Sisukord

Dokumentacja Directo API 3
 Pierwsze kroki 3
Wymagania wstępne 3
Szybki start 3
Bazowy URL i wersjonowanie 3
 Uwierzytelnianie 4
Jak się uwierzytelnić 4
Jak uzyskać klucz API 4
Dobre praktyki bezpieczeństwa 4
 Formaty odpowiedzi (JSON vs XML) 4
Format JSON (zalecany) 4
Format XML 5
 Opis API (openapi.json) 5
Co zawiera opis 5
Nazewnictwo parametrów filtrów 5
Jak z tego korzystać 6
 Zaawansowane filtrowanie (kluczowa funkcja) 6
 Filtrowanie w stylu ERP – siła Directo API 6
Co to oznacza dla programistów 6
Pola filtrowalne a pola wyliczane 7
Jak rozpoznać pola wyliczane 7
Przykłady praktyczne: pola filtrowalne a wyliczane 7
Filtrowanie podstawowe 8
Filtrowanie „zawiera” 8
Operatory porównania 8
Filtrowanie zakresu 9
Wiele wartości (logika OR) 9
Filtrowanie według znacznika czasu (synchronizacja) 9
Filtrowanie według pól wierszy 10
Filtrowanie według pól danych dodatkowych 10
Złożone kombinacje filtrów 10
Popularne parametry filtrów według zasobu 11
Dobre praktyki filtrowania 11
Rozwiązywanie problemów z filtrami 11
 API Resources & Endpoints 12
 Struktura odpowiedzi 13
Odpowiedź poprawna (HTTP 200) 14
Cechy odpowiedzi 15
⚠ ***Obsługa błędów*** 15
Kody statusu HTTP 15
Format odpowiedzi błędu 15
Typowe scenariusze błędów 16
 Limity zapytań i wydajność 16
Limity zapytań 16
Dobre praktyki wydajnościowe 17
 Typowe wzorce integracji 18
Wzorzec 1: Początkowa synchronizacja danych (pierwsza konfiguracja) 18
Wzorzec 2: Synchronizacja przyrostowa (bieżąca) 18

Wzorzec 3: Monitorowanie zamówień w czasie rzeczywistym 18
Wzorzec 4: Zarządzanie zapasami i sprawdzanie dostępności 18
Wzorzec 5: Synchronizacja danych klientów (integracja z CRM) 19
Wzorzec 6: Eksport faktur do systemu księgowego 19
Wzorzec 7: Synchronizacja cen ze sklepem internetowym 19
Przewodnik rozwiązywania problemów 19
Problem: pusta odpowiedź / brak wyników 19
Problem: błędy uwierzytelniania (401/403) 20
Problem: błędy limitu zapytań (429) 20
Problem: `usagelogs` – brak starszych wpisów 20
Wsparcie i zasoby 20
Historia zmian i wersji 21
Bezpieczeństwo i zgodność 21
Bezpieczeństwo danych 21
Kwestie zgodności 21
Dobre praktyki bezpieczeństwa 21
Wskazówki i triki 22
Szybka ściągawka 23
Najważniejsze endpointy 23
Uwierzytelnianie 23
Format odpowiedzi 23
Popularne filtry 23
Przykłady filtrów 23
Pamiętaj 23

Dokumentacja Directo API

Witamy w Directo API – kompleksowym interfejsie RESTful API do przeglądania i zarządzania danymi ERP, w tym artykułami, zamówieniami, klientami, fakturami, stanami magazynowymi i nie tylko.

Pierwsze kroki

Wymagania wstępne

- Aktywne konto Directo ERP
- Klucz API (uzyskasz go od administratora Directo)
- Podstawowa znajomość REST API i metod HTTP
- Zasoby i endpointy API znajdziesz w kolekcji Postman. Lista pól odpowiedzi jest zarządzana w Directo ERP. Kolekcję Postman można pobrać tutaj: POSTMAN

Szybki start

1. **Skonfiguruj uwierzytelnianie**: Ustaw klucz API w zakładce Authorization kolekcji lub zaktualizuj nagłówek X-Directo-Key
2. **Ustaw zmienne**: Skonfiguruj baseUrl i version w zmiennych kolekcji
3. **Wybierz format**: Ustaw nagłówek Accept na application/json lub application/xml
4. **Wykonaj pierwsze zapytanie**: Wypróbuj zapytanie „All items“, aby pobrać katalog artykułów
5. **Wywołaj GET /v1/openapi.json**, aby zobaczyć, które zasoby i pola są włączone dla Twojego klucza.

Bazowy URL i wersjonowanie

Struktura bazowego URL:

```
{{baseUrl}}/v{{version}}/{resource}
```

Bieżąca konfiguracja:

- **baseUrl**:

  ```
  https://login.directo.ee/apidirect
  ```

- **version**:

  ```
  1
  ```

  (bieżąca wersja API)

Przykładowy pełny URL:

```
https://login.directo.ee/apidirect/v1/items
```

API korzysta z wersjonowania, aby zapewnić kompatybilność wsteczną. Zawsze podawaj numer wersji w zapytaniach.

Uwierzytelnianie

Directo API używa uwierzytelniania kluczem API (API Key), aby zapewnić bezpieczny dostęp.

Jak się uwierzytelnić

Dołącz klucz API do nagłówka zapytania:

```
X-Directo-Key: YOUR_API_KEY_HERE
```

Jak uzyskać klucz API

Jeśli Twoja firma nie ma jeszcze klucza API, skontaktuj się ze swoim kierownikiem projektu Directo — wygeneruje go dla Ciebie.

Jeśli klucz został już utworzony, nie trzeba prosić o niego ponownie: każdy użytkownik z uprawnieniami SUPER może go zobaczyć na karcie pracownika w sekcji Ustawienia → Integracja.

Dobre praktyki bezpieczeństwa

- Nigdy nie ujawniaj klucza API w kodzie po stronie klienta ani w publicznych repozytoriach
- Przechowuj klucze API w bezpieczny sposób, np. w zmiennych środowiskowych
- Okresowo zmieniaj (rotuj) klucze
- Używaj różnych kluczy dla środowiska deweloperskiego i produkcyjnego

Formaty odpowiedzi (JSON vs XML)

API obsługuje formaty odpowiedzi JSON i XML. Format wybierasz za pomocą nagłówka Accept.

Format JSON (zalecany)

```
Accept: application/json
```

Przykładowe zapytanie:

```
curl -X GET "{{baseUrl}}/v{{version}}/items"
```

```
-H "X-Directo-Key: YOUR_API_KEY"

-H "Accept: application/json"
```

Format XML

```
Accept: application/xml
```

Przykładowe zapytanie:

```
curl -X GET "{{baseUrl}}/v{{version}}/items"

-H "X-Directo-Key: YOUR_API_KEY"

-H "Accept: application/xml"
```

Opis API (openapi.json)

Każdy klucz API ma czytelny maszynowo opis API w formacie OpenAPI 3.0:

```
curl -H "X-Directo-Key: YOUR_API_KEY_HERE" \
     https://login.directo.ee/apidirect/v1/openapi.json
```

Opis jest **zależny od klucza**: zawiera tylko zasoby i pola skonfigurowane dla danego klucza API w Directo. Jeśli konfiguracja klucza się zmieni, zmieni się również opis.

Co zawiera opis

Sekcja	Zawartość
`paths`	Zasoby włączone dla klucza (np. `/v1/items`). Wszystkie są zapytaniami GET.
`parameters`	Pola, według których można filtrować. Wszystkie mają typ `string`, ponieważ wartością jest wyrażenie filtra (np. `>2025-01-01`).
`responses → 200`	Pola odpowiedzi i ich typy danych (string, number, integer, boolean, date-time), zarówno w JSON, jak i w XML.
`responses → 400/403/500`	Możliwe komunikaty błędów wraz z przykładami.
`securitySchemes`	Uwierzytelnianie za pomocą nagłówka `X-Directo-Key`.

Nazewnictwo parametrów filtrów

Prefiks	Znaczenie	Przykład
(brak)	Pole nagłówka / rekordu głównego	`/invoices?customercode=K001`
`row_`	Pole wiersza dokumentu	`/invoices?row_item=9810`

Prefiks	Znaczenie	Przykład
`data_row_code`, `data_row_content`, `data_row_param`	Pole danych dodatkowych (`datafields`)	`/items?data_row_code=COLOR`
`row_sumdiscount_`, `row_campaign_`	Wiersze rabatu kwotowego i kampanii formuły cenowej	`/priceformulas?row_campaign_campaign=SUMMER`

Uwaga: Można filtrować tylko według pól, które są wybrane do wyświetlenia (output) dla danego klucza. W przeciwnym razie API zwraca `Filter „x“ not selected to output`. Dozwolone filtry są wymienione w sekcji `parameters` w `openapi.json`.

Jak z tego korzystać

- **Postman / Insomnia**: Import → Link. Automatycznie generuje kompletną kolekcję zapytań.
- **Swagger UI / Swagger Editor**: przeglądanie i testowanie API w przeglądarce.
- **Generatory kodu** (np. openapi-generator): generowanie bibliotek klienckich.
- **Agenci AI i serwery MCP**: automatyczny opis dostępnych narzędzi.



Opis pokazuje, jakie dane można pobrać za pomocą klucza. Traktuj go z taką samą ostrożnością jak sam klucz: nie publikuj go i zawsze przesyłaj klucz w nagłówku, nigdy w adresie URL.

Zaawansowane filtrowanie (kluczowa funkcja)

Filtrowanie w stylu ERP – siła Directo API

KLUCZOWA FUNKCJA: Directo API obsługuje dokładnie te same zaawansowane możliwości filtrowania co sam system Directo ERP.

To jedna z najpotężniejszych funkcji Directo API. Filtry można stosować tak samo jak domyślne filtry w systemie ERP. Do filtrowania można używać wszystkich pól z wyjątkiem pól wyliczanych.

Co to oznacza dla programistów

- **Płynna integracja:** Każdy filtr działający w interfejsie Directo ERP można zastosować przez API za pomocą parametrów zapytania
- **Brak krzywej uczenia się:** Jeśli umiesz filtrować w ERP, wiesz już, jak filtrować przez API
- **Maksymalna elastyczność:** Filtruj według DOWOLNEGO pola z danych odpowiedzi (z wyjątkiem pól wyliczanych)
- **Zaawansowane zapytania:** Łącz wiele filtrów, aby tworzyć złożone zapytania bez niestandardowych endpointów

Pola filtrowalne a pola wyliczane

 Pola filtrowalne (przechowywane w bazie danych) Są to pola przechowywane bezpośrednio w bazie danych, których można używać do filtrowania:

- **Pola artykułu:** code, name, class, status, country, unit, price, cost, supplier_code, barcode, weight, volume
- **Pola zamówienia/faktury:** number, date, status, customer_code, customer_name, total, currency, payment_terms, delivery_date, warehouse
- **Pola klienta:** code, name, country, city, address, phone, email, vat_number, payment_terms, credit_limit
- **Pola magazynowe:** item_code, warehouse, quantity, reserved_quantity, available_quantity, location, batch_number, serial_number
- **Pola wierszy dokumentu:** row_item, row_quantity, row_price, row_description, row_discount, row_warehouse
- **Pola danych dodatkowych:** Wszelkie pola niestandardowe zdefiniowane w systemie ERP (znajdują się w tablicy datafields)

 Pola wyliczane (nie można ich filtrować) Są to pola obliczane w locie, których nie można używać do filtrowania:

- **Wyliczane sumy:** Pola takie jak total_with_vat, line_total, discount_amount (gdy są obliczane z innych pól)
- **Wartości pochodne:** available_stock (gdy jest obliczane jako quantity - reserved), profit_margin (obliczane z ceny i kosztu)
- **Agregacje:** total_orders, average_price, sum_quantity (gdy są agregowane z wielu rekordów)
- **Wartości sformatowane:** Pola tylko do wyświetlania, takie jak formatted_date, formatted_price (gdy są jedynie sformatowaną postacią innych pól)
- **Pola wirtualne:** Pola, które istnieją tylko w odpowiedzi API, a nie w bazie danych

Jak rozpoznać pola wyliczane

- **Sprawdź interfejs ERP:** Jeśli nie możesz filtrować według danego pola w oknie filtra systemu ERP, prawdopodobnie jest ono wyliczane
- **Przetestuj filtr:** Spróbuj filtrować według pola przez API – jeśli nie ma wyników lub pojawia się błąd, pole jest wyliczane
- **Szukaj wzorców:** Pola o nazwach takich jak total_*, calculated_*, formatted_* lub display_* są często wyliczane
- **Sprawdź dokumentację:** Zajrzyj do dokumentacji pól systemu ERP lub zapytaj administratora

Przykłady praktyczne: pola filtrowalne a wyliczane

Przykład 1: Stan artykułu

```
 DZIAŁA: GET /items?quantity=>100
(quantity jest przechowywane w bazie danych)
 NIE DZIAŁA: GET /items?available_stock=>100
(jeśli available_stock = quantity - reserved, pole jest wyliczane)
```

Przykład 2: Sumy zamówień

```
 DZIAŁA: GET /orders?row_price=>1000
(row_price jest przechowywane dla każdego wiersza)
 MOŻE NIE DZIAŁAĆ: GET /orders?order_total=>5000
(jeśli order_total jest sumą wszystkich wierszy, pole jest wyliczane)
 ALTERNATYWA: GET /orders?date=>2025-01-01&status=COMPLETED
(następnie oblicz sumy w swojej aplikacji)
```

Przykład 3: Dane klienta

```
 DZIAŁA: GET /customers?country=EE&city=Tallinn
(oba pola są przechowywane)
 NIE DZIAŁA: GET /customers?full_address=*Tallinn*
(jeśli full_address jest połączeniem address+city+country)
 ALTERNATYWA: GET /customers?city=Tallinn
(filtruj według poszczególnych przechowywanych pól)
```

Filtrowanie podstawowe

Filtruj według dokładnego dopasowania, używając nazwy dowolnego przechowywanego pola:

```
GET {{baseUrl}}/v{{version}}/items?code=9810
GET {{baseUrl}}/v{{version}}/items?status=UUS
GET {{baseUrl}}/v{{version}}/orders?number=12345
GET {{baseUrl}}/v{{version}}/customers?country=EE
GET {{baseUrl}}/v{{version}}/items?class=KAUP
```

Kluczowa zasada: Używaj dokładnej nazwy pola, takiej jak w odpowiedzi API (lub takiej, jaką widzisz w systemie ERP).

Filtrowanie „zawiera”

Ten rodzaj filtrowania pozwala wyszukiwać wartości zawierające określony fragment tekstu.

Działa to jak logika SQL `LIKE '%text%'` . Używaj go, gdy potrzebujesz elastycznego filtrowania nazw, kodów, opisów lub innych pól tekstowych.

```
GET {{baseUrl}}/v{{version}}/customers?name=%john%
```

Operatory porównania

API obsługuje zaawansowane operatory porównania:

Większe niż (>)

```
GET {{baseUrl}}/v{{version}}/orders?date=>2025-09-01T00:00:00
```

```
GET {{baseUrl}}/v{{version}}/invoices?row_quantity=>5000
```

Mniejsze niż (<)

```
GET {{baseUrl}}/v{{version}}/orders?date=<2025-09-03T00:00:00
```

```
GET {{baseUrl}}/v{{version}}/invoices?row_quantity=<5000
```

Różne od (!)

```
GET {{baseUrl}}/v{{version}}/items?country=!EE
```

```
GET {{baseUrl}}/v{{version}}/orders?status=!CLOSED
```

Większe lub równe (>=)

```
GET {{baseUrl}}/v{{version}}/items?price=>100
```

```
GET {{baseUrl}}/v{{version}}/orders?date=>2025-01-01T00:00:00
```

Mniejsze lub równe (< =)

```
GET {{baseUrl}}/v{{version}}/items?weight=<5.5
```

```
GET {{baseUrl}}/v{{version}}/orders?date=<2025-12-31T23:59:59
```

Filtrowanie zakresu

Łącz operatory, aby tworzyć zakresy (logika AND):

```
GET
{{baseUrl}}/v{{version}}/orders?date=>2025-09-01T00:00:00&date=<2025-09-03T0
0:00:00
GET {{baseUrl}}/v{{version}}/items?quantity=>10&quantity=<100
```

Wiele wartości (logika OR)

Używaj list rozdzielonych przecinkami, aby filtrować według wielu wartości:

```
GET {{baseUrl}}/v{{version}}/items?warehouse=AALTR,AIPM,WAREHOUSE1
GET {{baseUrl}}/v{{version}}/orders?status=NEW,PENDING,PROCESSING
GET {{baseUrl}}/v{{version}}/customers?country=EE,LV,LT
```

Filtrowanie według znacznika czasu (synchronizacja)

Użyj parametru ts, aby pobrać rekordy zmodyfikowane po określonym znaczniku czasu (idealne do

synchronizacji):

```
GET {{baseUrl}}/v{{version}}/items?ts=>2025-10-01T08:11:05.129Z
GET {{baseUrl}}/v{{version}}/customers?ts=>2025-01-01T00:00:00Z
```

Dobra praktyka: Zapisz znacznik czasu ostatniej udanej synchronizacji i użyj go w następnym zapytaniu, aby pobrać tylko zmienione rekordy.

Filtrowanie według pól wierszy

W zasobach z wierszami (zamówienia, faktury) można filtrować według pól na poziomie wiersza:

```
GET {{baseUrl}}/v{{version}}/orders?row_item=343443
GET {{baseUrl}}/v{{version}}/invoices?row_quantity=>10
```

Uwaga: Przy filtrowaniu według pól wierszy API zwraca cały dokument (zamówienie/fakturę), jeśli KTÓRYKOLWIEK wiersz pasuje do filtra.

Filtrowanie według pól danych dodatkowych

Jeśli w systemie ERP zdefiniowano pola niestandardowe (widoczne w tablicy datafields), można według nich filtrować:

```
GET {{baseUrl}}/v{{version}}/items?data_row_code=ART_PROJ
GET {{baseUrl}}/v{{version}}/items?data_row_code=ART_PROJ,BRAND
```

Jak znaleźć nazwy pól niestandardowych: Wykonaj zapytanie bez filtrów i sprawdź tablicę datafields w odpowiedzi.

Złożone kombinacje filtrów

Łącz wiele filtrów, aby tworzyć zaawansowane zapytania:

Pobierz nowe zamówienia z ostatniego tygodnia dla konkretnego klienta:

```
GET
{{baseUrl}}/v{{version}}/orders?status=NEW&date=>2025-11-13T00:00:00&custome
r_code=CUST001
```

Pobierz artykuły o niskim stanie w wybranych magazynach, z wyłączeniem zamkniętych artykułów:

```
GET
{{baseUrl}}/v{{version}}/items?quantity=<10&warehouse=AALTR,AIPM&status=!CLO
SED
```

Popularne parametry filtrów według zasobu

Parametr	Opis	Przykład	Zasób
ts	Znacznik czasu zmodyfikowanych rekordów (ISO 8601)	2025-10-01T08:11:05.129Z	Wszystkie zasoby
date	Filtrowanie po dacie (obsługuje operatory)	>2025-09-01T00:00:00	Zamówienia, Faktury
number	Numer dokumentu	12345 lub !001	Zamówienia, Faktury
code	Kod artykułu/klienta	9810 lub CUST001	Artykuły, Klienci
name	Wyszukiwanie po nazwie	Kalev	Artykuły, Klienci
status	Filtr statusu	UUS, CLOSED, NEW	Zamówienia, Artykuły
country	Kod kraju (ISO, 2 litery)	EE, !EE	Artykuły, Klienci
warehouse	Kod magazynu/lokalizacji	AALTR,AIPM	Artykuły, Stany magazynowe, Zamówienia
quantity	Filtr ilości (obsługuje operatory)	>100, <50	Artykuły, Stany magazynowe
row_item	Kod artykułu w wierszach	9810 lub PROD-ABC	Zamówienia, Faktury
closed	Status zamknięcia	1 (zamknięty), 0 (otwarty)	Klienci, Zamówienia

Dobre praktyki filtrowania

- **Zacznij prosto:** Rozpocznij od pojedynczych filtrów i dodawaj kolejne w miarę potrzeb
- **Synchronizuj po znaczniku czasu:** Do synchronizacji przyrostowej zawsze używaj parametru ts
- **Łącz strategicznie:** Używaj wielu filtrów, aby ograniczyć transfer i przetwarzanie danych
- **Najpierw przetestuj w ERP:** Jeśli nie masz pewności, czy pole jest filtrowalne, najpierw sprawdź to w interfejsie ERP
- **Obsługuj puste wyniki:** Pusty wynik może oznaczać, że filtry są zbyt restrykcyjne lub pole jest wyliczane
- **Kodowanie URL:** Zawsze koduj wartości filtrów (URL-encode), zwłaszcza daty i znaki specjalne

Rozwiązywanie problemów z filtrami

Problem: filtr nie zwraca wyników

- Sprawdź: czy pole jest wyliczane? Spróbuj filtrować według bazowych pól przechowywanych
- Sprawdź: czy wartość jest poprawna? Zweryfikuj dokładne wartości pól w odpowiedzi bez filtrów
- Sprawdź: czy operatory są poprawne? Użyj > zamiast >= jeśli tak działa ERP

Problem: filtr wydaje się ignorowany

- Sprawdź: czy nazwa pola jest poprawnie zapisana? W nazwach pól rozróżniana jest wielkość liter
- Sprawdź: czy wartość jest zakodowana w URL? Znaki specjalne muszą być zakodowane
- Sprawdź: czy pole istnieje w tym zasobie? Nie wszystkie pola są dostępne we wszystkich zasobach

Problem: nieoczekiwane wyniki

- Sprawdź: czy używasz logiki OR (przecinek), gdy chodziło Ci o AND (wiele parametrów)?
- Sprawdź: w przypadku filtrów wierszy pamiętaj, że zwracany jest cały dokument, jeśli pasuje którykolwiek wiersz

API Resources & Endpoints

Zasoby i endpointy API znajdziesz w kolekcji Postman. Lista pól odpowiedzi jest zarządzana w Directo ERP.

```
GET {{baseUrl}}/v{{version}}/openapi.json       # Opis API – włączone zasoby, pola
GET {{baseUrl}}/v{{version}}/sales_quotations   # Oferty sprzedaży – nagłówek i wiersze
GET {{baseUrl}}/v{{version}}/orders             # Zamówienia sprzedaży – nagłówek i wiersze
GET {{baseUrl}}/v{{version}}/invoices           # Faktury sprzedaży – nagłówek i wiersze
GET {{baseUrl}}/v{{version}}/purchase_orders    # Zamówienia zakupu – nagłówek i wiersze
GET {{baseUrl}}/v{{version}}/stock_receipts     # Przyjęcia magazynowe – wiersze przyjętych towarów
GET {{baseUrl}}/v{{version}}/purchase_invoices  # Faktury zakupu – nagłówek i wiersze
GET {{baseUrl}}/v{{version}}/purchase_payments  # Płatności wychodzące – opłacone faktury zakupu
GET {{baseUrl}}/v{{version}}/customers          # Klienci – karta, kontakty, datafields
GET {{baseUrl}}/v{{version}}/suppliers          # Dostawcy – karta, rachunki bankowe
GET {{baseUrl}}/v{{version}}/contacts           # Osoby kontaktowe – kontakty klientów
GET {{baseUrl}}/v{{version}}/items              # Artykuły – karta, ceny, opakowania
GET {{baseUrl}}/v{{version}}/itemclasses        # Klasy artykułów – konta, ustawienia VAT
GET {{baseUrl}}/v{{version}}/priceformulas      # Formuły cenowe – rabaty, kampanie
GET {{baseUrl}}/v{{version}}/stocklevels        # Stany magazynowe – ilość w podziale na magazyny
GET {{baseUrl}}/v{{version}}/stocklevels_sn     # Stany magazynowe – według numeru seryjnego
GET {{baseUrl}}/v{{version}}/allocations        # Alokacje – ilości zarezerwowane i oczekiwane
GET {{baseUrl}}/v{{version}}/accounts           # Plan kont – konta księgi głównej
GET {{baseUrl}}/v{{version}}/transactions       # Transakcje KG – wiersze Wn/Ma
GET {{baseUrl}}/v{{version}}/deleted            # Dziennik usunięć – usunięte rekordy
```

```
GET {{baseUrl}}/v{{version}}/financial_budgets  # Budżety finansowe – wiersze budżetu
GET {{baseUrl}}/v{{version}}/events             # Zdarzenia – zdarzenia i zadania CRM
GET {{baseUrl}}/v{{version}}/receipts           # Płatności przychodzące – opłacone faktury sprzedaży
GET {{baseUrl}}/v{{version}}/deliveries         # Dostawy – wiersze wysłanych towarów
GET {{baseUrl}}/v{{version}}/movements          # Przesunięcia magazynowe – transfery między magazynami
GET {{baseUrl}}/v{{version}}/objects            # Obiekty – wymiary księgowe
GET {{baseUrl}}/v{{version}}/projects           # Projekty – karta i wiersze
GET {{baseUrl}}/v{{version}}/resources          # Zasoby – wpisy pracy i czasu
GET {{baseUrl}}/v{{version}}/sales_contracts    # Umowy sprzedaży – warunki fakturowania cyklicznego
GET {{baseUrl}}/v{{version}}/replacementlogs    # Dziennik zamian – historia zamiany kodów
GET {{baseUrl}}/v{{version}}/users              # Użytkownicy – karty użytkowników i pracowników
GET {{baseUrl}}/v{{version}}/recipes            # Receptury – produkcyjne zestawienia materiałowe
GET {{baseUrl}}/v{{version}}/nettings           # Kompensaty – dokumenty kompensat
GET {{baseUrl}}/v{{version}}/usagelogs          # Dziennik użycia API – historia zapytań
GET {{baseUrl}}/v{{version}}/expenses           # Raporty wydatków – wydatki pracowników
GET {{baseUrl}}/v{{version}}/approvals          # Zatwierdzenia – obieg zatwierdzania dokumentów
GET {{baseUrl}}/v{{version}}/absences           # Nieobecności – ewidencja nieobecności pracowników
GET {{baseUrl}}/v{{version}}/assets             # Majątek – urządzenia i wyposażenie
GET {{baseUrl}}/v{{version}}/customer_assets    # Majątek klienta – urządzenia należące do klienta
GET {{baseUrl}}/v{{version}}/stocks             # Magazyny – rejestr i ustawienia magazynów
GET {{baseUrl}}/v{{version}}/wage_calculations  # Naliczenia płac – wyliczone wynagrodzenia pracowników
```

Ta lista ma charakter ogólny. Zasoby dostępne dla Twojego klucza są wymienione w openapi.json.

Kolekcję Postman można pobrać tutaj: POSTMAN

Struktura odpowiedzi

Odpowiedź poprawna (HTTP 200)

Format JSON:

```
[
{
"code": "1011",
"class": "KAUP",
"name": "Kommid Lily, Kalev",
"price": 12.50,
"quantity": 150,
"warehouse": "AALTR",
"datafields": [
{
"code": "TEST_ARTIKKEL",
"content": "Test lisaväli sisu"
},
{
"code": "IS_WEIGHT",
"content": "0"
},
{
"code": "VISUAL_VERIFICATION",
"content": "0"
},
{
"code": "VENSAFE",
"content": "0"
}
]
},
{
"code": "1012",
"class": "KAUP",
"name": "Šokolaad Kalev",
"price": 8.75,
"quantity": 200,
"warehouse": "AIPM",
"datafields": []
}
]
```

Format XML:

```
<items>
<item code="1011" class="KAUP" name="Kommid Lily, Kalev" price="12.50"
quantity="150" warehouse="AALTR">
<datafields>
<data code="TEST_ARTIKKEL" content="Test lisaväli sisu"/>
<data code="IS_WEIGHT" content="0"/>
```

```
<data code="VISUAL_VERIFICATION" content="0"/>
<data code="VENSAFE" content="0"/>
</datafields>
</item>
<item code="1012" class="KAUP" name="Šokolaad Kalev" price="8.75"
quantity="200" warehouse="AIPM">
<datafields/>
</item>
</items>
```

Cechy odpowiedzi

- **Format tablicy:** Wszystkie endpointy zwracają tablice obiektów, nawet dla pojedynczych wyników
- **Spójna struktura:** Nazwy pól są takie same we wszystkich formatach odpowiedzi
- **Pola niestandardowe:** Pola danych dodatkowych pojawiają się w tablicy datafields
- **Wartości null:** Brakujące lub puste (null) wartości mogą być pomijane w odpowiedziach
- **Format daty:** Wszystkie daty są w formacie ISO 8601 (np. 2025-11-20T07:30:00Z)
- **Wartości liczbowe:** Liczby są zwracane jako liczby w JSON i jako ciągi znaków w XML

⚠ Obsługa błędów

Kody statusu HTTP

Kod statusu	Znaczenie	Opis
200	OK	Zapytanie zakończone powodzeniem
400	Bad Request	Nieprawidłowe parametry lub błędnie sformułowane zapytanie
400	API not enababled	Zasób nie jest włączony dla tego klucza API
401	Unauthorized	Brak klucza API lub klucz jest nieprawidłowy
403	Forbidden	Klucz API nie ma wymaganych uprawnień
404	Not Found	Nie znaleziono zasobu lub endpointu
429	Too Many Requests	Przekroczono limit zapytań
500	Internal Server Error	Błąd po stronie serwera
503	Service Unavailable	API jest tymczasowo niedostępne

Format odpowiedzi błędu

```
{
"error": {
"code": "INVALID_API_KEY",
"message": "The provided API key is invalid or expired",
"details": "Please check your API key and try again"
}
}
```

Typowe scenariusze błędów

401 Unauthorized:

- Brak nagłówka X-Directo-Key
- Nieprawidłowy lub wygasły klucz API
- **Rozwiązanie:** Sprawdź klucz API i upewnij się, że jest dołączony do nagłówka

403 Forbidden:

- Klucz API nie ma uprawnień do żądanego zasobu
- **Rozwiązanie:** Skontaktuj się z administratorem, aby nadał niezbędne uprawnienia

400 Bad Request:

- Nieprawidłowa składnia filtra
- Błędny format daty (użyj ISO 8601: 2025-11-20T07:30:00Z)
- Filtrowanie według pola wyliczanego
- **Rozwiązanie:** Sprawdź parametry filtrów i formaty dat

404 Not Found:

- Nieprawidłowy URL endpointu
- Nieprawidłowa wersja API
- **Rozwiązanie:** Sprawdź ścieżkę endpointu i numer wersji

429 Too Many Requests:

- Przekroczono limit zapytań
- **Rozwiązanie:** Zastosuj wykładnicze wydłużanie odstępów (exponential backoff) i zmniejsz częstotliwość zapytań

500 Internal Server Error:

- Błąd przetwarzania po stronie serwera
- **Rozwiązanie:** Ponów próbę po pewnym czasie; jeśli problem się utrzymuje, skontaktuj się z pomocą techniczną

Limity zapytań i wydajność

Limity zapytań

Konkretne limity zapytań zależą od poziomu Twojego konta, ale stosuj się do poniższych wskazówek:

- **Zalecane:** maks. 60 zapytań na minutę
- **Limit chwilowy (burst):** dopuszczalne mogą być krótkie skoki większego ruchu
- **Limity dzienne:** Mogą obowiązywać w zależności od subskrypcji

Dobre praktyki wydajnościowe

1. Używaj filtrowania po znaczniku czasu do synchronizacji Zamiast wielokrotnie pobierać wszystkie rekordy, użyj parametru ts:

```
GET {{baseUrl}}/v{{version}}/items?ts>=2025-11-20T07:00:00Z
```

* **Korzyści:** mniejszy transfer danych, krótszy czas odpowiedzi, mniejsze obciążenie serwera, mieszczenie się w limitach zapytań

2. Stosuj stronicowanie według zakresów dat W przypadku dużych zbiorów danych dziel zapytania na mniejsze części:

```
# Tydzień 1

GET {{baseUrl}}/v{{version}}/orders?date=>2025-11-01&date=<2025-11-08

# Tydzień 2

GET {{baseUrl}}/v{{version}}/orders?date=>2025-11-08&date=<2025-11-15
```

3. Buforuj dane statyczne Buforuj dane, które zmieniają się rzadko:

- Klasy artykułów (bufor na 24 godziny)
- Formuły cenowe (bufor na 1 godzinę)
- Dane podstawowe klientów (bufor na 6 godzin)

4. Używaj precyzyjnych filtrów Stosuj filtry, aby pobierać tylko potrzebne dane:

```
# Zamiast pobierać wszystkie artykuły i filtrować je lokalnie

GET {{baseUrl}}/v{{version}}/items

# Pobieraj tylko to, czego potrzebujesz

GET {{baseUrl}}/v{{version}}/items?warehouse=AALTR&quantity=>0
```

5. Stosuj wykładnicze wydłużanie odstępów (exponential backoff) W przypadku błędów przejściowych (429, 503) ponawiaj próby z coraz dłuższymi opóźnieniami:

- 1. ponowienie: odczekaj 1 sekundę
- 2. ponowienie: odczekaj 2 sekundy
- 3. ponowienie: odczekaj 4 sekundy
- 4. ponowienie: odczekaj 8 sekund

6. Grupuj powiązane zapytania Grupuj powiązane operacje, aby zminimalizować liczbę wywołań:

```
1. Pobierz zamówienia: ''GET /orders?ts=>{last_sync}''
2. Dla każdego zamówienia pobierz dane klienta (jeśli są potrzebne)
3. Zaktualizuj lokalną bazę danych w trybie wsadowym
```

7. Monitoruj użycie API Śledź wykorzystanie API:

- Rejestruj liczbę zapytań dla każdego endpointu
- Monitoruj czasy odpowiedzi
- Śledź współczynnik błędów
- Skonfiguruj alerty ostrzegające o zbliżaniu się do limitu zapytań

Typowe wzorce integracji

Wzorzec 1: Początkowa synchronizacja danych (pierwsza konfiguracja)

Cel: Załaduj wszystkie dane z Directo ERP do swojego systemu

- **Krok 1:** Pobierz wszystkie artykuły `GET /items`
- **Krok 2:** Pobierz wszystkich klientów `GET /customers`
- **Krok 3:** Pobierz wszystkie stany magazynowe `GET /stocklevels`
- **Krok 4:** Pobierz klasy artykułów `GET /itemclasses`
- **Krok 5:** Zapisz bieżący znacznik czasu na potrzeby przyszłych synchronizacji przyrostowych `timestamp = current_time()`

Uwagi: Może potrwać kilka minut – uruchamiaj poza godzinami szczytu i obsługuj przekroczenia czasu (timeout).

Wzorzec 2: Synchronizacja przyrostowa (bieżąca)

Cel: Utrzymuj swój system w zgodzie ze zmianami w Directo ERP

- **Krok 1:** Odczytaj znacznik czasu ostatniej synchronizacji ze swojej bazy danych
- **Krok 2:** Pobierz zmienione artykuły `GET /items?ts⇒{last_sync}`
- **Krok 3:** Pobierz zmienionych klientów `GET /customers?ts⇒{last_sync}`
- **Krok 4:** Pobierz zmienione stany magazynowe `GET /stocklevels?ts⇒{last_sync}`
- **Krok 5:** Pobierz usunięte rekordy `GET /deleted?ts⇒{last_sync}`
- **Krok 6:** Przetwórz zmiany (aktualizacja/dodanie/usunięcie)
- **Krok 7:** Zaktualizuj znacznik czasu ostatniej synchronizacji

Wzorzec 3: Monitorowanie zamówień w czasie rzeczywistym

Cel: Monitoruj nowe zamówienia w celu natychmiastowego przetwarzania

- **Krok 1:** Odpytuj endpoint zamówień `GET /orders?status=NEW&ts⇒{last_check}`
- **Krok 2:** Przetwórz nowe zamówienia (walidacja, sprawdzenie stanów, aktualizacja statusu)
- **Krok 3:** Zaktualizuj znacznik czasu ostatniego sprawdzenia
- **Krok 4:** Poczekaj do następnego interwału (np. 5 minut)

Wzorzec 4: Zarządzanie zapasami i sprawdzanie dostępności

Cel: Utrzymuj dokładne informacje o dostępności zapasów

- **Krok 1:** Pobierz bieżące stany magazynowe GET `/stocklevels?item_code={sku}`
- **Krok 2:** Pobierz alokacje (zarezerwowany zapas) GET `/allocations?item_code={sku}`
- **Krok 3:** Oblicz dostępną ilość: `available = stock_quantity - allocated_quantity`
- **Krok 4:** Zaktualizuj swój system magazynowy

Wzorzec 5: Synchronizacja danych klientów (integracja z CRM)

Cel: Utrzymuj synchronizację danych klientów między Directo a systemem CRM

- **Krok 1:** Pobierz zmienionych klientów GET `/customers?ts⇒{last_sync}`
- **Krok 2:** Synchronizuj z CRM (utworzenie/aktualizacja)
- **Krok 3:** Pobierz usuniętych klientów GET `/deleted?document=klient&ts⇒{last_sync}`
- **Krok 4:** Usuń usuniętych klientów z CRM

Wzorzec 6: Eksport faktur do systemu księgowego

Cel: Eksportuj faktury do zewnętrznego oprogramowania księgowego

- **Krok 1:** Pobierz nowe/zaktualizowane faktury GET `/invoices?ts⇒{last_sync}`
- **Krok 2:** Zmapuj pola i utwórz dokumenty w systemie księgowym
- **Krok 3:** Obsłuż błędy i oznacz jako wyeksportowane

Wzorzec 7: Synchronizacja cen ze sklepem internetowym

Cel: Utrzymuj aktualne ceny produktów w sklepie internetowym

- **Krok 1:** Pobierz zmienione artykuły wraz z cenami GET `/items?ts⇒{last_sync}`
- **Krok 2:** Pobierz formuły cenowe GET `/priceformulas?ts⇒{last_sync}`
- **Krok 3:** Oblicz ceny końcowe i zaktualizuj platformę e-commerce

Przewodnik rozwiązywania problemów

Problem: pusta odpowiedź / brak wyników

Możliwe przyczyny: zbyt restrykcyjne filtry, nieprawidłowy format daty, filtrowanie według pola wyliczanego. **Rozwiązania:**

```
# Usuwaj filtry jeden po drugim

GET /items

# Sprawdź format daty
```

```
 POPRAWNIE: ?date=>2025-11-20T00:00:00Z
```

Problem: błędy uwierzytelniania (401/403)

Możliwe przyczyny: brak klucza, nieprawidłowa nazwa nagłówka, wygasły klucz. **Rozwiązania:**

```
# Sprawdź nazwę nagłówka (wielkość liter ma znaczenie)

 POPRAWNIE: X-Directo-Key: YOUR_KEY
```

Problem: błędy limitu zapytań (429)

Rozwiązania (Python):

```
import time
def make_request_with_retry(url, max_retries=5):
for attempt in range(max_retries):
response = requests.get(url)
if response.status_code == 200:
return response
elif response.status_code == 429:
wait_time = 2 ** attempt
time.sleep(wait_time)
raise Exception("Max retries exceeded")
```

Problem: `usagelogs` – brak starszych wpisów

Możliwa przyczyna: Okres przechowywania logów jest ograniczony ustawieniami administratora („Log age in days“ / „Log age in transactions“ — zob. Ustawienie administratora, a nie przez samo API.

Rozwiązanie: Poproś administratora o sprawdzenie/zwiększenie tych dwóch ustawień lub zacznij regularnie pobierać i archiwizować `/usagelogs`, aby prowadzić własną historię.

Wsparcie i zasoby

Wsparcie techniczne:

- E-mail: info@directo.pl
- Skontaktuj się ze swoim opiekunem klienta w Directo
- Zgłoś problem przez Directo Chat

Historia zmian i wersji

Wersja 1 (bieżąca)

- Pierwsze wydanie API
- Obsługa głównych zasobów: artykuły, klienci, zamówienia, faktury, stany magazynowe
- Obsługa JSON/XML i filtrowanie w stylu ERP
- dodano opis API (openapi.json) 23.09.2026

Bezpieczeństwo i zgodność

Bezpieczeństwo danych

- **Szyfrowanie**: wszystkie zapytania API muszą korzystać z HTTPS
- **Uwierzytelnianie**: klucz API jest wymagany we wszystkich zapytaniach
- **Autoryzacja**: kontrola dostępu oparta na rolach za pomocą kluczy API
- **Dziennik audytu**: wszystkie zapytania API są rejestrowane po stronie serwera

Kwestie zgodności

- RODO (GDPR): dane klientów zawierają dane osobowe – przetwarzaj je zgodnie z wymogami RODO
- Przechowywanie danych: wdróż odpowiednie zasady przechowywania danych
- Kontrola dostępu: udostępniaj klucze API wyłącznie upoważnionym osobom
- Monitorowanie: monitoruj użycie API pod kątem nietypowych wzorców

Dobre praktyki bezpieczeństwa

- Chroń klucze API:
  - Nigdy nie zapisuj kluczy w systemie kontroli wersji
  - Używaj zmiennych środowiskowych lub bezpiecznych sejfów (vault)
  - Regularnie zmieniaj klucze (zalecane co kwartał)
  - Używaj różnych kluczy dla środowisk dev/staging/production
- Wdróż kontrolę dostępu:
  - Ogranicz uprawnienia klucza API do niezbędnego minimum
  - Używaj osobnych kluczy dla różnych integracji
  - Natychmiast unieważniaj klucze, które nie są już potrzebne
- Monitoruj użycie:
  - Rejestruj wszystkie zapytania API w swojej aplikacji
  - Skonfiguruj alerty o nietypowej aktywności
  - Regularnie przeglądaj logi dostępu
- Odpowiedzialnie przetwarzaj dane:
  - Szyfruj wrażliwe dane w spoczynku
  - Wdróż zasady przechowywania danych
  - Przestrzegaj przepisów o ochronie prywatności (RODO itp.)
  - Zabezpieczaj transmisję danych (wyłącznie HTTPS)

Wskazówki i triki

- **Wskazówka 1:** Poznaj dostępne pola, wykonując zapytanie bez filtrów: GET `/items?code=1011`
- **Wskazówka 2:** Najpierw testuj filtry w interfejsie ERP.
- **Wskazówka 3:** Twórz ciągi filtrów programowo.
- **Wskazówka 4:** Użyj openapi.json, aby poznać dostępne pola i filtry. Jest to szybsze niż zapytanie bez filtrów, nie pobiera danych i zawiera typy pól.

Przykład (Python):

```
# Przykład w Pythonie z pełną obsługą błędów
import requests
import time
def fetch_directo_data(endpoint, params=None, max_retries=3):
    url = f"{{{{baseUrl}}}}/v{{{{version}}}}/{endpoint}"
    headers = {
        "X-Directo-Key": "YOUR_API_KEY",
        "Accept": "application/json"
    }
    for attempt in range(max_retries):
        try:
            response = requests.get(url, headers=headers, params=params,
timeout=30)
            if response.status_code == 200:
                return response.json()
            elif response.status_code == 429:
                # Limit zapytań – odczekaj i ponów
                wait_time = 2 ** attempt
                time.sleep(wait_time)
            elif response.status_code in [401, 403]:
                # Błąd uwierzytelniania – nie ponawiaj
                raise Exception(f"Authentication error:
{response.status_code}")
            else:
                # Inny błąd – zarejestruj i ponów
                print(f"Error {response.status_code}: {response.text}")
                time.sleep(1)
        except requests.exceptions.Timeout:
            print(f"Timeout on attempt {attempt + 1}")
            time.sleep(2)
        except requests.exceptions.RequestException as e:
            print(f"Request failed: {e}")
            time.sleep(2)
    raise Exception(f"Failed after {max_retries} attempts")
# Użycie
items = fetch_directo_data("items", params={"warehouse": "AALTR",
"quantity": ">0"})
```

Ostatnia aktualizacja: 29-01-2026 **Wersja API:** 1

Szybka ściągawka

Najważniejsze endpointy

```
Opis API:       GET /v1/openapi.json
Artykuły:       GET /v1/items
Klienci:        GET /v1/customers
Zamówienia:     GET /v1/orders
Faktury:        GET /v1/invoices
Stany mag.:     GET /v1/stocklevels
Usunięcia:      GET /v1/deleted
```

Uwierzytelnianie

```
X-Directo-Key: YOUR_API_KEY
```

Format odpowiedzi

```
Accept: application/json  (lub application/xml)
```

Popularne filtry

```
Dokładne dopasowanie:      ?field=value
Większe niż:     ?field=>value
Mniejsze niż:        ?field=<value
Różne od:        ?field=!value
Wiele wartości:  ?field=value1,value2
Znacznik czasu:        ?ts=2025-11-20T07:00:00Z
```

Przykłady filtrów

```
?warehouse=AALTR
?quantity=>100
?date=>2025-11-01T00:00:00&date=<2025-12-01T00:00:00
?status=NEW,PENDING
?country=!EE
?ts=2025-11-20T07:00:00Z
```

Pamiętaj

- Można filtrować według wszystkich pól (z wyjątkiem pól wyliczanych)
- Filtry działają tak samo jak w systemie ERP
- Do synchronizacji używaj filtrowania po znaczniku czasu

- Koduj znaki specjalne w URL
- Ogranicz częstotliwość zapytań (60 zapytań/min)
- Obsługuj błędy z wykładniczym wydłużaniem odstępów (exponential backoff)

Tę dokumentację prowadzi zespół programistów Directo. W razie pytań lub uwag pisz na adres info@directo.pl

From:
https://wiki.directo.ee/ - **Directo Help**

Permanent link:
https://wiki.directo.ee/pl/api_direct

Last update: **2026/11/13 16:12**