Orders API Overview

This topic provides an overview of the orders and order drafts REST API components. Refer to Orders in the Guides section for more information about orders, including order attributes, order statuses, and managing orders in the Admin interface.

Orders Resource

You can interact with orders and perform order operations using the orders resource: commerce/orders. There are a number of operations you can perform on orders and order subresources such as order items and order attribute, but key order-level operations include:

- Get Order
- Get Orders
- Create Order
- Cancel Order
- Update Order

> Note that there are two similar fields related to credit card numbers in the Create Order API: `cardNumberPartOrMask` and `ccLastFour`. Order Management-only configurations store the credit card token in the card number field, which may result in a different set of last four digits than the actual card number. Thus, they should use `ccLastFour` to record the actual last four digits of the card number so that these digits can be displayed in the order details of the Admin UI - if not provided, Kibo will display the last four digits of the tokenized number.
>
> However, eCommerce and eCommerce+Order Management configurations use the `cardNumberPartOrMask` field as described in the Create Order API model in which its value is the actual or masked credit card number. Thus, `ccLastFour` is not required because Kibo will display the last four digits from `cardNumberPartOrMask` in the Admin UI. However, using either one or both fields is acceptable for validation - if both are provided, then `ccLastFour` will take priority.

Import Orders for OMS-Only

Order Management-only implementations that do not use Kibo eCommerce need to import orders from their front end. This process uses the Create Order API and requires the following basic information.

- **Email**
- **Customer Account ID**

- **Import Flag:** The `isImport` flag should be set to "true."
- **Items**: Provide as many product details from the API model as possible for the item, including pricing fields from the supported list.
- **Fulfillment Info**: This contains the fulfillment contact and shipping address of the customer, as well as the shipping method name and code. Supported shipping methods will depend on the configurations you enabled in your shipping settings.
- **Payment Info**: Credit card authorization will happen on the external front end, but details will need to be provided via API to perform subsequent operations such as capturing and crediting. It is your responsibility to tokenize the credit card with the gateway provider and send the tokenized card in clear text along with authorization details.
    - Authorization details are located in the interaction object, where the `interactionType` is "Authorization" and the status is "Authorized." Set the `amount` requested field to the authorized amount on the payment, and send required gateway information in `gatewayResponseData`.
    - Fraud check happens on your external front end system. Send the fraud results in the `validationResults` object, with the `status` as either "Pass" or "Review." Review will put the order in Pending Review status.
- **Billing Info**: This should be populated for both order and payment objects, containing the billing contact, address, and card. The card details should indicate `card.isTokenized=true` while `card.cardNumberPartOrMask` will contain the tokenized card number.

The example below shows an imported OMS-Only order for 3 quantity of a single product.

```
{
 "customerAccountId": 00100,
 "email": "example@kibo.com",
 "type": "Offline",
 "isImport": true,
 "externalId": "23948738939298",
 "tenantId": 12345,
 "siteId": 54321,
 "channelCode": "Online",
 "currencyCode": "USD",
 "items": [
   {
    "fulfillmentLocationCode": "AUS",
    "fulfillmentMethod": "Pickup",
    "product":
   {
    "fulfillmentLocationCode": "AUS",
    "fulfillmentMethod": "Pickup",
    "product": {
      "fulfillmentTypesSupported": [
        "DirectShip",
        "InStorePickup"
      ],
      "imageAlternateText": "",
        "imageAlternateText": "",
        "imageUrl": "//example.com/exampleImageURL",
        "options": [],
        "properties": [],
        "price": {
          "price": 10,
          "salePrice": 8.50
        },
        "discountsRestricted": false,
        "isTaxable": true,
        "productType": "Basic Standard",
        "productUsage": "Standard",
        "bundledProducts": [],
        "productCode": "sp_01",
        "name": "Standard Product - 1",
        "description": "Standard Product - 1",
        "goodsType": "Physical",
        "isPackagedStandAlone": false,
        "measurements": {},
      "quantity": 3,
      "discountTotal": 0,
      "productDiscounts": [],
      "shippingDiscounts": [],
      "weightedOrderDiscount": 0,
      "weightedOrderHandlingFeeTax": 0,
      "weightedOrderHandlingFeeDiscount": 0
    }
  ],
  "billingInfo": {
    "paymentType": "CreditCard",
    "billingContact": {
      "email": "example@kibo.net",
      "firstName": "Ming",
      "lastNameOrSurname": "Ming",
      "phoneNumbers": {
        "mobile": "512-555-5555"
      },
      "address": {
        "address1": "123 ABC",
        "address2": "",
        "cityOrTown": "Austin",
        "stateOrProvince": "TX",
        "postalOrZipCode": "78758",
        "countryCode": "US",
        "addressType": "Residential",
        "isValidated": false
      }
    },
    "isSameBillingShippingAddress": false,
    "card": {
      "isUsedRecurring": false,
      "nameOnCard": "Ming Ming",
      "isCardInfoSaved": false,
      "paymentOrCardType": "VISA",
      "cardNumberPartOrMask": "0160184007701111",
      "isTokenized": true,
      "expireMonth": 12,
      "expireYear": 2020
    },
    "auditInfo": {
```

```
    "updateDate": "2017-05-24T00:50:52.926Z",
    "createDate": "2017-05-24T00:21:49.613Z",
    "updateBy": "355060a60a5e48eeb7f2fb8d92af2ba5",
    "createBy": "355060a60a5e48eeb7f2fb8d92af2ba5"
  }
},
"payments": [
  {
    "paymentType": "CreditCard",
    "paymentWorkflow": "Mozu",
    "billingInfo": {
      "paymentType": "CreditCard",
      "billingContact": {
        "email": "example@kibo.net",
        "firstName": "Ming",
        "lastNameOrSurname": "Ming",
        "phoneNumbers": {
          "mobile": "512-555-5555"
        },
        "address": {
          "address1": "123 ABC",
          "cityOrTown": "Austin",
          "stateOrProvince": "TX",
          "postalOrZipCode": "78758",
          "countryCode": "US",
          "addressType": "Residential",
          "isValidated": false
        }
      },
      "isSameBillingShippingAddress": false,
      "card": {
        "isUsedRecurring": false,
        "nameOnCard": "Sam Billing",
        "isCardInfoSaved": false,
        "paymentOrCardType": "MC",
        "cardNumberPartOrMask": "0160184007701111",
        "isTokenized": true,
        "expireMonth": 12,
        "expireYear": 2021
      },
      "auditInfo": {
        "updateDate": "2018-09-17T17:54:21.027Z",
        "createDate": "2018-09-17T17:53:24.598Z",
        "updateBy": "355060a60a5e48eeb7f2fb8d92af2ba5",
        "createBy": "355060a60a5e48eeb7f2fb8d92af2ba5"
      }
    },
    "status": "Authorized",
    "subPayments": [],
    "interactions": [
      {
        "gatewayInteractionId": 155627160,
        "paymentId": "d5abdc686f4b4247ae75a95e01271446",
        "currencyCode": "USD",
        "interactionType": "Authorization",
        "status": "Authorized",
        "paymentEntryStatus": "New",
        "isRecurring": false,
        "isManual": false,
        "gatewayTransactionId": "40018563181",
          "gatewayTransactionId": "40018563191",
          "gatewayAuthCode": "ASQLVO",
          "gatewayAVSCodes": "Y",
          "gatewayCVV2Codes": "P",
          "gatewayResponseCode": "1",
          "gatewayResponseText": "This transaction has been approved.",
          "gatewayResponseData": [
            {
              "key": "AuthorizationRequestId",
              "value": "5642607567736184003012"
            },
            {
              "key": "AuthorizationRequestToken",
              "value": "Ahj/7wSTMeLpFwo2JFnENyzVs0ZNmDdq2bt2bZi4aMGDNgxZKNP3ock7AVGn70O
Sd6QNnBx/EMmkmW6QHez7QJyZjxdIuFGxIs4gLxmU"
            },
            {
              "key": "currencyCode",
              "value": "USD"
            }
          ],
          "amount": 30,
          "interactionDate": "2018-09-17T17:54:21.098Z"
        }
      ],
      "isRecurring": false,
      "amountCollected": 0,
      "amountCredited": 0,
      "amountRequested": 30
    }
  ],
  "amountAvailableForRefund": 30,
  "amountRemainingForPayment": 0,
  "amountRefunded": 0,
  "customerInteractionType": "Unknown",
  "fulfillmentInfo": {
    "fulfillmentContact": {
      "email": "example2@kibo.net",
      "firstName": "Ga",
      "lastNameOrSurname": "Ga",
      "phoneNumbers": {
        "mobile": "1-512-739-1485"
      },
      "address": {
        "address1": "123 W Example Ave",
        "cityOrTown": "Sheridan",
        "stateOrProvince": "TX",
        "postalOrZipCode": "78717",
        "countryCode": "US",
        "addressType": "Residential",
        "isValidated": true
      }
    },
    "shippingMethodCode": "fedex_FEDEX_2_DAY",
    "shippingMethodName": "5% of order",
  },
  "total": 30,
  "isOptInForSms": true,
}
```

Supported Pricing Fields

You can only send certain item-level pricing fields when importing an OMS-Only order, as shown above in the `items` object. These fields are:

Field	**Data Type**	**Description**
itemTaxTotal	number	The tax price for all quantity of this item, not for a single unit.
shippingTotal	number	The total shipping price for all quantity of this item, not for a single unit.
shippingTaxTotal	number	The total shipping tax for all quantity of this item, not for a single unit.
handlingAmount	number	The total handling fee for all quantity of this item, not for a single unit.
dutyAmount	number	The total duty fee for all quantity of this item, not for a single unit.
discountTotal	number	The overall discount applied to the line item, including item-level product and shipping discounts (excluding order-level discounts).
weightedOrderDiscount	number	The portion of an order-level discount that applies to the individual item (excluding any portion of the order handling fee discount).
weightedOrderHandlingFeeDiscount	number	The portion of the order-level handling fee discount applied to the individual item.
weightedOrderHandlingFeeTax	number	The portion of the order-level handling fee tax applied to the individual item.
price	number	The unit price of the product that you intend to sell the product at if no sale price is present. This is part of the Item.Product.Price object.
tenantOverridePrice	number	The override price applied on the product by the tenant. This is part of the Item.Product.Price object.

Field	Data Type	Description
salePrice	number	The current sale price of the product, which is a specific numerical amount (not a percentage off). This is part of the Item.Product.Price object.

OMS-Only Payment Details

Payment information will look different depending on whether you are using Authorize.net or a No-Op/Third Party gateway.

Authorize.net Tokenization

The Authorize.net gateway adapter supports tokenization from Authorize.net using customer profiles. The required information is:

- `card.cardNumberPartOrMask`
- `customerPaymentProfileId` in `payment.data`
- `TransactionId` and `ApprovalCode` (authCode) in `interactions.gatewayResponseData`

```
{
   "paymentType": "CreditCard",
   "paymentWorkflow": "Mozu",
   "billingInfo": {
      "paymentType": "CreditCard",
      "billingContact": {
         "email": "sam.billing@email.com",
         "firstName": "Sam",
         "middleNameOrInitial": "",
         "lastNameOrSurname": "Billing",
         "phoneNumbers": {
            "home": "1234567895",
            "mobile": "1234567895",
            "work": ""
         },
         "address": {
            "address1": "1845 Kramer Ln",
            "address2": "",
            "address3": "",
            "address4": "",
            "cityOrTown": "Austin",
            "stateOrProvince": "TX",
            "postalOrZipCode": "78758",
            "countryCode": "US",
            "addressType": "Residential",
            "isValidated": false
         }
      },
      "isSameBillingShippingAddress": false,
        "card": {
            "isUsedRecurring": false,
            "nameOnCard": "Sam Billing",
            "isCardInfoSaved": false,
            "paymentOrCardType": "VISA",
            "cardNumberPartOrMask": "1510780557",
            "isTokenized": true,
            "expireMonth": 12,
            "expireYear": 2021
        }
    },
    "status": "Authorized",
    "subPayments": [],
    "data": {
        "customerPaymentProfileId": "1510480795"
    },
    "interactions": [
        {
            "gatewayInteractionId": 155627160,
            "paymentId": "d5abdc686f4b4247ae75a95e01271446",
            "currencyCode": "USD",
            "interactionType": "Authorization",
            "status": "Authorized",
            "paymentEntryStatus": "New",
            "isRecurring": false,
            "isManual": false,
            "gatewayTransactionId": "40018563191",
            "gatewayAuthCode": "ASQLVO",
            "gatewayAVSCodes": "Y",
            "gatewayCVV2Codes": "P",
            "gatewayResponseCode": "1",
            "gatewayResponseText": "This transaction has been approved.",
            "gatewayResponseData": [
                {
                    "key": "TransactionId",
                    "value": "40045193885"
                },
                {
                    "key": "ApprovalCode",
                    "value": "C7TZL9"
                }
            ],
            "amount": 192.5,
            "interactionDate": "2018-09-17T17:54:21.098Z"
        }
    ],
    "isRecurring": false,
    "amountCollected": 0,
    "amountCredited": 0,
    "amountRequested": 192.5
}
```

No-Op and Third Party Payments

For no-operation payments, send the payment as a credit card with dummy payment details and set up a no-op gateway to process different card types. This gateway will return a success response for every transaction on the payment without any real interaction with the payment gateway.

Paypal Express 2 is supported as a third party payment for imported orders. You need to have the Paypal Express 2 application installed on your tenant. The Create Order request should include:

- The “PaypalExpress2” payment type
- PayPal token in `payment.externalTransactionId`
- Payer ID in `payment.billingInfo.data`
- Auth transaction ID in `payment.interaction.gatewayTransactionId`

Below is a sample payment object with the PayPal payment.

```
{
  "paymentType": "PayPalExpress2",
  "paymentWorkflow": "PayPalExpress2",
  "externalTransactionId": "EC-74R44913L24993252",
  "billingInfo": {
    "paymentType": "PayPalExpress2",
    "paymentWorkflow": "PayPalExpress2",
    "isSameBillingShippingAddress": false,
    "billingContact": {
      "email": "example@kibocommerce.com"
    },
    "data": {
      "paypal": {
        "payerId": "B373JG5S4Y388"
      }
    }
  },
  "status": "Authorized",
  "subPayments": [],
  "interactions": [
    {
      "id": "6cc37e2c8b6d41e8a2afaaf6010c3108",
      "paymentId": "b719cda9a6734d9599c3aaf6010bc5dc",
      "orderId": "0eba14f72bb54200014d8a720000433f",
      "target": {
        "targetType": "Order",
        "targetId": "0eba14f72bb54200014d8a720000433f",
        "targetNumber": 2022
      },
      "interactionType": "Authorization",
      "status": "Authorized",
      "paymentEntryStatus": "New",
      "isRecurring": false,
      "isManual": false,
      "gatewayTransactionId": "8AF73994TM546221D",
      "gatewayResponseCode": "200",
      "gatewayResponseText": "Success - c3b9386056e4a",
      "amount": 26.10,
      "interactionDate": "2019-10-30T16:16:27.329Z",
      "auditInfo": {}
    }
  ],
  "isRecurring": false,
  "amountCollected": 0,
  "amountCredited": 0,
  "amountRequested": 26.10
}
```

Import Completed Orders

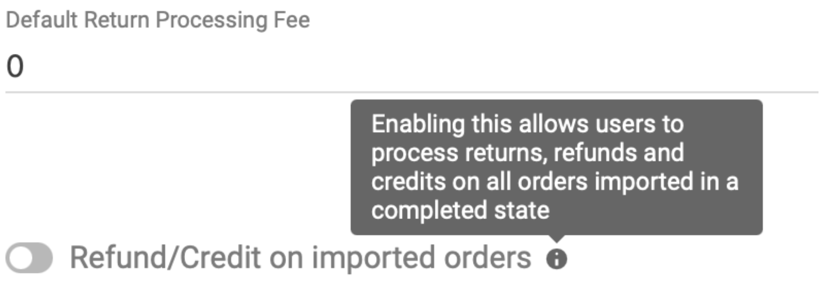
Implementations that include eCommerce can import past orders from a previous system that have been completed. Importing these historical orders allows you to maintain their records in KCCP and perform returns, refunds, and credits on them.

These imported orders can only have the Completed or Canceled order status.

Enable Site Setting

Before importing, you may want to enable a site setting to allow making returns and credits on imported completed orders. This will allow the system to create fulfilled shipments for these orders and process returns, refunds, and credits on them.

1. Go to **System** > **Settings** > **General**.
2. Click the **Site** tab and locate the fulfiller settings section.
3. Toggle on **Refund/Credit on imported orders**.


Default Return Processing Fee

0

Enabling this allows users to process returns, refunds and credits on all orders imported in a completed state

Refund/Credit on imported orders

If this setting is not enabled, then you will still be able to import completed orders. However, their shipment records will not be created and you will not be able to process any returns, refunds, or credits.

Import Completed Order Data

Use the full Create Order payload to formulate your request, setting `isHistoricalImport` to "true" to indicate that it is a past order. You must include payment authorization and capture interaction information in order to support any refunds or credits on this order.

Shipments will be created in the Fulfilled state and the `isHistoricalImport` flag will be passed to them, which will not trigger any events such as status change notifications. If you do not include fulfillment location codes in the order item data, KCCP will use the default location as configured in the shipping settings. If there is no provided location code or default location, then an error will be returned and the order will not be imported.

Once imported, you will be able to view the completed orders and shipments in the Admin UI and Fulfiller UI where you can also perform refunds, returns, and credits if enabled. In both interfaces, there will be a message indicating that the order/shipment is a historical import and cannot be edited.

Pricing Field Mappings

Some pricing fields use slightly different names in the Admin UI compared to the Orders API. The following table maps each field in the Admin to its corresponding API property.

Admin Field	API Property
Base price of line item	`Order.items.UnitPrice` (`listAmount` or `saleAmount`) - `Order.items.product.bundledProducts.deltaPrice` (The sum of the order price)
Manual adjustment on base price	`Order.items.UnitPrice.overrideAmount`
Extras	`Order.items.product.bundledProducts.deltaPrice`
Quantity	`Order.items.quantity`
Weighted order adjustment	`Order.item.weightedOrderAdjustment`
Weighted order discount	`Order.item.weightedOrderDiscount.` This property only stores the discount's value. `Order.orderDiscounts.discount.name` stores the discount's name.
Adjusted line item subtotal	`Order.item.adjustedLineItemSubtotal`
Line item tax	`Order.item.itemTaxTotal` or `Order.item.weightedOrderTax`
Line item total	`Order.item.totalWithoutWeightedShippingandHandling`
Weighted order shipping fee	`Order.item.weightedOrderShipping`
Weighted order shipping discount	`Order.item.weightedOrderShippingDiscount`
Line item shipping fee	`Order.item.ShippingAmountBeforeDiscountAndAdjustments`
Line item shipping discount	`Order.item.shippingDiscounts.discount.discount` (`id` , `name` , and `amount`)
Shipping tax	`Order.item.shippingTax`
Line item order handling fee	`Order.item.handlingAmount`

Admin Field	API Property
Weighted order handling fee	`Order.item.weightedOrderHandlingFee`
Handling tax	`Order.item.weightedOrderHandlingFeeTax`
Weighted order handling discount	`Order.item.weightedOrderHandlingFeeDiscount`
Weighted order duty	`Order.item.weightedOrderDuty`
Line item total w/ shipping and handling	`Order.item.totalWithWeightedShippingandHandling`

The following table maps each order-level pricing field in the Admin to its corresponding API property.

Admin Field	API Property
Order adjustments	Sum of all manual adjustments + all order discounts. Order discounts is a calculated value of all order level discounts.
Manual adjustment	`Order.adjustment`
Order discounts	`Order.orderDiscounts` (`id` , `name` , and `amount`)
Order subtotal	`Order.lineItemSubtotalsWithOrderAdjustments`
Order shipping fees	`Order.shippingAmountBeforeDiscountsAndAdjustments`
Order shipping discounts	`Order.shippingDiscounts`
Order handling fees	`Order.handlingSubtotal`
Order handling discounts	`Order.handlingDiscounts`
Shipping tax	`Order.shippingTaxTotal`

Admin Field	API Property
Handling tax	`Order.handlingTaxTotal`
Order tax	`Order.itemTaxTotal`
Duty	`Order.dutyTotal`
Shipping	Sum of order shipping fees + order shipping discounts + line item shipping fees + line item shipping discounts
Handling	Sum of order handling fees + order handling discounts + line item handling fees + line item handling discounts
Tax	Sum of order tax + shipping tax + handling tax + duty
Order total	`Order.total`

Order Drafts

Order drafts let you make incremental modifications to the order without immediately committing those changes. You can use order drafts to make order item adjustments, change item quantities, or apply additional coupons or discounts. You cannot make changes to an order payment, package, shipment, or pickup using order drafts.

Kibo stores the order draft separately from the original order, but the draft maintains a link to the original order. When performing the GetOrder operation, you can choose to retrieve the draft or the original using the `draft` parameter.

After either you or a shopper creates an order, you can use the following API order operations to create or update an order draft:

- CreateOrder
- UpdateOrder
- DeleteOrderDraft

When you update an order draft, you can perform any of the following actions:

Apply to draft	This action saves and applies any modifications made to the existing draft. If the draft includes previously saved modifications, the new modifications are added to the draft. To perform this action, run the UpdateOrder operation and set the `updateMode` parameter to `ApplyToDraft`.

Apply to original	This action saves and applies any modifications made to the draft to the original order and deletes the existing draft. If the draft included any previously saved modifications, these modifications are lost. To perform this action, run the UpdateOrder operation and set the `updateMode` parameter to `ApplyToOriginal`.
Apply and commit	This action applies all new and prior modifications associated with the draft to the original order and deletes the existing draft. To perform this action, run the UpdateOrder operation and set the `updateMode` parameter to `ApplyAndCommit`.

Each time you save an order draft as the original, or apply and commit the order draft, Kibo creates a new version of the order and increments the version number by one. If multiple users make simultaneous modifications to an order draft, Kibo applies the modifications to the draft based on the first user to save or commit the changes.

Queued Orders

When orders are submitted to Kibo, they are placed into a queue and then processed on a first-in, first-out basis, allowing detection of any failures during order processing.

You can view the queued orders and historical queued orders by using the following APIs:

- Get Queued Order: Allows you to view a specific order from all orders in the queue.
- Get Queued Orders: Allows you to view all orders in the queue.
- Get Historical Order: Allows you to view a single order from all the historical orders in the queue.
- Get Historical Orders: Allows you to view all the historical orders in the queue.

Edit and Cancellation Restrictions

After an order has been submitted, you can use the Update Order Restrictions API to set two flags which will determine whether a customer service representative is allowed to edit or cancel the order. These flags are `restrictEdit` and `restrictCancellation`. If either is set to true, then only roles with the Override Order Update Restriction user behavior will be able to edit and/or cancel the order. If not set, then they will default to false and the order will not be restricted.

If the user has been restricted from editing the order, then it will be displayed in a read-only state on the Admin (but they will still be able to edit internal order notes and initiate or process returns). If the user has been restricted from cancelling the order but they are still allowed to edit it, then they will be able to cancel shipment line items.

The call is made using the below format:

```
.../commerce/orders//updateorderrestrictions?restrictEdit=true&restrictCancellation=true
```