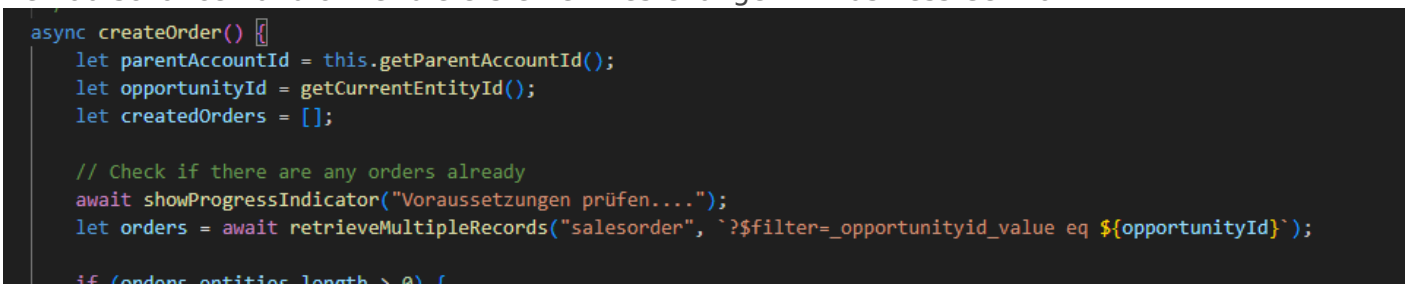
2. Die Klasse VonmaehlenOrderCreationClass

Die Hauptklasse, die in diesem Skript verwendet wird, ist `VonmaehlenOrderCreationClass`. Diese Klasse enthält Methoden zur Erstellung von Bestellungen und unterstützende Hilfsfunktionen.

2.1. Methoden

`createOrder()`: Diese Methode initiiert den Prozess zur automatisierten Auftragsanlage. Sie überprüft vorhandene Bestellungen für eine bestimmte Verkaufschance, speichert die Verkaufschance, ruft erforderliche Daten ab, erstellt Bestellungen basierend auf den Verkaufschancen und öffnet die erstellten Bestellungen in Business Central.



```
async createOrder() {
    let parentAccountId = this.getParentAccountId();
    let opportunityId = getCurrentEntityId();
    let createdOrders = [];

    // Check if there are any orders already
    await showProgressIndicator("Voraussetzungen prüfen....");
    let orders = await retrieveMultipleRecords("salesorder", `?$filter=_opportunityid_value eq ${opportunityId}`);

    if (orders.entities.length > 0) {
```

```
let orderBaseData = {
    "name": `${opportunity["nps_id"]} | ${i} | ${productNumber} | ${opportunity[`cre56_logo_name_${i}`]}`,
    "customerid_account@odata.bind": "/accounts(" + account["accountid"] + ")",
    "opportunityid@odata.bind": "/opportunities(" + opportunityId + ")",
    "pricelevelid@odata.bind": "/pricelevels(" + priceListId + ")",
    "ownerid@odata.bind": "/systemusers(" + opportunity["_ownerid_value"] + ")",
    "cre56_scm_owner@odata.bind": "/systemusers(" + opportunity["_cre56_scm_owner_value"] + ")",
    "cre56_customer_logo": opportunity[`cre56_logo_name_${i}`],
    "billto_contactname": account["address1_primarycontactname"],
    "billto_telephone": account["address1_telephone1"],
    "billto_fax": account["address1_fax"],
    "billto_name": account["address1_name"],
    "emailaddress": account["emailaddress1"],
    "billto_line1": account["address1_line1"],
    "billto_line2": account["address1_line2"],
    "billto_line3": account["address1_line3"],
    "billto_city": account["address1_city"],
    "billto_stateorprovince": account["address1_stateorprovince"],
    "billto_postalcode": account["address1_postalcode"],
    "billto_country": account["address1_country"],
    "transactioncurrencyid": account["transactioncurrencyid"],
    "shippingmethodcode": account["address1_shippingmethodcode"],
    "paymenttermscode": account["paymenttermscode"],
    "freighttermscode": account["address1_freighttermscode"],
    "cre56_order_date": opportunity["cre56_order_date"],
};

let mainSalesOrderData = Object.assign({}, orderBaseData);

if (opportunity["cre56_ship_to_different_address"]) {
    mainSalesOrderData["shipto_contactname"] = opportunity["cre56_ship_to_contact_name"]
    mainSalesOrderData["shipto_name"] = opportunity["cre56_ship_to_address_name"]
    mainSalesOrderData["shipto_line1"] = opportunity["cre56_ship_to_address_line_1"]
    mainSalesOrderData["shipto_line2"] = opportunity["cre56_ship_to_address_line_2"]
    mainSalesOrderData["shipto_city"] = opportunity["cre56_ship_to_city"]
    mainSalesOrderData["shipto_country"] = opportunity["cre56_ship_to_country"]
    mainSalesOrderData["shipto_postalcode"] = opportunity["cre56_ship_to_post_code"]
    mainSalesOrderData["cre56_different_ship_to_address"] = opportunity["cre56_ship_to_different_address"]
}

if (opportunity["cre56_different_bill_to_address"]) {
    mainSalesOrderData["billto_name"] = opportunity["cre56_bill_to_name_1"];
    mainSalesOrderData["billto_contactname"] = opportunity["cre56_bill_to_name_2"];
    mainSalesOrderData["billto_line1"] = opportunity["cre56_bill_to_address_line_1"];
    mainSalesOrderData["billto_line2"] = opportunity["cre56_bill_to_address_line_2"];
    mainSalesOrderData["billto_city"] = opportunity["cre56_bill_to_city"];
    mainSalesOrderData["billto_country"] = opportunity["cre56_bill_to_country"];
    mainSalesOrderData["billto_postalcode"] = opportunity["cre56_bill_to_post_code"];
}
```

- **`postOrder(salesOrderData, salesLinesData)`**: Diese Methode erstellt eine Bestellung und die dazugehörigen Auftragspositionen basierend auf den übergebenen Daten.

```
        if (opportunity['cre56_approval_sample_required']) {
            let approvalSampleSalesOrderData = Object.assign({}, orderBaseData);
            approvalSampleSalesOrderData["name"] += ' | Freigabemuster';

            if (opportunity['cre56_different_approval_sample_address']) {
                approvalSampleSalesOrderData["shipto_name"] = opportunity["cre56_approval_sample_to_name_1"]
                approvalSampleSalesOrderData["shipto_contactname"] = opportunity["cre56_approval_sample_to_name_2"]
                approvalSampleSalesOrderData["shipto_line1"] = opportunity["cre56_approval_sample_to_address_line_1"]
                approvalSampleSalesOrderData["shipto_line2"] = opportunity["cre56_approval_sample_to_address_line_2"]
                approvalSampleSalesOrderData["shipto_city"] = opportunity["cre56_approval_sample_to_city"]
                approvalSampleSalesOrderData["shipto_country"] = opportunity["cre56_approval_sample_to_country"]
                approvalSampleSalesOrderData["shipto_postalcode"] = opportunity["cre56_approval_sample_to_post_code"]
                approvalSampleSalesOrderData["cre56_different_ship_to_address"] = true;
            }

            createdOrders.push(await this.postOrder(approvalSampleSalesOrderData, [
                this.makeSalesLine(account, null, productNumber, 1, pricePerUnit, 100)
            ]));
        }

        if (opportunity['cre56_different_archive_sample_to_address'] && archiveSampleQuantity > 0) {
            let archiveSampleSalesOrderData = Object.assign({}, orderBaseData);

            archiveSampleSalesOrderData["name"] += ' | Belegmuster';
            archiveSampleSalesOrderData["shipto_name"] = opportunity["cre56_archive_sample_to_name_1"]
            archiveSampleSalesOrderData["shipto_contactname"] = opportunity["cre56_archive_sample_to_name_2"]
            archiveSampleSalesOrderData["shipto_line1"] = opportunity["cre56_archive_sample_to_address_line_1"]
            archiveSampleSalesOrderData["shipto_line2"] = opportunity["cre56_archive_sample_to_address_line_2"]
            archiveSampleSalesOrderData["shipto_city"] = opportunity["cre56_archive_sample_to_city"]
            archiveSampleSalesOrderData["shipto_country"] = opportunity["cre56_archive_sample_to_country"]
            archiveSampleSalesOrderData["shipto_postalcode"] = opportunity["cre56_archive_sample_to_post_code"]
            archiveSampleSalesOrderData["cre56_different_ship_to_address"] = true;

            createdOrders.push(await this.postOrder(archiveSampleSalesOrderData, [
                this.makeSalesLine(account, null, productNumber, archiveSampleQuantity, pricePerUnit, discountPercentage)
            ]));
        }
    }
```

- **makeSalesLine(account, salesOrder, product, quantity, pricePerUnit, discountPercentage, discountAbsolute, customName)**: Diese Methode erstellt eine Auftragsposition mit den angegebenen Parametern wie Konto, Produkt, Menge, Preis pro Einheit und Rabatt.

```
            createdOrders.push(await this.postOrder(approvalSampleSalesOrderData, [
                this.makeSalesLine(account, null, productNumber, 1, pricePerUnit, 100)
            ]));
        }
        You, 7 hours ago • change line …
        if (opportunity['cre56_different_archive_sample_to_address'] && archiveSampleQuantity > 0) {
            let archiveSampleSalesOrderData = Object.assign({}, orderBaseData);

            archiveSampleSalesOrderData["name"] += ' | Belegmuster';
            archiveSampleSalesOrderData["shipto_name"] = opportunity["cre56_archive_sample_to_name_1"]
            archiveSampleSalesOrderData["shipto_contactname"] = opportunity["cre56_archive_sample_to_name_2"]
            archiveSampleSalesOrderData["shipto_line1"] = opportunity["cre56_archive_sample_to_address_line_1"]
            archiveSampleSalesOrderData["shipto_line2"] = opportunity["cre56_archive_sample_to_address_line_2"]
            archiveSampleSalesOrderData["shipto_city"] = opportunity["cre56_archive_sample_to_city"]
            archiveSampleSalesOrderData["shipto_country"] = opportunity["cre56_archive_sample_to_country"]
            archiveSampleSalesOrderData["shipto_postalcode"] = opportunity["cre56_archive_sample_to_post_code"]
            archiveSampleSalesOrderData["cre56_different_ship_to_address"] = true;

            createdOrders.push(await this.postOrder(archiveSampleSalesOrderData, [
                this.makeSalesLine(account, null, productNumber, archiveSampleQuantity, pricePerUnit, discountPercentage)
            ]));
        }
    }
```

- **makeAnnotationLine(salesOrder, annotation)**: Diese Methode erstellt eine Anmerkung als Auftragsposition für zusätzliche Informationen.
- **getVatRate(customer)** Diese Methode berechnet die Steuer anhand der Informationen aus dem Kunden.

```
/**
 * Make this async too, so it behaves the same as makeSalesLine
 *
 * @param salesOrder
 * @param annotation
 * @returns {{isproductoverridden: boolean, productdescription}}
 */
async makeAnnotationLine(salesOrder, annotation) {
    return {
        "isproductoverridden": true,
        "productdescription": annotation,
        "quantity": 1,
    };
}

/**
 * Get the VAT rate
 * @returns {number}
 */
getVatRate(customer) {
    if (this.includeVAT(customer)) {
        return 19;
    }

    return 0;
}

/**
 *
 * @param account
 * @param opportunity
 * @returns {Promise<*>}
 */
async getPriceListId(account, opportunity) {
    if (account['_defaultpricelevelid_value']) {
```

2.2. Hilfsfunktionen

- Verschiedene Hilfsfunktionen wie `getVatRate`, `getPriceListId`, `includeVAT`, etc., werden verwendet, um die automatisierte Auftragsanlage zu unterstützen.

```
 * @returns {Promise<*>}
 */
async getPriceListId(account, opportunity) {      You, 7 hours ago • change line …
    if (account['_defaultpricelevelid_value']) {
        return account['_defaultpricelevelid_value'];
    }

    let priceListName = 'EUR_CO-B. IND 24';

    if (opportunity['cre56_ship_to_country'] === "CH") {
        priceListName = 'EUR_CO-B. IND CH 24';
    }

    let priceList = await this.getPriceListByName(priceListName);

    return priceList.pricelevelid;
}
```

```
    /**
     * Get the VAT rate
     * @returns {number}
     */
    getVatRate(customer) {
        if (this.includeVAT(customer)) {
            return 19;
        }

        return 0;
    }

    /**
```

Version #1
Erstellt: 26 März 2024 13:49:02 von Kay Dietrich
Zuletzt aktualisiert: 26 März 2024 13:56:36 von Kay Dietrich

create_order.js development documentation

1. Einführung

Die automatisierte Auftragsanlage ermöglicht es Benutzern, Bestellungen basierend auf Verkaufschancen automatisch zu generieren. Dies verbessert die Effizienz und Genauigkeit des Bestellprozesses und reduziert den manuellen Aufwand erheblich.

2. Klasse: VonmaehlenOrderCreationClass

Die Hauptklasse, die in diesem Skript verwendet wird, ist **VonmaehlenOrderCreationClass**. Diese Klasse enthält Methoden zur Erstellung von Bestellungen und unterstützende Hilfsfunktionen.

2.1. Methoden

- **createOrder()**: Diese Methode initiiert den Prozess zur automatisierten Auftragsanlage. Sie überprüft vorhandene Bestellungen für eine bestimmte Verkaufschance, speichert die Verkaufschance, ruft erforderliche Daten ab, erstellt Bestellungen basierend auf den Verkaufschancen und öffnet die erstellten Bestellungen in Business Central.
- **postOrder(salesOrderData, salesLinesData)**: Diese Methode erstellt eine Bestellung und die dazugehörigen Auftragspositionen basierend auf den übergebenen Daten.
- **makeSalesLine(account, salesOrder, product, quantity, pricePerUnit, discountPercentage, discountAbsolute, customName)**: Diese Methode erstellt eine Auftragsposition mit den angegebenen Parametern wie Konto, Produkt, Menge, Preis pro Einheit und Rabatt.
- **makeAnnotationLine(salesOrder, annotation)**: Diese Methode erstellt eine Anmerkung als Auftragsposition für zusätzliche Informationen.

2.2. Hilfsfunktionen

- Verschiedene Hilfsfunktionen wie **getVatRate**, **getPriceListId**, **includeVAT**, etc., werden verwendet, um die automatisierte Auftragsanlage zu unterstützen.

3. Verwendung

Um die automatisierte Auftragsanlage zu nutzen, muss die Klasse **VonmaehlenOrderCreationClass** initialisiert werden. Dann kann die Methode **createOrder()** aufgerufen werden, um den Prozess zur automatisierten Auftragsanlage zu starten.

import { VonmaehlenOrderCreationClass } from './VonmaehlenOrderCreationClass.js'; const orderCreation = new VonmaehlenOrderCreationClass(); orderCreation.createOrder();

4. Fehlerbehandlung

Das Skript beinhaltet Mechanismen zur Fehlerbehandlung, um sicherzustellen, dass auftretende Fehler angemessen behandelt werden und dem Benutzer hilfreiche Informationen zur Verfügung gestellt werden.

5. Sicherheit

Das Skript implementiert Sicherheitsmechanismen, um sicherzustellen, dass nur autorisierte Benutzer Bestellungen erstellen können und dass sensible Daten angemessen geschützt sind.

6. Test und Qualitätssicherung

Die Funktionalität wurde umfassend getestet, um sicherzustellen, dass sie den Anforderungen entspricht und zuverlässig funktioniert. Alle Änderungen wurden vor der Bereitstellung auf Fehler und Leistungsprobleme überprüft.

7. Zusammenfassung

Die automatisierte Auftragsanlage bietet eine effiziente und zuverlässige Möglichkeit, Bestellungen basierend auf Verkaufschancen zu generieren. Durch die Nutzung dieses Skripts wird der Bestellprozess optimiert und die Produktivität gesteigert.

Version #2
Erstellt: 26 März 2024 13:46:25 von Kay Dietrich
Zuletzt aktualisiert: 26 März 2024 13:47:22 von Kay Dietrich