OrderSummary Class

Work with orders in Order Management.

Namespace

ConnectApi

OrderSummary Methods

These methods are for `OrderSummary`. All methods are static.

- **adjustPreview(orderSummaryId, adjustInput)**
  Retrieve the expected results of adjusting the price of one or more OrderItemSummaries from an OrderSummary, without actually executing the adjustment. The response data contains the financial changes that would result from submitting the proposed adjustment.
- **adjustSubmit(orderSummaryId, adjustInput)**
  Adjust the price of one or more OrderItemSummaries from an OrderSummary, and create corresponding change orders.
- **createCreditMemo(orderSummaryId, creditMemoInput)**
  Create a credit memo to represent the refund for one or more change orders associated with an OrderSummary.
- **createMultipleInvoices(invoicesInput)**
  Create Invoices to represent the charges for one or more change orders. Create Invoices for change orders that increase order amounts, such as for return fees. When you ensure the refund for a return, include the invoices for any associated return fees in the request.
- **ensureFundsAsync(orderSummaryId, ensureFundsInput)**
  Ensure funds for an Invoice and apply them to it. If needed, capture authorized funds by sending a request to a payment provider. This method inserts a background operation into an asynchronous job queue and returns the ID of that operation so you can track its status. Payment gateway responses appear in the payment gateway log and do not affect the background operation status.
- **ensureRefundsAsync(orderSummaryId, ensureRefundsInput)**
  Ensure refunds for a CreditMemo or excess funds by sending a request to a payment provider. This method inserts a background operation into an asynchronous job queue and returns the ID of that operation so you can track its status. Payment gateway responses appear in the payment gateway log and don’t affect the background operation status.
- **multipleEnsureFundsAsync(multipleEnsureFundsInput)**
  Ensure and apply funds for one or more Invoices. If needed, capture authorized funds by sending a request to a payment provider. This method inserts a background operation into an asynchronous job queue and returns the ID of that operation so you can track its status. Payment gateway responses appear in the payment gateway log and do not affect the background operation status.
- **previewCancel(orderSummaryId, changeInput)**
  Retrieve the expected change order values for canceling one or more OrderItemSummaries from an OrderSummary, without actually executing the cancel.
- **previewReturn(orderSummaryId, changeInput)**
  Retrieve the expected change order values for a simple return of one or more OrderItemSummaries from an OrderSummary, without actually executing the return.

Return one or more OrderItemSummaries from an OrderSummary, and create a corresponding change order. This return is a simple return that creates a change order but not a ReturnOrder.

adjustPreview(orderSummaryId, adjustInput)

Retrieve the expected results of adjusting the price of one or more OrderItemSummaries from an OrderSummary, without actually executing the adjustment. The response data contains the financial changes that would result from submitting the proposed adjustment.

API Version

49.0

Requires Chatter

No

Signature

```
public static ConnectApi.AdjustOrderSummaryOutputRepresentation adjustPreview(String orderSummaryId, ConnectApi.AdjustOrderItemSummaryInputRepresentation adjustInput)
```

Parameters

orderSummaryId

Type: String

ID of the OrderSummary.

adjustInput

Type: ConnectApi.AdjustOrderItemSummaryInputRepresentation

Price adjustments to order item summaries that together make up a price adjustment to an order, with options for adjusting items in the process of being fulfilled.

Return Value

Type: ConnectApi.AdjustOrderSummaryOutputRepresentation

Usage

When a price adjustment is applied to an OrderItemSummary, its quantities are considered in three groups:

Pre-fulfillment

QuantityAvailableToFulfill, which is equal to QuantityOrdered - QuantityCanceled - QuantityAllocated

In-fulfillment

QuantityAllocated - QuantityFulfilled

Post-fulfillment

QuantityAvailableToReturn, which is equal to QuantityFulfilled - QuantityReturnInitiated

You can apply adjustments to these groups in three different ways, controlled by the *allocatedItemsChangeOrderType* input property:

- Distribute the adjustment evenly between pre-fulfillment and post-fulfillment quantities. Ignore in-fulfillment quantities. Submitting the adjustment would create one change order for the adjustments to pre-fulfillment quantities and one change order for the adjustments to post-fulfillment quantities.

- Distribute the adjustment evenly between pre-fulfillment, in-fulfillment, and post-fulfillment quantities. Submitting the adjustment would create one change order for the adjustments to pre-fulfillment quantities, one change order for the adjustments to in-fulfillment quantities, and one change order for the adjustments to post-fulfillment quantities.

See Also

- createCreditMemo(orderSummaryId, creditMemoInput)
- ensureRefundsAsync(orderSummaryId, ensureRefundsInput)
- adjustSubmit(orderSummaryId, adjustInput)

adjustSubmit(orderSummaryId, adjustInput)

Adjust the price of one or more OrderItemSummaries from an OrderSummary, and create corresponding change orders.

API Version

49.0

Requires Chatter

No

Signature

```
public static ConnectApi.AdjustOrderSummaryOutputRepresentation adjustSubmit(String
orderSummaryId, ConnectApi.AdjustOrderItemSummaryInputRepresentation adjustInput)
```

Parameters

orderSummaryId

Type: String

ID of the OrderSummary.

adjustInput

Type: ConnectApi.AdjustOrderItemSummaryInputRepresentation

Price adjustments to order item summaries that together make up a price adjustment to an order, with options for adjusting items in the process of being fulfilled.

Return Value

Type: ConnectApi.AdjustOrderSummaryOutputRepresentation

Usage

When a price adjustment is applied to an OrderItemSummary, its quantities are considered in three groups:

Pre-fulfillment

QuantityAvailableToFulfill, which is equal to QuantityOrdered - QuantityCanceled - QuantityAllocated

In-fulfillment

QuantityAllocated - QuantityFulfilled

Post-fulfillment

QuantityAvailableToReturn, which is equal to QuantityFulfilled - QuantityReturnInitiated fulfillment quantities and one change order for the adjustments to post-fulfillment quantities.

- Distribute the adjustment evenly between pre-fulfillment, in-fulfillment, and post-fulfillment quantities. Create one change order for the adjustments to both pre-fulfillment and in-fulfillment quantities, and one change order for the adjustments to post-fulfillment quantities.
- Distribute the adjustment evenly between pre-fulfillment, in-fulfillment, and post-fulfillment quantities. Create one change order for the adjustments to pre-fulfillment quantities, one change order for the adjustments to in-fulfillment quantities, and one change order for the adjustments to post-fulfillment quantities.

After submitting a price adjustment, process refunds as appropriate:

- If the discount only applied to OrderItemSummaries for which payment hasn't been captured, it doesn't require a refund. This situation normally applies to OrderItemSummaries in the US that haven't been fulfilled.
- If the discount applied to OrderItemSummaries that haven't been fulfilled and for which payment has been captured, process a refund. In this case, pass the *totalExcessFundsAmount* from the output representation to the `ensureRefundsAsync()` method.
- If the discount applied to OrderItemSummaries that have been fulfilled, process a refund. Pass the *postFulfillmentChangeOrderId* from the output representation to the `createCreditMemo()` method, then pass the CreditMemo to the `ensureRefundsAsync()` method.
- If the discount applied to both fulfilled and unfulfilled OrderItemSummaries for which payment has been captured, process both refunds. Pass the *postFulfillmentChangeOrderId* from the output representation to the `createCreditMemo()` method, then pass the credit memo and the *totalExcessFundsAmount* from the output representation to the `ensureRefundsAsync()` method.

See Also

- createCreditMemo(orderSummaryId, creditMemoInput)
- ensureRefundsAsync(orderSummaryId, ensureRefundsInput)
- adjustPreview(orderSummaryId, adjustInput)

createCreditMemo(orderSummaryId, creditMemoInput)

Create a credit memo to represent the refund for one or more change orders associated with an OrderSummary.

API Version

48.0

Requires Chatter

No

Signature

```
public static ConnectApi.CreateCreditMemoOutputRepresentation createCreditMemo(String
orderSummaryId, ConnectApi.CreateCreditMemoInputRepresentation creditMemoInput)
```

Parameters

orderSummaryId

Type: String

The list of change order IDs.

Return Value

Type: ConnectApi.CreateCreditMemoOutputRepresentation

createMultipleInvoices(invoicesInput)

Create Invoices to represent the charges for one or more change orders. Create Invoices for change orders that increase order amounts, such as for return fees. When you ensure the refund for a return, include the invoices for any associated return fees in the request.

API Version

56.0

Requires Chatter

No

Signature

```
public static ConnectApi.CreateMultipleInvoicesFromChangeOrdersOutputRepresentation
createMultipleInvoices(ConnectApi.CreateMultipleInvoicesFromChangeOrdersInputRepresentation
invoicesInput)
```

Parameters

invoicesInput

Type: ConnectApi.CreateMultipleInvoicesFromChangeOrdersInputRepresentation

Data about the change orders to create Invoices for.

Return Value

Type: ConnectApi.CreateMultipleInvoicesFromChangeOrdersOutputRepresentation

See Also

- ensureRefundsAsync(orderSummaryId, ensureRefundsInput)
- createReturnOrder(returnOrderInput)
- returnItems(returnOrderId, returnItemsInput)

ensureFundsAsync(orderSummaryId, ensureFundsInput)

Ensure funds for an Invoice and apply them to it. If needed, capture authorized funds by sending a request to a payment provider. This method inserts a background operation into an asynchronous job queue and returns the ID of that operation so you can track its status. Payment gateway responses appear in the payment gateway log and do not affect the background operation status.

API Version

48.0

Requires Chatter

No

Signature

isConsiderReservedBalanceAmount

Type: Boolean

If true, the reserved balance amount is used for the Order Summary to fund the invoice. If not enough reserved balance amount, any available balance that isn’t reserved by another Order Summary is used. If false, any available balance is used.

orderSummaryId

Type: String

ID of the OrderSummary.

ensureFundsInput

Type: ConnectApi.EnsureFundsAsyncInputRepresentation

The ID of the Invoice.

Return Value

Type: ConnectApi.EnsureFundsAsyncOutputRepresentation

Usage

This method checks the OrderPaymentSummaries associated with the specified OrderSummary for funds to apply to the Invoice balance following this logic:

> **Note**
>
> If multiple OrderPaymentSummaries have equal `BalanceAmount` values, their order of selection is random.

1. Verify that the Invoice balance doesn’t exceed the total `BalanceAmount` of all the OrderPaymentSummaries associated with the OrderSummary.
2. If an OrderPaymentSummary has a `BalanceAmount` equal to the Invoice balance, apply the funds from that OrderPaymentSummary.
3. If no exact match was found, apply funds from the OrderPaymentSummary with the largest `BalanceAmount`.
4. If the Invoice still has a balance to ensure, repeat steps 2 and 3 until the full balance is ensured or no captured funds remain.
5. If the Invoice still has a balance, look for an OrderPaymentSummary with an authorized amount equal to the remaining Invoice balance. If one exists, capture and apply the funds from that OrderPaymentSummary.
6. If no exact match was found, capture and apply funds from the OrderPaymentSummary with the largest authorized amount.
7. If the Invoice still has a balance to ensure, repeat steps 5 and 6 until the full balance is ensured.

> **Note**
>
> If the method creates a payment, the payment record’s ClientContext value isn’t predictable. Don't use it in custom logic.

See Also

- multipleEnsureFundsAsync(multipleEnsureFundsInput) log and don’t affect the background operation status.

API Version

48.0

Requires Chatter

No

Signature

```
public static ConnectApi.EnsureRefundsAsyncOutputRepresentation ensureRefundsAsync(String
orderSummaryId, ConnectApi.EnsureRefundsAsyncInputRepresentation ensureRefundsInput)
```

Parameters

isConsiderReservedBalanceAmount

Type: Boolean

If true, the refundable amount is used to open the payment balance for the reservedBalanceAmount in the Order Payment Summaries. The remaining refundable amount considers the sequence of order payment summaries, if provided. If false, any reserved balance amount for exchanges is refunded.

orderSummaryId

Type: String

ID of the OrderSummary.

ensureRefundsInput

Type: ConnectApi.EnsureRefundsAsyncInputRepresentation

ID of a credit memo to ensure refunds for, an amount of excess funds to refund, or both. At least one is required. Also includes any invoices for fees that reduce the refund amount, such as return fees. If multiple payment methods are available, you can specify how to distribute the refund.

Return Value

Type: ConnectApi.EnsureRefundsAsyncOutputRepresentation

Usage

This method applies the refund to the OrderPaymentSummaries associated with the specified OrderSummary following this logic.

> **Note**
>
> If multiple OrderPaymentSummaries have equal `AvailableToRefund` amounts, their order of selection is random.

1. Verify that the CreditMemo balance and excess funds amount don't exceed the total `AvailableToRefund` amount of all the OrderPaymentSummaries associated with the OrderSummary.
2. If `sequences` is specified, follow these steps.
   a. Traverse the `sequences` list in order and apply the specified refund amounts to the specified OrderPaymentSummaries.
   b. If the specified CreditMemo and excess funds are fully refunded, or if `isAllowPartial` is true, then the action stops here.

enough `AvailableToRefund` amount to cover the balance, use the OrderPaymentSummary with the smallest `AvailableToRefund` amount.

   c. If no single OrderPaymentSummary has a large enough `AvailableToRefund` amount, use multiple OrderPaymentSummaries in descending order of `AvailableToRefund` amount. This ensures the fewest OrderPaymentSummaries are used.

4. If only one OrderPaymentSummary is specified but has multiple payments, follow these steps.
   a. If a payment has an amount matching the CreditMemo's remaining balance, apply the refund to that payment.
   b. If no exact match was found but one or more payment has a large enough amount to cover the balance, use the payment with the smallest amount.
   c. If no single payment has a large enough amount, use multiple payments in descending order of amount. This ensures the fewest payments are used.
5. If an excess funds amount is specified, follow these steps.
   a. Examine those OrderPaymentSummaries. If one has an `AvailableToRefund` amount matching the excess funds amount, apply the refund to that OrderPaymentSummary.
   b. If no exact match was found but one or more OrderPaymentSummary has a large enough `AvailableToRefund` amount to cover the balance, use the OrderPaymentSummary with the smallest `AvailableToRefund` amount.
   c. If no single OrderPaymentSummary has a large enough `AvailableToRefund` amount, use multiple OrderPaymentSummaries in descending order of `AvailableToRefund` amount. This ensures the fewest OrderPaymentSummaries are used.

Note

If the method creates a refund, the refund record's ClientContext value isn't predictable. Don't use it in custom logic.

See Also

- createReturnOrder(returnOrderInput)
- returnItems(returnOrderId, returnItemsInput)
- createMultipleInvoices(invoicesInput)

multipleEnsureFundsAsync(multipleEnsureFundsInput)

Ensure and apply funds for one or more Invoices. If needed, capture authorized funds by sending a request to a payment provider. This method inserts a background operation into an asynchronous job queue and returns the ID of that operation so you can track its status. Payment gateway responses appear in the payment gateway log and do not affect the background operation status.

API Version

56.0

Requires Chatter

No

Signature

```
public static ConnectApi.MultipleAsyncOutputRepresentation
multipleEnsureFundsAsync(ConnectApi.MultipleEnsureFundsAsyncInputRepresentation
multipleEnsureFundsInput)
```

List of Invoices and the associated OrderSummaries.

Return Value

Type: ConnectApi.MultipleAsyncOutputRepresentation

Usage

For each Invoice in the request, this method checks the OrderPaymentSummaries associated with the specified OrderSummary for funds to apply to the Invoice balance following this logic.

> **Note**
>
> If multiple OrderPaymentSummaries have equal `BalanceAmount` values, their order of selection is random.

1. Verify that the Invoice balance doesn't exceed the total `BalanceAmount` of all the OrderPaymentSummaries associated with the OrderSummary.
2. If an OrderPaymentSummary has a `BalanceAmount` equal to the invoice balance, apply the funds from that OrderPaymentSummary.
3. If no exact match was found, apply funds from the OrderPaymentSummary with the largest `BalanceAmount`.
4. If the Invoice still has a balance to ensure, repeat steps 2 and 3 until the full balance is ensured or no captured funds remain.
5. If the Invoice still has a balance, look for an OrderPaymentSummary with an authorized amount equal to the remaining Invoice balance. If one exists, capture and apply the funds from that OrderPaymentSummary.
6. If no exact match was found, capture and apply funds from the OrderPaymentSummary with the largest authorized amount.
7. If the Invoice still has a balance to ensure, repeat steps 5 and 6 until the full balance is ensured.

> **Note**
>
> If the method creates a payment, the payment record's ClientContext value isn't predictable. Don't use it in custom logic.

See Also

- ensureFundsAsync(orderSummaryId, ensureFundsInput)

previewCancel(orderSummaryId, changeInput)

Retrieve the expected change order values for canceling one or more OrderItemSummaries from an OrderSummary, without actually executing the cancel.

API Version

48.0

Requires Chatter

No

Signature

orderSummaryId

Type: String

ID of the OrderSummary.

changeInput

Type: ConnectApi.ChangeInputRepresentation

A list of changes to OrderItemSummaries that make up an order change, such as a cancel or return.

Return Value

Type: ConnectApi.PreviewCancelOutputRepresentation

See Also

- createCreditMemo(orderSummaryId, creditMemoInput)
- ensureRefundsAsync(orderSummaryId, ensureRefundsInput)
- submitCancel(orderSummaryId, changeInput)

previewReturn(orderSummaryId, changeInput)

Retrieve the expected change order values for a simple return of one or more OrderItemSummaries from an OrderSummary, without actually executing the return.

API Version

48.0

Requires Chatter

No

Signature

```
public static ConnectApi.PreviewReturnOutputRepresentation previewReturn(String orderSummaryId,
ConnectApi.ChangeInputRepresentation changeInput)
```

Parameters

orderSummaryId

Type: String

ID of the OrderSummary.

changeInput

Type: ConnectApi.ChangeInputRepresentation

A list of changes to OrderItemSummaries that make up an order change, such as a cancel or return.

Return Value

Type: ConnectApi.PreviewReturnOutputRepresentation

See Also

- createCreditMemo(orderSummaryId, creditMemoInput)
- ensureRefundsAsync(orderSummaryId, ensureRefundsInput)

Cancel one or more OrderItemSummaries from an OrderSummary, and create a corresponding change order.

API Version

48.0

Requires Chatter

No

Signature

```
public static ConnectApi.SubmitCancelOutputRepresentation submitCancel(String orderSummaryId,
ConnectApi.ChangeInputRepresentation changeInput)
```

Parameters

orderSummaryId

Type: String

ID of the OrderSummary.

changeInput

Type: ConnectApi.ChangeInputRepresentation

A list of changes to OrderItemSummaries that make up an order change, such as a cancel or return.

Return Value

Type: ConnectApi.SubmitCancelOutputRepresentation

See Also

- createCreditMemo(orderSummaryId, creditMemoInput)
- ensureRefundsAsync(orderSummaryId, ensureRefundsInput)
- previewCancel(orderSummaryId, changeInput)

submitReturn(orderSummaryId, changeInput)

Return one or more OrderItemSummaries from an OrderSummary, and create a corresponding change order. This return is a simple return that creates a change order but not a ReturnOrder.

API Version

48.0

Requires Chatter

No

Signature

```
public static ConnectApi.SubmitReturnOutputRepresentation submitReturn(String orderSummaryId,
ConnectApi.ChangeInputRepresentation changeInput)
```

Parameters

orderSummaryId

Type: ConnectApi.ChangeInputRepresentation

A list of changes to OrderItemSummaries that make up an order change, such as a cancel or return.

Return Value

Type: ConnectApi.SubmitReturnOutputRepresentation

Usage

After submitting a return, process a refund. Pass the *changeOrderId* from the output representation to the `createCreditMemo()` method, then pass the credit memo to the `ensureRefundsAsync()` method.

See Also

- createCreditMemo(orderSummaryId, creditMemoInput)
- ensureRefundsAsync(orderSummaryId, ensureRefundsInput)
- previewReturn(orderSummaryId, changeInput)