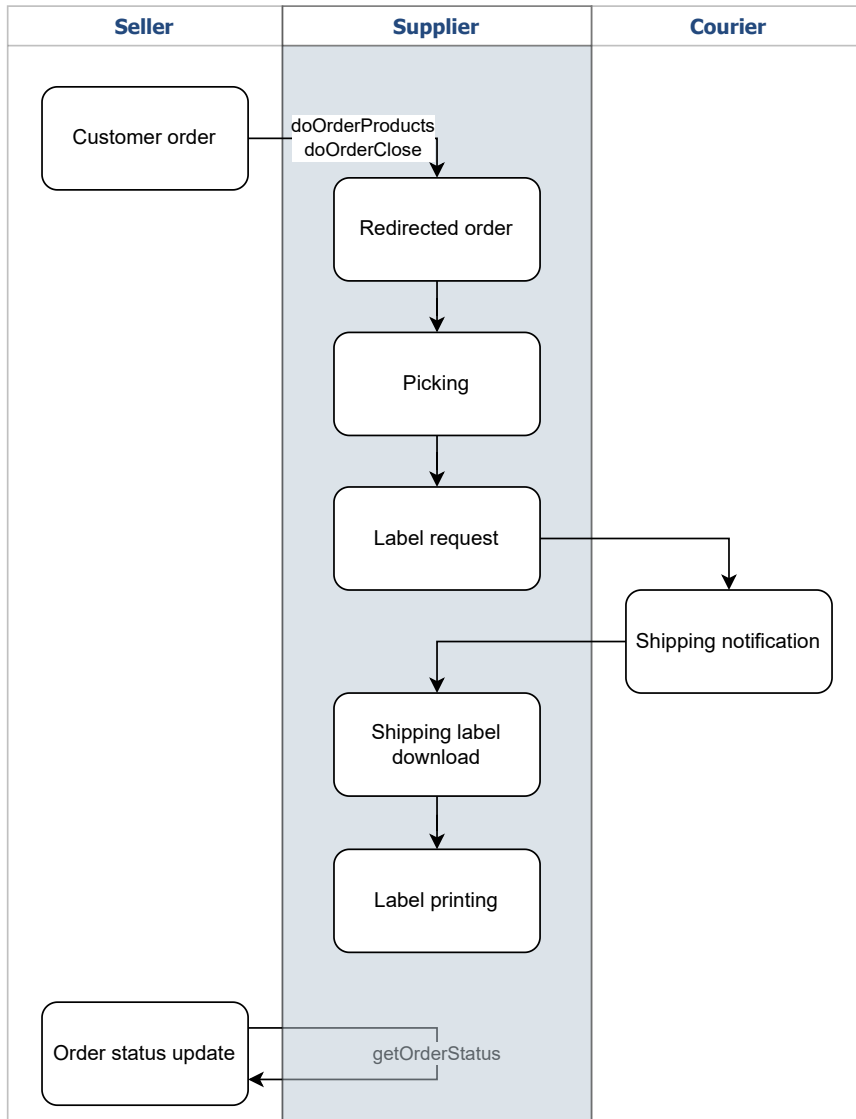
Scenario B

In this arrangement, the supplier is responsible for sending the package to the recipient. The recipient's data must be transferred together with the order in the doOrderProducts method. Additionally, depending on the arrangements between the seller and the supplier, the shipment can be reported to the courier with the supplier's data (then it must be settled independently with the seller) or the seller (the data must be entered into the supplier's system at the connection configuration stage). In each arrangement, when ordering the goods, the seller should indicate the route - i.e. the form of delivery to the recipient - together with the order (doOrderProducts method).

Note! Assuming that we are using couriers already integrated in the cairo.WMS system, implementing communication in this form does not require modification of the system, only its appropriate configuration.

List of methods suggested for use in Dropshipping:

- doLogin
- getProductsInfo
- doOrderProducts
- doOrderClose
- getOrderStatus
- getMyInvoices
- getMyRoutes



Example of a doOrderProducts query for courier delivery:

```
{  "doOrderProducts": {
    "forceNewOrder": 1,
    "onlyFoundItems": 1,
    "newOrderInfo": {
      "deliveryAddress": {
        "name": "*****",
        "street": "*****",
        "postcode": "*****",
        "city": "*****",
        "country": "*",
        "phone": "*****",
        "email": "*****"
      },
      "externalId": "*****",
      "routeId": "*****"
    },
    "productOrderList": {
      "productOrder": [{
        "tecidd": "350",
        "tecnum": "ADV184326",
        "quantity": 2
      }
    ]
  },
  "sessionId": "*****"
}
```


Example of a doOrderProducts query for shipping to a collection point:

```
{
  "doOrderProducts": {
    "forceNewOrder": 1,
    "newOrderInfo": {
      "pickupPointAddress": {
        "name": "*****",
        "street": "*****",
        "postcode": "*****",
        "city": "*****",
        "country": "*****",
        "phone": "*****",
        "email": "*****",
        "pickupPointId": "*****"
      },
      "externalId": "208755449",
      "routeId": "*****"
    },
    "productOrderList": {
      "productOrder": [
        {
          "reference": "VK22 DENSO",
          "quantity": 4
        }
      ]
    },
    "sessionId": "*****"
  }
}
```