

V O L V O

# SZYBKĄ IMPLEMENTACJĄ REST API

w oparciu o ILElastic

2025-08-22



# **VOLVO GROUP DIGITAL & IT**

WHO WE ARE | OUR STRATEGIC DIRECTION | WHAT WE DO

# Volvo Group Digital & IT

Volvo Group Digital & IT manages overall digital and IT strategies and plans for the Volvo Group and has the end-to-end responsibility for all development, delivery, and support of IT solutions and services globally.



VOLVO PENTA



VOLVO ENERGY



VOLVO AUTONOMOUS  
SOLUTIONS



VOLVO FINANCIAL  
SERVICES



ARQUUS



MACK TRUCKS



RENAULT TRUCKS



VOLVO TRUCKS



VOLVO CONSTRUCTION  
EQUIPMENT



VOLVO BUSES



GROUP TRUCKS  
TECHNOLOGY



GROUP TRUCKS  
OPERATIONS



GROUP TRUCKS  
PURCHASING

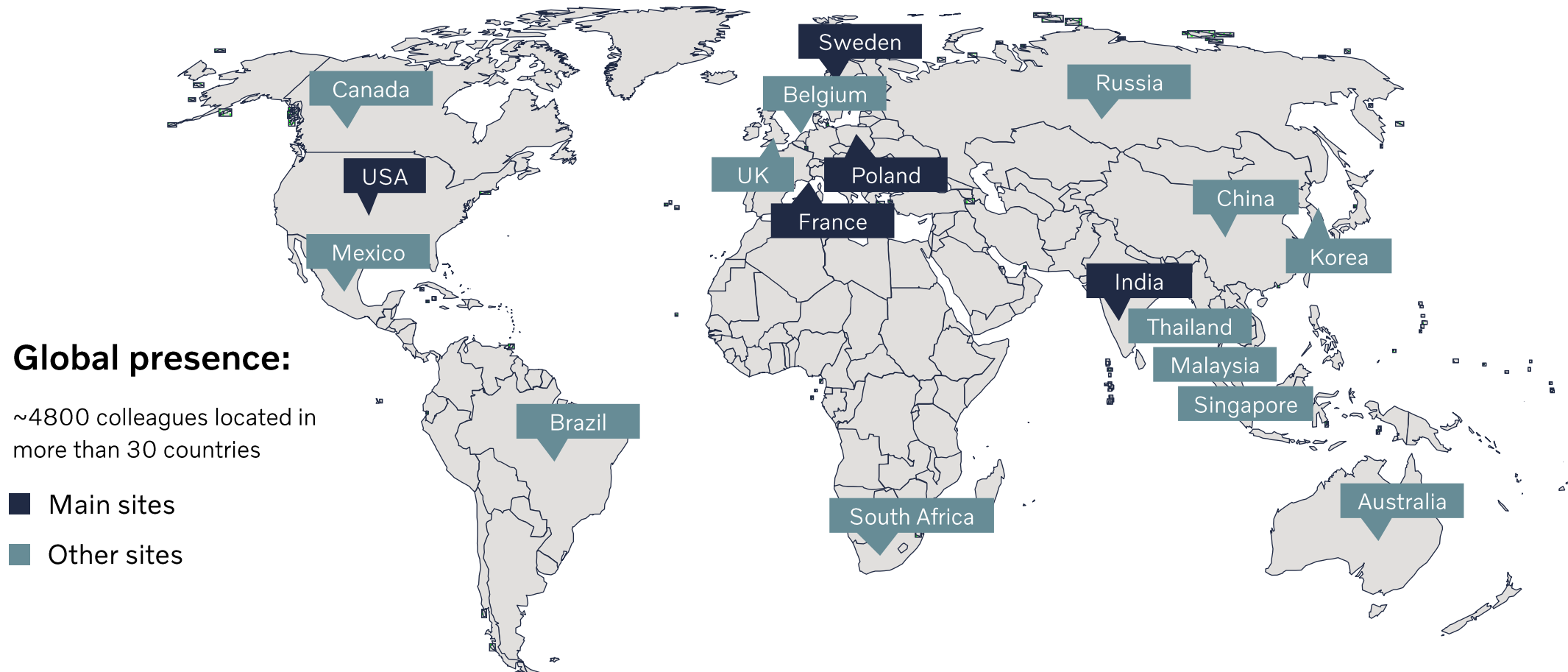


GROUP  
PEOPLE & CULTURE



GROUP  
FINANCE

# We are colleagues from around the globe



# Przygotowanie do tworzenia usług

## REST

- Zorientowanie na zasoby
- Operacje na API w nawiązaniu do metod HTTP

Zasób	POST	GET	PUT	DELETE
/customers	Utwórz nowego klienta	Pobierz wszystkich klientów	Masowa zmiana na klientach	Usuń wszystkich klientów
/customers/1	Błąd	Pobierz szczegóły dla klienta 1	Zmień szczegóły klienta 1 jeżeli istnieje	Usuń klienta 1
/customers/1/orders	Utwórz nowe zlecenie dla klienta 1	Pobierz wszystkie zamówienia dla klienta 1	Masowa zmiana zamówień dla klienta 1	Usuń wszystkie zamówienia klienta 1

- Kody stanów
  - 200(OK), 201(Created), 204(No Content), 400(Bad Request), 404(Not Found), 409(Conflict), 415 (Unsupported Media Type)
- Filtrowanie i stronicowanie
  - Parametry metody GET
  - Limit i offset

/orders?limit=25&offset=50

# Przygotowanie do tworzenia usług

## Specyfikacja OpenAPI

The OpenAPI Specification (OAS) definiuje standard, niezależnego od języka interfejsu do REST API

Specyfikacja może zostać zapisana w JSON lub YAML i definiować elementy:

- Punkty końcowe
- Parametry WE/WY dla operacji
- Metody autentykacji
- Informacje dodatkowe i kontaktowe

```
openapi: 3.0.0
info:
  title: Sample API
  description: Optional multiline or single-line description in [CommonMark](http://commonmark.org/help/) or HTML.
  version: 0.1.9


servers:
  - url: http://api.example.com/v1
    description: Optional server description, e.g. Main (production) server
  - url: http://staging-api.example.com
    description: Optional server description, e.g. Internal staging server for testing

paths:
  /users:
    get:
      summary: Returns a list of users.
      description: Optional extended description in CommonMark or HTML.
      responses:
        '200': # status code
          description: A JSON array of user names
          content:
            application/json:
              schema:
                type: array
                items:
                  type: string
```

# ILEastic

## Projekt


Samodzielny serwer aplikacji dla środowiska ILE na IBMi do uruchamiania mikroserwisów

 Apache-2.0 license

 35 stars

 15 watching

 23 forks

 master ▾

 Commits on Oct 11, 2022

And readme also have static on 44012



NielsLiisberg committed 9 days ago



2130365



Static demo on port 44012



NielsLiisberg committed 9 days ago



f6dfc03



Better noxdb demo



NielsLiisberg committed 9 days ago



0ea46a6



Contributors 8





# ILEastic

Szablon aplikacji

- Opcje programu oraz referencje
- Główna procedura
  - Konfiguracja
  - Dodanie ścieżek
  - Uruchomienie odbiornika
- Procedury dodatkowe
  - Struktury żądania i odpowiedzi
  - Funkcje zapisu odpowiedzi

```
ctl-opt decEdit('0,') datEdit(*YMD.) main(main);
ctl-opt debug(*yes) bndDir('ILEASTIC');
ctl-opt thread(*CONCURRENT);
/include qrppleref,ileastic

dcl-proc main;
  dcl-ds config likeds(il_config);

  config.port = 15300;
  config.host = '*ANY';

  il_addRoute(config : %paddr(helloILEASTIC) : IL_GET: '^/helloILEASTIC$');
  il_listen(config);
end-proc;

dcl-proc helloILEASTIC;
  dcl-pi *n;
    request likeds(IL_REQUEST);
    response likeds(IL_RESPONSE);
  end-pi;
  response.status = 200;
  response.contentType = 'application/json';

  il_responseWrite(response : '{"hello":"world"}');
end-proc;
```



# ILeastic

## Definicja ścieżek

- Ścieżki jako wyrażenia regularne
- Przekazywanie parametrów w ścieżce

```
il_addRoute(config : %paddr(test1) : IL_ANY : '/simpletest/{myId}/list/{myNumber}$');
il_addRoute(config : %paddr(test1) : IL_ANY : '/simpletest/{myId}/list$');
il_addRoute(config : %paddr(test2) : IL_ANY : '/simpletest/{myId}$');
il_addRoute(config : %paddr(test2) : IL_ANY : '/simpletest$');

il_responseWrite(response : 'Test1');
il_responseWrite(response : ' myId:');
il_responseWrite(response : il_getPathParameter (request: 'myId' : '000'));
il_responseWrite(response : ' myNumber:');
il_responseWrite(response : il_getPathParameter (request: 'myNumber' : '9999'));

il_responseWrite(response : 'Test2');
il_responseWrite(response : ' myId:');
il_responseWrite(response : il_getPathParameter (request: 'myId' : '000'));
il_responseWrite(response : ' myNumber:');
il_responseWrite(response : il_getPathParameter (request: 'myNumber' : '9999'));
```

# ILElastic

## Dodatkowe funkcjonalności

- noxDB
- Wtyczki
  - authsystem
  - basicauth
  - cors
  - JSON Web Token (JWT)
    - HS256

```
// noxDB Example
search = il_getParmStr(request : 'search');
pResult = json_sqlResultSet ('-
    select * from qsys2.services_info -
    where service_name like upper(' + strquot('%' + search + '%') + ')');
il_responseWriteStream(response : json_stream( pResult));

// -----
// ILElastic plugin Example
il_addPlugin(config : %paddr(logRouteId) : IL_PREREQUEST);

dcl-proc logRouteId export;
  dcl-pi *n ind;
    request likeds(IL_REQUEST);
    response likeds(IL_RESPONSE);
  end-pi;

  if (request.routeId = *blank);
    il_joblog('No route id.');
- else;
- il_joblog('Route id: %s' : request.routeId);
- endif;
- return *on;

end-proc
```

# IBMi JSON i SQL

## Przydatne funkcje

- **JSON\_OBJECT**

```
VALUES (JSON_OBJECT('first' : 'John', 'last' : 'Doe'));
```

- **JSON\_ARRAY**

```
VALUES (JSON_ARRAY('Washington', 'Jefferson', 'Hamilton'));
```

- **JSON\_QUERY**

```
VALUES JSON_QUERY('{ "id": "701", "name": { "first": "John", "last": "Doe" } }', '$.name');
```

- **JSON\_VALUE**

```
VALUES (JSON_VALUE('{ "id": "987" }', '$.id' RETURNING INTEGER));
```

- **JSON\_TABLE**

```
{ "id": 903, "name" : { "first": "Mary", "last": "Jones" }, "office" : "E-739" }
SELECT t.first, t.last, t.office
FROM emp,
     JSON_TABLE(
       emp.jsondoc,
       'lax $'
       COLUMNS (
         first VARCHAR(10) PATH 'lax $.name.first',
         last VARCHAR(10) PATH 'lax $.name.last',
         office VARCHAR(10) PATH 'lax $.office'
       )
     ) AS t;
```

```
select json_object ('id' value id,
                   'name' value json_object ( 'first' value first_name,
                                              'last' value last_name),
                   'office' value office_number
                   absent on null)
from empdata;

{ "id": 901, "name": { "first": "John", "last": "Doe" }, "office": "E-334" }
```

# OpenAPI ILEastic

- Generator w JavaScript
- Wygenerowane obiekty
  - webapp.rpgle – aplikacja bazowa, zawiera routing.
  - validation.rpgle – zawiera funkcje do walidowania każdego z API.
  - structs.rpgle – szablony struktur.
  - intoStructs.rpgle – wygenerowane procedury do obsługi struktur.
  - outofStructs.rpgle - wygenerowane procedury do obsługi struktur.

V O L V O

THANK YOU