

# Inhaltsverzeichnis

In case of technical questions.....	2
Server access.....	2
Authentication.....	3
OAuth2.....	3
Api Key.....	4
Endpoint for most-recent API versions.....	5
Errors and Exceptions.....	6
Mediation, Transformation and Orchestration.....	6
Classes.....	6
Invoice class.....	7
Invoice.....	7
CalculatedInvoice.....	11
Operations/Endpoints.....	12
Ping.....	13
Validate.....	13
Phive.....	13
pdf.....	15
parse.....	15
Invoice2xml.....	15
Extract.....	15
detach.....	16
combine.....	16
combineXML.....	16
cii2UBL.....	17
xmltohtml.....	17
xmltopdf.....	17
Example PHP Client.....	17
Mustangserver Hello World.....	17
Preparations.....	17
Allowing Client Credentials.....	19
Get access token.....	20
OAuth2 Authentication.....	21
Validation of electronic invoices.....	27
Example C# client.....	29
Converting PDF.....	30
Interactive testing using Postman/Bruno.....	30
Performance Tests with Jmeter.....	33
Terms of service.....	36
Test terms.....	36
Production terms.....	36
Troubleshooting.....	37
Inhouse variant.....	38
Version history.....	41

## In case of technical questions

Please contact

**Jochen Stärk**

[jstaerk@usegroup.de](mailto:jstaerk@usegroup.de)

or

**Martin Wachtveitl**

[mwachtveitl@usegroup.de](mailto:mwachtveitl@usegroup.de)

## Server access

User access to the web interface is possible via <https://api.usegroup.de/>. Terms of service are listed in the chapter Terms of service on page 36.

To register please access <https://api.usegroup.de:9443/devportal/services/configs>

and click "Create Account" link on the bottom left. Select a username, "Proceed to self register", enter the rest of the data and have your email verified.

Afterwards you can login on <https://api.usegroup.de/devportal/>.

You will need it your username in order to log in and in case you want to reset your password, both are *not possible using your email address*. In case you forgot your username feel free to inquire at [info@mustangproject.org](mailto:info@mustangproject.org) using the email address you are requesting the username for. This username does not have anything to do with optional username parameters mentioned below.

After being logged in you need to register your interest, i.e. „subscribe“ to the APIs you require.

If you select the desired Mustangserver version and click on the blue "try out" button (not the link in the navigation) on the next page you should be able to click a "get test key" button.

The screenshot shows the WSO2 API Manager Developer Portal. The left sidebar has links for Overview, Subscriptions, Try Out, Comments, Documentation, and SDKs. The main content area shows the 'Overview' of the 'Mustangserver' API, which is version v0.4.0 by admin. It includes a URL: <https://gw.usegroup.de:8243/mustangserver/v0.4.0>. Below this is a section for 'Business Plans' with four options: Bronze (1000 Requests/min), Gold (5000 Requests/min), Silver (2000 Requests/min), and Unlimited (Unlimited Requests/min). Under 'Comments', there is a link to 'Write A New Comment' and a message stating 'No Comments Yet'.

e.g. when you open the "ping" operation and click "try it out" and "execute" you should get a "pong" response.

You can always change your password on <https://api.usegroup.de/devportal/settings/change-password/>

There is an optional „username“ field for all operations: Please ignore it. It serves as a placeholder where the API management transmits your username, if it were set by any application the value would be overwritten anyway.

## Authentication

### OAuth2

First of all you will have to subscribe to the API. You can manage your subscriptions in the left navigation of the devportal but it's often easiest to subscribe via overview|try out, which also allows you to try it.

Then you will have to enable client credentials in the applications tab,  
<https://api.usegroup.de:9443/devportal/applications> Default Application, Oauth2 tokens. The

procedure is described more detailed in the PHP Client chapter „Allowing Client Credentials“ on page 19 but is generic to all examples and does not apply for PHP only.

Please note that most clients will use the Mustangserver version which had been *selected* in the backend when downloading the OpenAPI definition. Feel free to replace /mustang/<version>/mustang by as described in Endpoint for most-recent API versions on page 5.

## Api Key

In the applications tab, <https://api.usegroup.de:9443/devportal/applications> select Default Application, Oauth2 tokens, Production Keys, API Key. Select according restrictions if required, click Generate Key, select lifetime and click generate. Copy&safely store the generated key.

The screenshot shows the WSO2 API Manager Developer Portal interface. The URL in the browser is https://api.usegroup.de:9443/devportal/applications/e5ca7ceb-5a8f-47fc-af65-462dc72f9034/productionkeys/apikey. The left sidebar has a dark theme with icons for Overview, Production Keys, OAuth2 Tokens, API Key, Sandbox Keys, OAuth2 Tokens, API Key, and Subscriptions. The main content area shows a "DefaultApplication" with "3 Subscriptions". Under "Production API Key", there are "Key Restrictions" with three options: "None" (selected), "IP Addresses", and "HTTP Referrers (Web Sites)". Below this is a "GENERATE KEY" button with the instruction "Use the Generate Key button to generate a self-contained JWT token." A modal dialog titled "Generate API Key" is open, containing a checked checkbox "API Key with infinite validity period" and two buttons: "GENERATE" and "CLOSE".

An API key can be used e.g. within Bruno (pp 30ff)

Within PHP you also might pass just as additional header attribute making your source code look like

```
<?php
require_once(__DIR__ . '/vendor/autoload.php');

$apikey="<your key>";
$config = Swagger\Client\Configuration::getDefaultConfiguration();
$gc=new GuzzleHttp\Client(['headers' => ['apikey' => $apikey]]);

$instance = new Swagger\Client\Api\MustangControllerApi(
    $gc,
    $config
);

try {
    $result = $instance->ping();
    print_r($result);
} catch (Exception $e) {
    http_response_code(500);
    echo 'Exception when calling ErrorControllerApi->handle: ', $e->getMessage(), PHP_EOL;
}
```

## Endpoint for most-recent API versions

If you leave out the version number in the request to the gateway you will always be using the latest recommended (usually the latest) version. In that case e.g. your „ping“ endpoint changes from <https://gw.usegroup.de:8243/mustang/1.0.0/mustang/ping> to <https://gw.usegroup.de:8243/mustang/mustang/ping>.

New versions may be retired as soon as six month after release of the successor. It is possible to always use the latest (more precisely: recommended) version by removing the version from the endpoint. This is an example of the validate endpoint:

```
- - - - - "https://gw.usegroup.de:8243/mustang/v0.5.0/mustang/validate",
+ - - - - "https://gw.usegroup.de:8243/mustang/mustang/validate",
```

## Errors and Exceptions

In case of an exception Mustangserver will return a http status code of 400 and the message of the Exception will be returned in the „message“ field of the according JSON.

```
{  
    "requestUrl": "http://127.0.0.1:8000/mustang/combineXML",  
    "httpCode": 400,  
    "errorCode": "MSE1000:Unbekannte Fehler während der Request Ausführung!",  
    "message": "File is not a valid PDF/A-1 input file"  
}
```

## Mediation, Transformation and Orchestration

The <http://api.usegroup.de/> uses WSO<sup>2</sup> as API management which in turn uses Apache Synapse (<https://synapse.apache.org/>) for mediation/transformation/orchestration. This means that mediation and orchestration can be developed e.g. in WSO<sup>2</sup>‘s Integration Studio (<https://wso2.com/integration/integration-studio/>) and uploaded as XML file. Apart from acting as a load balancer and central authentication this allow to

- override certain states in the process, e.g. implement a timeout after a certain number of seconds
- invoke a chain of operations in only one virtual endpoint, e.g. conversion from plain PDF, parallelly converting invoice data to XML, merging PDF/A and XML and validation thereof and/or
- map any custom specific input- or output parameter to the values used by Mustangserver internally

## Classes

Mustangserver has two important main classes, invoice and calculatedInvoice. Invoice contains tradeparty classes for recipients and senders and item classes which in turn contain instances of product classes.

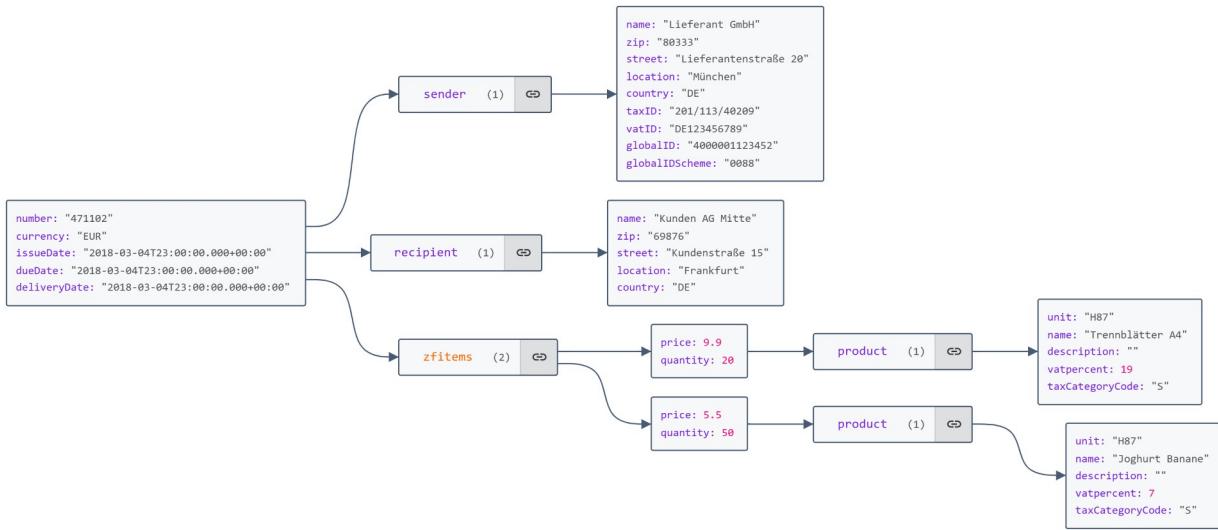
The difference between invoice and calculatedInvoice is that the latter contains (redundant, because calculatable) properties like grandTotal. These attributes are provide for courtesy when reading invoices but are *not required* when writing. However, if they are present when writing (e.g. combine or invoice2XML) they will raise an error if the invoice calculation does not match. This can be useful when a PDF file has been generated with another process and the Factur-X-XML is supposed to have the same values, in which case the PDF values can be provided because an error would be more helpful than a silently incorrect Factur-X file because the PDF has been calculated incorrectly, or uses features not yet available in Mustangserver.

## Invoice class

### *Invoice*

#### Minimal example

```
{  
    "number": "471102",  
    "currency": "EUR",  
    "issueDate": "2018-03-04T00:00:00.000+01:00",  
    "dueDate": "2018-03-04T00:00:00.000+01:00",  
    "deliveryDate": "2018-03-04T00:00:00.000+01:00",  
    "sender": {  
        "name": "Lieferant GmbH",  
        "zip": "80333",  
        "street": "Lieferantenstraße 20",  
        "location": "München",  
        "country": "DE",  
        "taxID": "201/113/40209",  
        "vatID": "DE123456789",  
        "globalID": "4000001123452",  
        "globalIDScheme": "0088"  
    },  
    "recipient": {  
        "name": "Kunden AG Mitte",  
        "zip": "69876",  
        "street": "Kundenstraße 15",  
        "location": "Frankfurt",  
        "country": "DE"  
    },  
    "zfitems": [  
        {  
            "price": 9.9,  
            "quantity": 20,  
            "product": {  
                "unit": "H87",  
                "name": "Trennblätter A4",  
                "description": "",  
                "vatpercent": 19,  
                "taxCategoryCode": "S"  
            }  
        },  
        {  
            "price": 5.5,  
            "quantity": 50,  
            "product": {  
                "unit": "H87",  
                "name": "Joghurt Banane",  
                "description": "",  
                "vatpercent": 7,  
                "taxCategoryCode": "S"  
            }  
        }  
    ]  
}
```



*Schaubild 1: Sample JSON structure to write invoices*

The following BTs are mapped as follows:

BT ID	CII reading
	<b>JSONPath</b>
	<b>recipient</b>
BT-44	recipient.name
BT-46	recipient.id
BT-48	recipient.vatid ( also mentioned as recipient.vatID)
BT-50	recipient.street
BT-51	recipient.additionalAddress
BT-163	recipient.additionalAddressExtension
BT-52	recipient.location
BT-53	recipient.zip
BT-55	recipient.country
ID BT-46	recipient.globalID
ID Scheme BT-46	recipient.globalIDScheme
BT-56	recipient.contact.name
BT-57	recipient.contact.phone
BT-58	recipient.contact.email
BT-45	recipient.legalOrganisation.tradingBusinessName
	<b>sender</b>
BT-27	sender.name
BT-28	sender.legalOrganisation.tradingBusinessName

BT-29	sender.globalID
BT-29-ID	sender.globalIDScheme
BT-31	sender.vatid ( also mentioned as sender.vatID)
BT-32	sender.taxID
BT-35	sender.street
BT-36	sender.additionalAddress
BT-162	sender.additionalAddressExtension
BT-38	sender.zip
BT-37	sender.location
BT-40	sender.country
BT-41	sender.contact.name
BT-42	sender.contact.phone
BT-43	sender.contact.email

### **Invoice**

BT-1	number
BT-3	documentCode
BT-5	currency
BT-13	buyerOrderReferencedDocumentID
BT-14	sellerOrderReferencedDocumentID
BT-16	despatchAdviceReferencedDocumentID
BT-10	referenceNumber

### **Allowance at invoice level**

BT-92	zfallowances.totalAmount
BT-95	zfallowances.categoryCode
BT-96	zfallowances.taxPercent
BT-97	zfallowances.reason
BT-98	zfallowances.reasonCode

### **Charge amount at invoice level**

BT-102	zfcharges.categoryCode
BT-99	zfcharges.totalAmount
BT-103	zfcharges.taxPercent
BT-104	zfcharges.reason

BT-105 zfcharges.reasonCode

### **Payment Details & Invoice Comment**

BT-9 dueDate

BT-84 tradeSettlement.iban

### **Price Details**

BT-129 zitems.quantity

BT-130 zitems.product.unit

BT-146 zitems.price

BT-151 zitems.product.taxCategoryCode

BT-152 zitems.product.vatpercent

BT-132 zitems.buyerOrderReferencedDocumentLineID

BT-153 zitems.product.name

BT-155 zitems.product.sellerAssignedID

BT-157 zitems.product.globalID

BT-157-ID zitems.product.globalIDScheme

BT-128 zitems.additionalReferences.issuerAssignedID

### **Allowance at line level**

BT-137 Missing – basic amount

BT-136 zfallowances.totalAmount

BT-138 Missing – percentage

BT-139 zfallowances.reason

BT-140 zfallowances.reasonCode

### **Charges at invoice inline level**

BT-142 Missing – basic amount

BT-143 Missing – percentage

BT-141 zfcharges.totalAmount

BT-144 zfcharges.reason

BT-145 zfcharges.reasonCode

## **CalculatedInvoice**

E.g. [https://www.mustangproject.org/files/MustangGnuaccountingBeispielRE-20201121\\_508.pdf](https://www.mustangproject.org/files/MustangGnuaccountingBeispielRE-20201121_508.pdf)

parses as

```
{ "documentName": null, "documentCode": "380", "number": "RE-20201121/508",  
"ownOrganisationFullPlaintextInfo": null, "referenceNumber": "AB321", "shipToOrganisationID": null,  
"shipToOrganisationName": null, "shipToStreet": null, "shipToZIP": null, "shipToLocation": null,  
"shipToCountry": null, "buyerOrderReferencedDocumentID": null, "invoiceReferencedDocumentID": null,  
"buyerOrderReferencedDocumentIssueDateTime": null, "ownForeignOrganisationID": null,  
"ownOrganisationName": "Bei Spiel GmbH", "currency": "EUR", "paymentTermDescription": null,  
"issueDate": "2020-11-20T23:00:00.000+00:00", "dueDate": "2020-12-11T23:00:00.000+00:00",  
"deliveryDate": "2020-11-09T23:00:00.000+00:00", "sender": { "name": "Bei Spiel GmbH", "zip":  
"12345", "street": "Ecke 12", "location": "Stadthausen", "country": "DE", "taxID": null, "vatID":  
"DE136695976", "additionalAddress": null, "additionalAddressExtension": null, "bankDetails":  
[ { "accountName": null, "bic": null, "iban": "DE8820080000970375700" } ], "contact": null,  
"legalOrganisation": null, "uriUniversalCommunicationID": null, "uriUniversalCommunicationIDScheme":  
null, "globalID": null, "globalIDScheme": null, "vatID": "DE136695976", "id": null, "email": null,  
"asTradeSettlement": [ { "accountName": null, "bic": null, "iban": "DE8820080000970375700" } ] },  
"recipient": { "name": "Theodor Est", "zip": "88802", "street": "Bahnstr. 42", "location":  
"Spielkreis", "country": "DE", "taxID": null, "vatID": null, "additionalAddress": null,  
"additionalAddressExtension": null, "bankDetails": [], "contact": null, "legalOrganisation": null,  
"uriUniversalCommunicationID": null, "uriUniversalCommunicationIDScheme": null, "globalID": null,  
"globalIDScheme": null, "vatID": null, "id": "2", "email": null, "asTradeSettlement": null },  
"deliveryAddress": null, "cashDiscounts": [], "notes": null, "sellerOrderReferencedDocumentID":  
null, "contractReferencedDocument": null, "totalPrepaidAmount": null, "paymentTerms": null,  
"invoiceReferencedIssueDate": null, "specifiedProcuringProjectID": null,  
"specifiedProcuringProjectName": null, "despatchAdviceReferencedDocumentID": null,  
"creditorReferenceID": null, "grandTotal": 571.04, "valid": false, "ownStreet": "Ecke 12", "ownZIP":  
"12345", "zfallowances": null, "zfitems": [ { "price": 160, "quantity": 1, "tax": null,  
"grossPrice": null, "lineTotalAmount": null, "basisQuantity": 1, "detailedDeliveryPeriodFrom": null,  
"detailedDeliveryPeriodTo": null, "id": null, "product": { "unit": "HUR", "name": "Design (hours)" },  
"sellerAssignedID": null, "buyerAssignedID": null, "description": "", "countryOfOrigin": null,  
"attributes": null, "intraCommunitySupply": false, "vatpercent": 7, "globalID": null,  
"globalIDScheme": null, "reverseCharge": false, "taxCategoryCode": "S", "taxExemptionReason":  
null }, "notes": null, "referencedDocuments": null, "additionalReferences": null,  
"buyerOrderReferencedDocumentLineID": null, "itemAllowances": null, "itemCharges": null,  
"itemTotalAllowances": null, "additionalReferencedDocumentID": null, "value": 160 }, { "price":  
0.79, "quantity": 400, "tax": null, "grossPrice": null, "lineTotalAmount": null, "basisQuantity": 1,  
"detailedDeliveryPeriodFrom": null, "detailedDeliveryPeriodTo": null, "id": null, "product":  
{ "unit": "H87", "name": "Ballons", "sellerAssignedID": null, "buyerAssignedID": null,  
"description": "", "countryOfOrigin": null, "attributes": null, "intraCommunitySupply": false,  
"vatpercent": 19, "globalID": null, "globalIDScheme": null, "reverseCharge": false,  
"taxCategoryCode": "S", "taxExemptionReason": null }, "notes": null, "referencedDocuments": null,  
"additionalReferences": null, "buyerOrderReferencedDocumentLineID": null, "itemAllowances": null,  
"itemCharges": null, "itemTotalAllowances": null, "additionalReferencedDocumentID": null, "value":  
0.79 }, { "price": 0.025, "quantity": 800, "tax": null, "grossPrice": null, "lineTotalAmount": null,  
"basisQuantity": 1, "detailedDeliveryPeriodFrom": null, "detailedDeliveryPeriodTo": null, "id":  
null, "product": { "unit": "LTR", "name": "Hot air „heiße Luft“ (litres)", "sellerAssignedID": null,  
"buyerAssignedID": null, "description": "", "countryOfOrigin": null, "attributes": null,  
"intraCommunitySupply": false, "vatpercent": 19, "globalID": null, "globalIDScheme": null,  
"reverseCharge": false, "taxCategoryCode": "S", "taxExemptionReason": null }, "notes": null,  
"referencedDocuments": null, "additionalReferences": null, "buyerOrderReferencedDocumentLineID":  
null, "itemAllowances": null, "itemCharges": null, "itemTotalAllowances": null,  
"additionalReferencedDocumentID": null, "value": 0.025 } ], "ownTaxID": null, "tradeSettlement": [ {  
"accountName": null, "bic": null, "iban": "DE8820080000970375700" } ], "ownVATID": "DE136695976",  
"ownLocation": "Stadthausen", "ownCountry": "DE", "zfcharges": null, "detailedDeliveryPeriodTo":  
null, "notesWithSubjectCode": null, "detailedDeliveryPeriodFrom": null, "vatdueDateTypeCode": null,  
"zflogisticsServiceCharges": null, "additionalReferencedDocuments": null, "subjectNote": null,  
"tradeSettlementPayment": null }
```

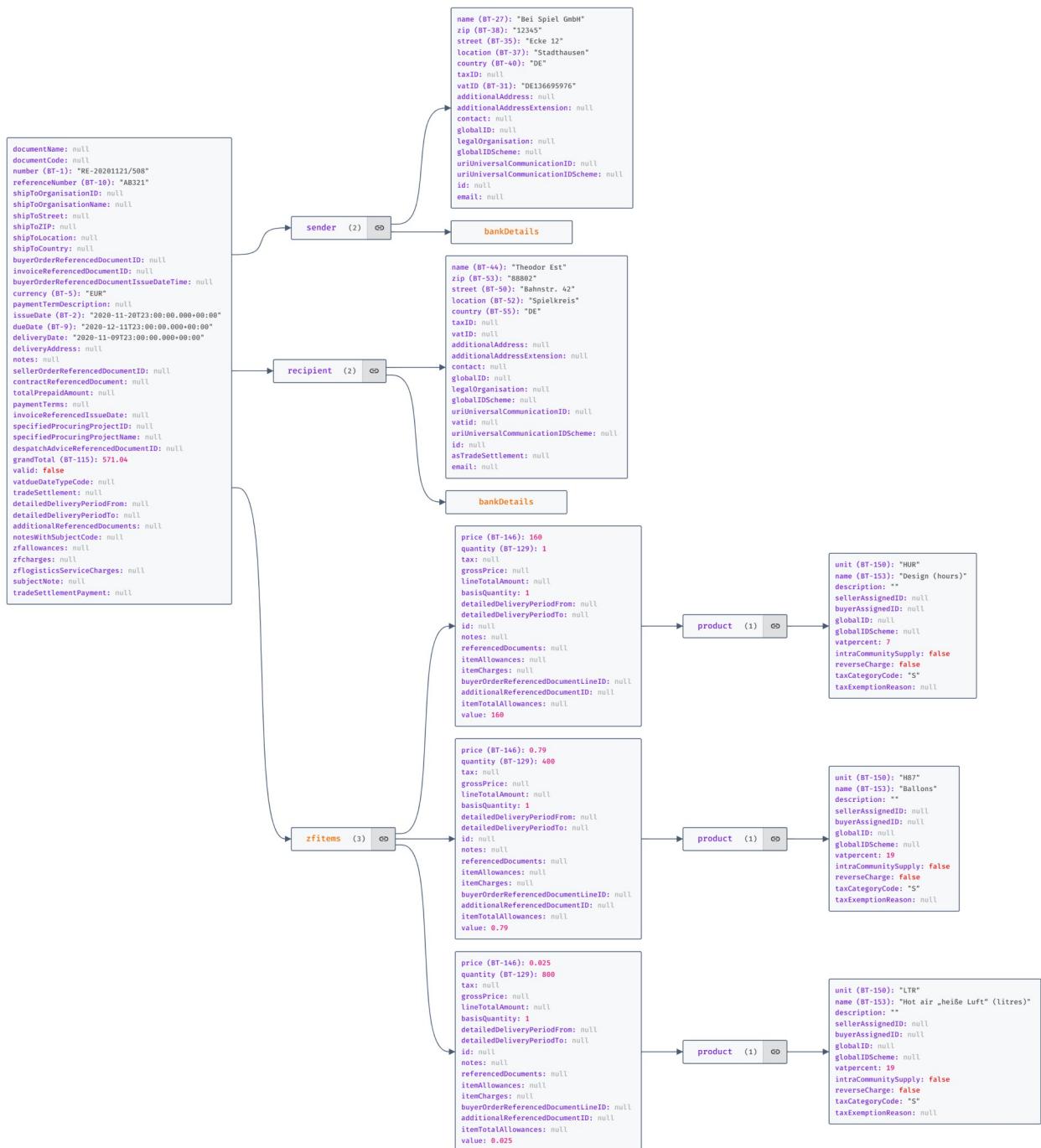


Schaubild 2: Sample read invoice JSON structure

## Operations/Endpoints

Mustangserver's available operations are

## Ping

Just a test, always just returns „pong“. The only operation accessible via HTTP GET, the rest is POST.

## Validate

validate: Validate a Factur-X/ZUGFeRD or XRechnung CII or Order-X-CIO File using Mustang's validator. Requires a file and returns a XML report. The format to be validated against will be read from it's guideline ID.

## Phive

Validate a CII or UBL file using <https://github.com/phax/phive>. Requires a file and returns JSON.

The available 173 format/standard/version-combinations (VES IDs) are

de.xrechnung:cii:1.2.0	de.xrechnung:ubl-invoice:3.0.2
de.xrechnung:cii:1.2.1	es.gob:facturae:3.0.0
de.xrechnung:cii:1.2.2	es.gob:facturae:3.1.0
de.xrechnung:cii:2.0.0	es.gob:facturae:3.2.0
de.xrechnung:cii:2.0.1	es.gob:facturae:3.2.1
de.xrechnung:cii:2.1.1	es.gob:facturae:3.2.2
de.xrechnung:cii:2.2.0	eu.cen.en16931:cii:1.0.0
de.xrechnung:cii:2.3.1	eu.cen.en16931:cii:1.1.0
de.xrechnung:cii:3.0.0	eu.cen.en16931:cii:1.2.0
de.xrechnung:cii:3.0.1	eu.cen.en16931:cii:1.2.1
de.xrechnung:cii:3.0.2	eu.cen.en16931:cii:1.2.3
de.xrechnung:ubl-creditnote:1.2.0	eu.cen.en16931:cii:1.3.0
de.xrechnung:ubl-creditnote:1.2.1	eu.cen.en16931:cii:1.3.1
de.xrechnung:ubl-creditnote:1.2.2	eu.cen.en16931:cii:1.3.10
de.xrechnung:ubl-creditnote:2.0.0	eu.cen.en16931:cii:1.3.11
de.xrechnung:ubl-creditnote:2.0.1	eu.cen.en16931:cii:1.3.12
de.xrechnung:ubl-creditnote:2.1.1	eu.cen.en16931:cii:1.3.2
de.xrechnung:ubl-creditnote:2.2.0	eu.cen.en16931:cii:1.3.3
de.xrechnung:ubl-creditnote:2.3.1	eu.cen.en16931:cii:1.3.4
de.xrechnung:ubl-creditnote:3.0.0	eu.cen.en16931:cii:1.3.5
de.xrechnung:ubl-creditnote:3.0.1	eu.cen.en16931:cii:1.3.6
de.xrechnung:ubl-creditnote:3.0.2	eu.cen.en16931:cii:1.3.6; qualifier=a
de.xrechnung:ubl-invoice:1.2.0	eu.cen.en16931:cii:1.3.7
de.xrechnung:ubl-invoice:1.2.1	eu.cen.en16931:cii:1.3.8
de.xrechnung:ubl-invoice:1.2.2	eu.cen.en16931:cii:1.3.9
de.xrechnung:ubl-invoice:2.0.0	eu.cen.en16931:ubl-creditnote:1.0.0
de.xrechnung:ubl-invoice:2.0.1	eu.cen.en16931:ubl-creditnote:1.1.0
de.xrechnung:ubl-invoice:2.1.1	eu.cen.en16931:ubl-creditnote:1.2.0
de.xrechnung:ubl-invoice:2.2.0	eu.cen.en16931:ubl-creditnote:1.2.1
de.xrechnung:ubl-invoice:2.3.1	eu.cen.en16931:ubl-creditnote:1.2.3
de.xrechnung:ubl-invoice:3.0.0	eu.cen.en16931:ubl-creditnote:1.3.0
de.xrechnung:ubl-invoice:3.0.1	eu.cen.en16931:ubl-creditnote:1.3.1

eu.cen.en16931:ubl-creditnote:1.3.10	eu.peppol.bis3.sg.ubl:creditnote:2023.7.0
eu.cen.en16931:ubl-creditnote:1.3.11	eu.peppol.bis3.sg.ubl:invoice:2023.12.0
eu.cen.en16931:ubl-creditnote:1.3.12	eu.peppol.bis3.sg.ubl:invoice:2023.7.0
eu.cen.en16931:ubl-creditnote:1.3.2	eu.peppol.bis3:catalogue-response:2023.11.0
eu.cen.en16931:ubl-creditnote:1.3.3	eu.peppol.bis3:catalogue-response:2023.5.0
eu.cen.en16931:ubl-creditnote:1.3.4	eu.peppol.bis3:catalogue-response:2024.5.0
eu.cen.en16931:ubl-creditnote:1.3.5	eu.peppol.bis3:catalogue:2023.11.0
eu.cen.en16931:ubl-creditnote:1.3.6	eu.peppol.bis3:catalogue:2023.5.0
eu.cen.en16931:ubl-creditnote:1.3.6; qualifier=a	eu.peppol.bis3:catalogue:2024.5.0
eu.cen.en16931:ubl-creditnote:1.3.7	eu.peppol.bis3:creditnote:2023.11.0
eu.cen.en16931:ubl-creditnote:1.3.8	eu.peppol.bis3:creditnote:2023.5.0
eu.cen.en16931:ubl-creditnote:1.3.9	eu.peppol.bis3:creditnote:2024.5.0
eu.cen.en16931:ubl:1.0.0	eu.peppol.bis3:despatch-advice:2023.11.0
eu.cen.en16931:ubl:1.1.0	eu.peppol.bis3:despatch-advice:2023.5.0
eu.cen.en16931:ubl:1.2.0	eu.peppol.bis3:despatch-advice:2024.5.0
eu.cen.en16931:ubl:1.2.1	eu.peppol.bis3:invoice-message-response:2023.11.0
eu.cen.en16931:ubl:1.2.3	eu.peppol.bis3:invoice-message-response:2023.5.0
eu.cen.en16931:ubl:1.3.0	eu.peppol.bis3:invoice-message-response:2024.5.0
eu.cen.en16931:ubl:1.3.1	eu.peppol.bis3:invoice:2023.11.0
eu.cen.en16931:ubl:1.3.10	eu.peppol.bis3:invoice:2023.5.0
eu.cen.en16931:ubl:1.3.11	eu.peppol.bis3:invoice:2024.5.0
eu.cen.en16931:ubl:1.3.12	eu.peppol.bis3:order-agreement:2023.11.0
eu.cen.en16931:ubl:1.3.2	eu.peppol.bis3:order-agreement:2023.5.0
eu.cen.en16931:ubl:1.3.3	eu.peppol.bis3:order-agreement:2024.5.0
eu.cen.en16931:ubl:1.3.4	eu.peppol.bis3:order-cancellation:2023.11.0
eu.cen.en16931:ubl:1.3.5	eu.peppol.bis3:order-cancellation:2023.5.0
eu.cen.en16931:ubl:1.3.6	eu.peppol.bis3:order-cancellation:2024.5.0
eu.cen.en16931:ubl:1.3.6; qualifier=a	eu.peppol.bis3:order-change:2023.11.0
eu.cen.en16931:ubl:1.3.7	eu.peppol.bis3:order-change:2023.5.0
eu.cen.en16931:ubl:1.3.8	eu.peppol.bis3:order-change:2024.5.0
eu.cen.en16931:ubl:1.3.9	eu.peppol.bis3:order-response-advanced:2023.11.0
eu.peppol.bis3.aunz.ubl:creditnote-self-billing:1.0.10	eu.peppol.bis3:order-response-advanced:2023.5.0
eu.peppol.bis3.aunz.ubl:creditnote-self-billing:1.0.11	eu.peppol.bis3:order-response-advanced:2024.5.0
eu.peppol.bis3.aunz.ubl:creditnote-self-billing:1.0.9	eu.peppol.bis3:order-response-advanced:2023.11.0
eu.peppol.bis3.aunz.ubl:creditnote:1.0.10	eu.peppol.bis3:order-response:2023.5.0
eu.peppol.bis3.aunz.ubl:creditnote:1.0.11	eu.peppol.bis3:order-response:2024.5.0
eu.peppol.bis3.aunz.ubl:creditnote:1.0.9	eu.peppol.bis3:order:2023.11.0
eu.peppol.bis3.aunz.ubl:invoice-self-billing:1.0.10	eu.peppol.bis3:order:2023.5.0
eu.peppol.bis3.aunz.ubl:invoice-self-billing:1.0.11	eu.peppol.bis3:order:2024.5.0
eu.peppol.bis3.aunz.ubl:invoice-self-billing:1.0.9	eu.peppol.bis3:punch-out:2023.11.0
eu.peppol.bis3.aunz.ubl:invoice:1.0.10	eu.peppol.bis3:punch-out:2023.5.0
eu.peppol.bis3.aunz.ubl:invoice:1.0.11	
eu.peppol.bis3.aunz.ubl:invoice:1.0.9	
eu.peppol.bis3.sg.ubl:creditnote:2023.12.0	

eu.peppol.bis3:punch-out:2024.5.0	org.peppol.jp.pint:credit-note:0.1.2
eu.peppol.directory:businesscard:1.0.0	org.peppol.jp.pint:invoice:0.1.2
eu.peppol.directory:businesscard:2.0.0	org.peppol.pint.my:creditnote-self-billing:1.0.0
eu.peppol.directory:businesscard:3.0.0	org.peppol.pint.my:creditnote:1.0.0
eu.peppol.reporting:eusr:1.1.4	org.peppol.pint.my:invoice-self-billing:1.0.0
eu.peppol.reporting:eusr:1.1.5	org.peppol.pint.my:invoice:1.0.0
eu.peppol.reporting:tsr:1.0.4	org.peppol.pint:credit-note:1.0.0
eu.peppol.reporting:tsr:1.0.5	org.peppol.pint:credit-note:1.0.1
it.fatturapa:invoice:1.2.0	org.peppol.pint:invoice:1.0.0
it.fatturapa:invoice:1.2.1	org.peppol.pint:invoice:1.0.1
it.fatturapa:invoice:1.2.2	

## pdf

(in the Mustangserver-docs API) Create a PDF/A file from any input PDF. Requires a PDF file (plain PDF, PDF A/1, PDF/A-3 or PDF/X) and a integer PDFAVersion. This operation will remove all non-PDF/A features as well as any embedded files, including potentially available Factur-X/ZUGFeRD files, and embed only available fonts. PDFAVersion should be 1, 2 or 3 for PDF/A-1, PDF/A-2 or PDF/A-3 respectively.

## parse

Read a Factur-X/ZUGFeRD/XRechnung and create a JSON representation. Requires a Factur-X , Order-X or Xrechnung-file.

Will return a calculatedInvoice, i.e. a grandTotal will be available. If it is set when writing (invoice2XML and combineInvoice), it will be compared vis à vis the calculated items and an exception will be thrown if the values do not match. In particular when writing a Factur-X PDF this should be used to ensure the machine readable XML-values match the human readable PDF values.

The object will look as described for CalculatedInvoice example on page 11.

## Invoice2xml

Convert a Factur-X/ZUGFeRD/XRechnung JSON representation to XML. Requires a input JSON string, a format (ZUGFeRD = zf, XRechnung = xr, Factur-X = fx or Order-X=ox), a version (usually 2 for ZUGFeRD and 1 for Factur-X) and a profile ("MINIMUM","BASICWL","BASIC","EN16931","EXTENDED" or "XRECHNUNG" for Factur-X, for ZUGFeRD 1 "BASIC","COMFORT" or "EXTENDED"). For XRechnung only "XRECHNUNG".

Please refer to the documentation of the Invoice class on page 7

## Extract

Extracts just the XML (not as JSON like parse) from a Factur-X/ZUGFeRD/Order-X file.

## detach

Parameter: file

Extracts all file attachments from the PDF (including e.g. a faktur-x.xml) and the XML, if the invoice has attachments, and returns a JSON structure with base64-encoded contents like this:

```
{
  "pdf": [],
  "xml": [ {"aFileA.png": "iVBORw0KGgoAAAANSUhEUgAAAAgAAC1CAQAAADIUnarAAAABGdBTUEALGPC/xhBQAAACBjSFJNAAAB6JgAAgIQAAPoAAAC6AAAAdTAAAOpGAAA6mAAAFC3Cc1E8AAAAAmJLR0QA/4ePzL8AAAACjEhZcwAACxMAAAsTAQCanBgAAAAHdElNRQfkAQIXGQ0qsHJfAAAAo01EQVRix+3MoQrCUBjF8f+9dzBtgha72eAYLJgMQ4tvMREfQuw+wGA2mxarxTeYgi9gEwyDnaOfYTrM5u+kw4/DcQAwI4PO+Q5gK2AsUVW+UEDBQFBQUFB4W91QhMRL3oABpKppLaLmEIm2cXMR6897c++cEM3WPBPUBw2eVrzfr/Gtd6Cs4SFH11/DFn18oDTLbOcgPwWFacpNVpstvAG3bSYVfhBdGAAAACV0RVh0ZGF0ZTpjcmVhdGUAMjAyMC0wMS0wM1QyMzoyNToxMyswMDowMEN9AywAAA1dEVYdGRhdGU6bW9kaWZ5ADIwMjAtMDEtMDJUMjM6MjU6MTMrMDA6MDAyILuQAAAAAE1FTkSuQmCC"}, {"sameFileB.png": "iVBORw0KGgoAAAANSUhEUgAAAAgAAC1CAQAAADIUnarAAAABGdBTUEALGPC/xhBQAAACBjSFJNAAAB6JgAAgIQAAPoAAAC6AAAAdTAAAOpGAAA6mAAAFC3Cc1E8AAAAAmJLR0QA/4ePzL8AAAACjEhZcwAACxMAAAsTAQCanBgAAAAHdElNRQfkAQIXGQ0qsHJfAAAAo01EQVRix+3MoQrCUBjF8f+9dzBtgha72eAYLJgMQ4tvMREfQuw+wGA2mxarxTeYgi9gEwyDnaOfYTrM5u+kw4/DcQAwI4PO+Q5gK2AsUVW+UEDBQFBQUFB4W91QhMRL3oABpKppLaLmEIm2cXMR6897c++cEM3WPBPUBw2eVrzfr/Gtd6Cs4SFH11/DFn18oDTLbOcgPwWFacpNVpstvAG3bSYVfhBdGAAAACV0RVh0ZGF0ZTpjcmVhdGUAMjAyMC0wMS0wM1QyMzoyNToxMyswMDowMEN9AywAAA1dEVYdGRhdGU6bW9kaWZ5ADIwMjAtMDEtMDJUMjM6MjU6MTMrMDA6MDAyILuQAAAAAE1FTkSuQmCC"}]
```

The input file can be a PDF/A-3, or a XML. Please note a PDF/A-3 may have both pdf and xml attachments at the same time (embedded within the xml which is embedded in the PDF).

## combine

Combines a JSON encoded invoice object (as described for „parse“, pg 15) and a PDF/A document to a Factur-X/ZUGFeRD PDF/A-3 document. Requires a input PDF/A-1 or A-3 file, a format (ZUGFeRD = zf, Factur-X = fx or Order-X = ox), a version (usually 2 for ZUGFeRD and 1 for Factur-X) and a profile ("MINIMUM", "BASICWL", "BASIC", "EN16931", "EXTENDED" or "XRECHNUNG" for Factur-X, for ZUGFeRD 1 "BASIC", "COMFORT" or "EXTENDED").

If returnJSON (optional) is true (default false) the return value will be a JSON whose key „pdf“ is base64 encoded. If ignorePDFAErrors (optional) is true (default false) the PDFbox pre-validation will raise no exceptions if the input PDF/A file is invalid.

The attribute grandTotal will be calculated by multiplying item quantities with their prices, adding the lines, adding charges and removing allowances, and adding the calculated VAT amounts.

Sample for writing, e.g. Invoice2XML: Please refer to the documentation of the Invoice class on page 7

## combineXML

Combines CII XML and a PDF/A document to a Factur-X/ZUGFeRD PDF/A-3 document. Requires a input PDF/A-1 or A-3 file, a format (ZUGFeRD = zf, Factur-X = fx or Order-X =

ox), a version (usually 2 for ZUGFeRD and 1 for Factur-X) and a profile ("MINIMUM","BASICWL","BASIC","EN16931","EXTENDED" or "XRECHNUNG" for Factur-X, for ZUGFeRD 1 "BASIC","COMFORT" or "EXTENDED").

If returnJSON (optional) is true (default false) the return value will be a JSON whose key „pdf“ is base64 encoded. If ignorePDFAErrors (optional) is true (default false) the PDFbox pre-validation will raise no exceptions if the input PDF/A file is invalid.

## **cii2UBL**

transforms XML from the UN/CEFACT Cross Industry Invoice (CII) XML format, the basis of factur-x/ZUGFeRD and the CII version of the XRechnung, to the Universal Business Language format, UBL. Requires a CII string.

## **xmltohtml**

converts a UBL or CII XML file (parameter file) into a human readable HTML in the language specified in language, which can be EN, DE or FR. The resulting file will require the additional files in the same directory:

- [xrechnung-viewer.css](#) and
- [xrechnung-viewer.js](#)

## **xmltopdf**

converts a UBL or CII XML file (parameter file) into a human readable PDF/A-3 in german language. The PDF will not have file attachments, i.e. you still have to combineXML if you want to get a Factur-X/ZUGFeRD file from a XML.

## **Example PHP Client**

This example operates in a PHP context but <https://editor.swagger.io/> also allows C#, Dart, HTML, Go, Java, Javascript, Kotlin, Python, R, Ruby, Scala, Swift and Typescript.

## **Mustangserver Hello World**

### ***Preparations***

Screenshot 1:

#1. Log in on <https://api.usegroup.de/devportal/>, select the latest Mustangserver API and download the OpenAPI (=Swagger) definition of the API (->1.)

#2. Open the file in a text editor, select all and copy

#3. Go to editor.swagger.io, paste the definition and confirm conversion to yaml. Select Generate Client|PHP (3.). The API is public so usually there is no need to create code in a private matter. However, it is possible: Swagger editor is open source under the APL license

(<https://github.com/swagger-api/swagger-editor>) and e.g. a Docker Image can be obtained from <https://registry.hub.docker.com/r/sebp/swagger-editor>, i.e. using

```
sudo docker run --rm -p 8080:8080 sebp/swagger-editor
```

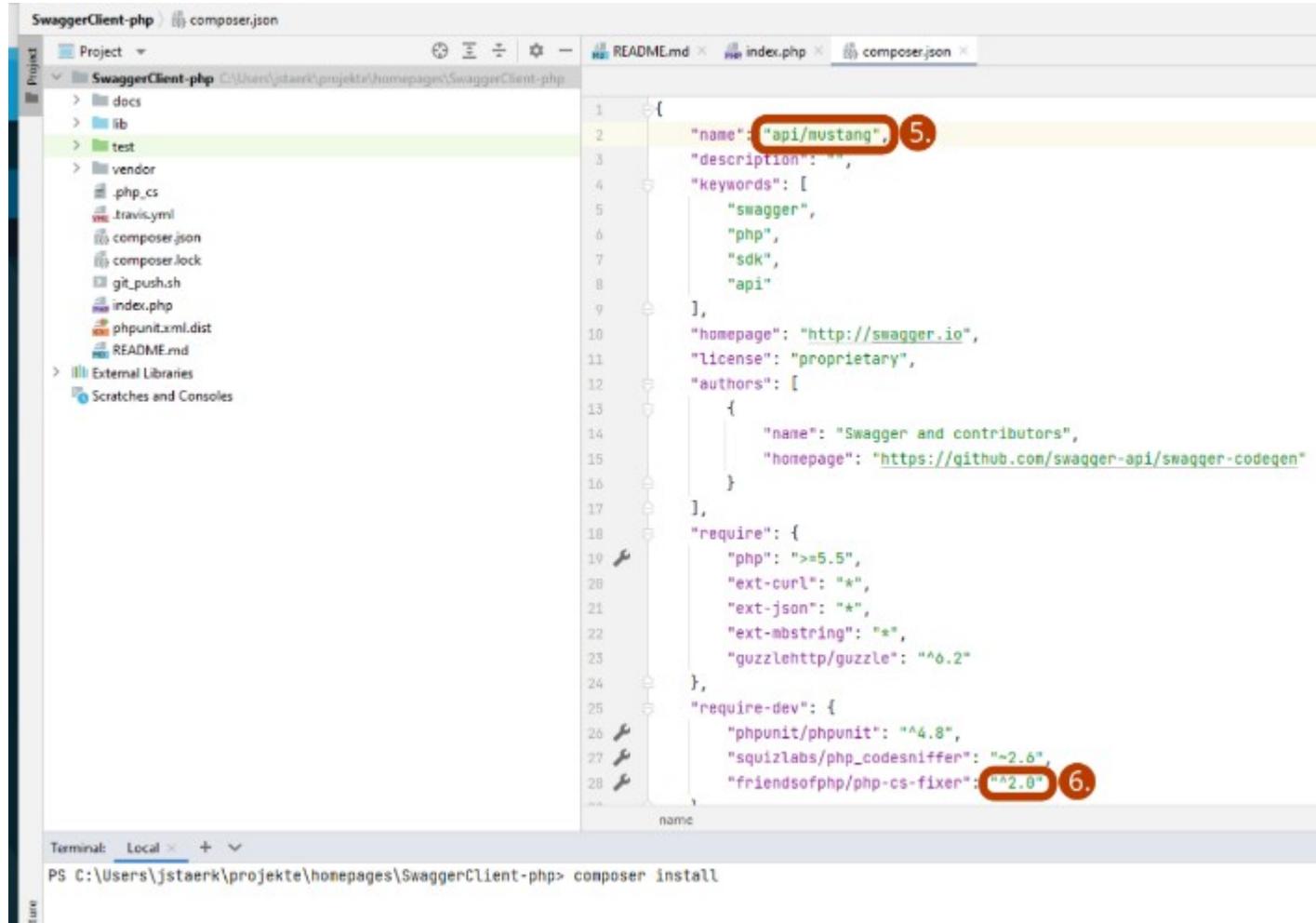
to run locally via port 8080.

Screenshot 2:

#4. Extract the downloaded file, edit composer.json.

#5. change the name of the project in the composer.json file to lowercaps/lowercaps (5.)

Screenshot 3:



```
SwaggerClient-php composer.json
Project  README.md index.php composer.json
SwaggerClient-php C:\Users\jstaerk\projekte\homepages\SwaggerClient-php
> docs
> lib
> test
> vendor
  -php.cs
  .travis.yml
  composer.json 5.
  composer.lock
  git_push.sh
  index.php
  phpunit.xml.dist
  README.md
> External Libraries
Scatches and Consoles

{
  "name": "api/mustang", 5.
  "description": "",
  "keywords": [
    "swagger",
    "php",
    "sdk",
    "api"
  ],
  "homepage": "http://swagger.io",
  "license": "proprietary",
  "authors": [
    {
      "name": "Swagger and contributors",
      "homepage": "https://github.com/swagger-api/swagger-codegen"
    }
  ],
  "require": {
    "php": ">=5.5",
    "ext-curl": "*",
    "ext-json": "*",
    "ext-mbstring": "*",
    "guzzlehttp/guzzle": "^6.2"
  },
  "require-dev": {
    "phpunit/phpunit": "^4.8",
    "squizlabs/php_codesniffer": "^2.6",
    "friendsofphp/php-cs-fixer": "2.0" 6.
  }
}, name
Terminal Local + ^
PS C:\Users\jstaerk\projekte\homepages\SwaggerClient-php> composer install
```

#6. If you want to use PHP8+ upgrade the version number of php-cs-fixer to ^2.0. Then run "composer install" in that directory.

#7. Copy the example from the "Getting started" section of the readme.md to a new file, called index.php

## Allowing Client Credentials

Screenshot 4:

The screenshot shows the WSO2 API Manager Developer Portal interface. The URL is https://api.usegroup.de:9443/devportal/applications/d06a3f0d-2997-4a4f-9a70-89bac2cecb12/productionkeys/oauth. The top navigation bar has a 'DEVELOPER PORTAL' tab, a 'WSO2 API MANAGER' logo, and a 'Applications' button which is highlighted with a red circle labeled '8.'. Below the navigation is a search bar with 'Search APIs'. On the left sidebar, there are several menu items: 'Overview', 'Production Keys', 'OAuth2 Tokens' (which is also highlighted with a red circle labeled '9.'), 'API Key', 'Sandbox Keys', 'OAuth2 Tokens' (repeated), 'API Key', and 'Subscriptions'. The main content area is titled 'DefaultApplication' with '3 Subscriptions'. Under 'Production Keys', it says '9. Production OAuth2 Keys'. It shows a 'Key and Secret' section with a 'Consumer Key' field containing 'MfVADlwYvTab1AzkHFQXzq5ASaEa' (highlighted with a red circle labeled '15.') and a 'Key and Secret' field below it. There are two buttons: 'GENERATE ACCESS TOKEN' (highlighted with a red circle labeled '11.') and 'TO GENERATE ACCESS TOKEN'. Under 'Key Configurations', it lists 'Token Endpoint' as 'https://localhost:9443/oauth2/token' and 'Revoke Endpoint' as 'https://localhost:9443/oauth2/revoke'. In the 'Grant Types' section, there are checkboxes for 'Refresh Token', 'SAML2', 'Password' (which is checked), and 'Client Credentials' (which is also checked). A note at the bottom says 'The application can use the following grant types to generate Access Tokens. Based on the a...'. The right side of the page has a vertical scroll bar.

#8. Click on Applications (8.), Default Application,  
#9. Production Keys/OAuth2 Token (9.).

#10. As preparation for authentication: Check Client Credentials and click the Update button on the bottom of the page.

## Get access token

#11. For this part we will use a token which will expire shortly. Click Generate Access Token (11.), Generate and copy the resulting token. Paste it in

Screenshot 5:

```

1 <?php
2 require_once(__DIR__ . '/vendor/autoload.php');
3
4 // Configure OAuth2 access token for authorization: default
5 $config = Swagger\Client\Configuration::getDefaultConfiguration()->setAccessToken( access_token: "12.eyJ4NXQ10iJNbVZpTnpFMU1HVm
6
7 $apiInstance = new Swagger\Client\Api\MustangControllerApi(); 13.
8 // If you want use custom http client, pass your client which implements 'GuzzleHttp\ClientInterface'.
9 // This is optional, 'GuzzleHttp\Client' will be used as default.
10 new GuzzleHttp\Client(),
11 $config
12 );
13
14 try {
15     $result = $apiInstance->ping(); 14.
16     print_r($result);
17 } catch (Exception $e) {
18     echo 'Exception when
19 }

```

The code completion dropdown for line 14 shows the following methods:

- ping() string
- pingAsync() GuzzleHttp\Promise\PromiseInterface
- pingAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- pingWithHttpInfo() array
- parseWithHttpInfo([input\_file: null|string = null]) array
- phiveWithHttpInfo(vesid: string, [input\_file: null|string = null]) array
- parseAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- phiveAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- cii2ublAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- cii2ublWithHttpInfo(body: string) array
- combineAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- combineWithHttpInfo([input\_file: null|string = null]) array

#12. index.php (12.), in the same file

#13. change ErrorController to Mustangcontroller (13.) and

#14. handle() to ping() (14.). Please note that usual PHP editors will give you code completion.

Now you can open resulting index.php via your server and PHP processor in your browser, it should now look like Screenshot 6:



## OAuth2 Authentication

Back to Screenshot 4:

In index.php paste the following code

```
$client = new GuzzleHttp\Client();
```

```

$res = $client->request('POST', 'https://gw.usegroup.de:9443/oauth2/token', [
    'auth' => ['<15.>', '<16.>'],
    'form_params' => [
        'grant_type' => 'client_credentials',
    ],
]);

```

\$json = json\_decode(\$res->getBody(), true);

Screenshot 7:

```

<?php

require_once(__DIR__ . '/vendor/autoload.php');

$client = new GuzzleHttp\Client();
$res = $client->request( method: 'POST',  url: 'https://gw.usegroup.de:9443/oauth2/token', [
    'auth' 16. MfVADlwYvTab1AzkHFQXzq5ASaEa,  client secret 18.
    'form_params' => [
        'grant_type' => 'client_credentials',
    ],
]);

```

// Configure OAuth2 access token for authorization: default

```

$config = Swagger\Client\Configuration::getDefaultConfiguration()->setAccessToken \$json\["access\_token"\] 19.

$apiInstance = new Swagger\Client\Api\MustangControllerApi(
// If you want use custom http client, pass your client which implements 'GuzzleHttp\ClientInterface'.
// This is optional, 'GuzzleHttp\Client' will be used as default.
    new GuzzleHttp\Client(),
    $config
);

try {
    $result = $apiInstance->validateFile( in file: "factur-x.pdf"); 21.
    print_r(htmlentities($result)) 22.
}

```

#15. copy Consumer Key (15., from screen 4) to the beginning of index.php (16.).

#17. reveal and copy Consumer Secret (18.).

#19 replace the static access token which will become invalid by \$json["access\_token"]

#20 Create or download a invoice to be validated, e.g.

[https://www.mustangproject.org/files/MustangGnuaccountingBeispielRE-20201121\\_508.pdf](https://www.mustangproject.org/files/MustangGnuaccountingBeispielRE-20201121_508.pdf) and save it as factur-x.pdf

#21 Change the method to validateFile and

#22 html escape the validation result, so that the result in the browser looks like screenshot 8:



```
<?xml version="1.0" encoding="UTF-8"?> <validation filename="tovalidate14506017597035795196mustang">
<releaseDetails id="validation-model" version="1.16.1" buildDate="2020-05-12T00:46:00+02:00"/> </buildIn>
<validation profile="statement" statement="PDF file is compliant with Validation Profile requirements." isCompliant="true">
<duration start="1665170599476" finish="1665170599476">00:00:04.691</duration> </job> </jobs> <batchSummary totalJobs="1" failedToParse="0" succeededJobs="1" succeeded="true" repairReports="0" failedJobs="0" failed="false" start="1665170587541" finish="1665170599531" />
</pdf> <xml> <info> <version>2</version> <profile>urn:cen.eu:en16931:2017#conformant#urn:fatur-x.eu:1.0</profile> <status>valid</status> </info> <summary status="valid"></summary> </xml> </validation>
```

That's it. Instead of displaying the XML you can now parse it :-)

Feel free to also try the async functions.

Screenshot 1:

A screenshot of the "API Manager" interface. On the left, there's a sidebar with options like Overview, Subscriptions, My API, Comments, Documentation, and SDKs. The main area shows an "Overview" for the "Mustang" API. It has a "MU" icon, the name "Mustang", and a "Version v0.5.1" section. Below that is an "IRI" field containing "https://api.usegroup.de/devportal/mustang/v0.5.1". There are "Test API" and "Edit API" buttons. To the right, there are sections for "Tags" (None), "Business Plans" (Unlimited), and a "Source" button with a "Download Swagger" link. A red box highlights the "Download Swagger" button.

#1. Log in on <https://api.usegroup.de/devportal/>, select the latest Mustangserver API and download the OpenAPI (=Swagger) definition of the API (->1.)

#2. Open the file in a text editor, select all and copy

#3. Go to editor.swagger.io, paste the definition and confirm conversion to yaml. Select Generate Client|PHP (3.)

Screenshot 2:

The screenshot shows the Swagger Editor interface at <https://editor.swagger.io>. On the left, the API definition is displayed in a code editor:

```

1  openapi: 3.0.1
2  info:
3    title: Mustang
4    description: Mustangproject e-invoice REST server API
5    contact:
6      name: Jochen Steenk
7      url: https://mustangproject.org/
8      email: jsteenk@seegroup.de
9    license:
10      name: proprietary
11      url: http://mustangproject.org/server
12    version: v0.5.1
13    externalDocs:
14      description: Mustang Documentation
15      url: https://mustangproject.org/use/
16    servers:
17      - url: https://gw.usigroup.de:8241/mustang/v0.5.1
18      - url: https://gw.usigroup.de:8288/mustang/v0.5.1
19    security:
20      - default: []
21    paths:
22      /mustang/validate:
23        post:
24          tags:
25            - mustang-controller
26            description: Checks a PDF or XML file for syntactical and math errors (Factor-X, ZUGFeRD) or a XML file (Factur-X, ZUGFeRD), if no operationId: validateFile
27            requestBody:
28              content:
29                multipart/form-data:
30                  schema:
31                    type: object
32                    properties:
33                      inFile:
34                        type: string
35                        format: binary
36
37        responses:
38          '200':
39            description: OK
40            content:
41              application/json:
42                schema:
43                  type: string
44            security:
45              - default: []
46            x-throttling-tier: Unlimited
47        /mustang/phive:
48          post:
49            tags:
50              - mustang-controller
51              description: Returns a <!--> caching validation string
52              operationId: phive
53              parameters:
54                - name: VESID
55                in: query
56                required: true
57                style: form
58                explode: true
59                schema:
60                  type: string
61                  example: de.rechnungslizenz
62              requestBody:

```

On the right, a sidebar lists various client generator options:

- csharp
- csharp-dotnet2
- dart
- dynamic-html
- go
- html
- html2
- java
- javascript
- objc-client
- kotlinclient
- openapi
- openapi-yaml
- python
- r
- ruby
- scala
- swf43
- swift5
- typescript-angular
- typescript-axios
- typescript-fetch

A red circle with the number "3." is overlaid on the "php" button. A dropdown menu shows "5.1" selected.

Below the sidebar, a list of API endpoints is shown with green "POST" buttons:

- /mustang/validate
- /mustang/phive
- /mustang/parse
- /mustang/invoice2XML
- /mustang/extract
- /mustang/combineXML
- /mustang/combine

#4. Extract the downloaded file, edit composer.json.

#5. change the name of the project in the composer.json file to lowercaps/lowercaps (5.)

Screenshot 3:

```
1 {  
2     "name": "api/mustang", 5.  
3     "description": "",  
4     "keywords": [  
5         "swagger",  
6         "php",  
7         "sdk",  
8         "api"  
9     ],  
10    "homepage": "http://swagger.io",  
11    "license": "proprietary",  
12    "authors": [  
13        {  
14            "name": "Swagger and contributors",  
15            "homepage": "https://github.com/swagger-api/swagger-codegen"  
16        }  
17    ],  
18    "require": {  
19        "php": ">=5.5",  
20        "ext-curl": "*",  
21        "ext-json": "*",  
22        "ext-mbstring": "*",  
23        "guzzlehttp/guzzle": "^6.2"  
24    },  
25    "require-dev": {  
26        "phpunit/phpunit": "^4.8",  
27        "squizlabs/php_codesniffer": "~2.6",  
28        "friendsofphp/php-cs-fixer": "^2.0" 6.  
29    }  
30}
```

Terminal Local + ↻  
PS C:\Users\jstaerk\projekte\homepages\SwaggerClient-php> composer install

#6. If you want to use PHP8+ upgrade the version number of php-cs-fixer to ^2.0. Then run "composer install" in that directory.

#7. Copy the example from the "Getting started" section of the readme.md to a new file, called index.php

Screenshot 4:

The screenshot shows the WSO2 API Manager Developer Portal interface. The URL is https://api.usegroup.de:9443/devportal/applications/d06a3f0d-2997-4a4f-9a70-89bac2cecb12/productionkeys/oauth. The top navigation bar has a 'DEVELOPER PORTAL' tab, a 'WSO2 API MANAGER' logo, and a 'Applications' button which is highlighted with a red circle labeled '8.'. Below the navigation is a search bar with 'Search APIs'. On the left sidebar, there are several tabs: 'Overview', 'Production Keys', 'OAuth2 Tokens' (which is also highlighted with a red circle labeled '9.'), 'API Key', 'Sandbox Keys', 'OAuth2 Tokens' (repeated), 'API Key', and 'Subscriptions'. The main content area shows a 'DefaultApplication' with '3 Subscriptions'. Under 'Production OAuth2 Keys', it says 'Key and Secret' and shows a 'Consumer Key' field containing 'MfVADlwYvTab1AzkHFQXzq5ASaEa' (highlighted with a red circle labeled '15.'). Below it is a 'GENERATE ACCESS TOKEN' button (highlighted with a red circle labeled '11.') and a 'TO GENERATE ACCESS TOKEN' link. Under 'Key Configurations', there are sections for 'Token Endpoint' (https://localhost:9443/oauth2/token) and 'Revoke Endpoint' (https://localhost:9443/oauth2/revoke). In the 'Grant Types' section, there are checkboxes for 'Refresh Token', 'SAML2', 'Password' (which is checked), and 'Client Credentials' (which is also checked). A note below says 'The application can use the following grant types to generate Access Tokens. Based on the a...'. The bottom right corner of the page has a small 'i' icon.

#8. Click on Applications (8.), Default Application,  
#9. Production Keys/OAuth2 Token (9.).

#10. As preparation for Part B: Check Client Credentials and click the Update button on the bottom of the page.

#11. For this part we will use a token which will expire shortly. Click Generate Access Token (11.), Generate and copy the resulting token. Paste it in

Screenshot 5:

```

1 <?php
2 require_once(__DIR__ . '/vendor/autoload.php');
3
4 // Configure OAuth2 access token for authorization: default
5 $config = Swagger\Client\Configuration::getDefaultConfiguration()->setAccessToken( access_token: "12. eyJ4NXQ10iJNbVZpTnpFMU1HVm" );
6
7 $apiInstance = new Swagger\Client\Api\MustangControllerApi(); 13.
8 // If you want use custom http client, pass your client which implements 'GuzzleHttp\ClientInterface'.
9 // This is optional, 'GuzzleHttp\Client' will be used as default.
10 new GuzzleHttp\Client(),
11 $config
12 );
13
14 try {
15     $result = $apiInstance->ping(); 14.
16     print_r($result);
17 } catch (Exception $e) {
18     echo 'Exception when';
19 }

```

The code completion dropdown for line 14 shows the following methods:

- m ping() string
- m pingAsync() GuzzleHttp\Promise\PromiseInterface
- m pingAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- m pingWithHttpInfo() array
- m parseWithHttpInfo([input\_file: null|string = null]) array
- m phiveWithHttpInfo(vesid: string, [input\_file: null]) array
- m parseAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- m phiveAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- m cii2ublAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- m cii2ublWithHttpInfo(body: string) array
- m combineAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterface
- m combineWithHttpInfo([input\_file: null|string = null]) array

#12. index.php (12.), in the same file

#13. change ErrorController to Mustangcontroller (13.) and

#14. handle() to ping() (14.). Please note that usual PHP editors will give you code completion.

Now you can open resulting index.php via your server and PHP processor in your browser, it should now look like Screenshot 6:



## Validation of electronic invoices

Concerning Screenshot 4:

In index.php paste the following code

```
$client = new GuzzleHttp\Client();
$res = $client->request('POST', 'https://gw.usegroup.de:9443/oauth2/token', [
```

```

'auth' => ['<15.>', '<16.>'],
'form_params' => [
    'grant_type' => 'client_credentials',
]
]);
$ json = json_decode($res->getBody(), true);

```

Screenshot 7:

```

<?php

require_once(__DIR__ . '/vendor/autoload.php');

$client = new GuzzleHttp\Client();
$res = $client->request('POST', 'https://gw.usegroup.de:9443/oauth2/token', [
    'auth' => 16. MfVADlwYvTab1AzkHFQXzq5ASaEa, 'client secret' => 18.
    'form_params' => [
        'grant_type' => 'client_credentials',
    ]
]);
$ json = json_decode($res->getBody(), associative: true);

// Configure OAuth2 access token for authorization: default
$config = Swagger\Client\Configuration::getDefaultConfiguration()->setAccessToken($json["access_token"]) 19.

$apiInstance = new Swagger\Client\Api\MustangControllerApi(
// If you want use custom http client, pass your client which implements 'GuzzleHttp\ClientInterface'.
// This is optional, 'GuzzleHttp\Client' will be used as default.
    new GuzzleHttp\Client(),
    $config
);

try {
    $result = $apiInstance->validateFile(in_file: "factur-x.pdf"); 21.
    print_r(htmlentities($result)) 22.
}

```

#15. copy Consumer Key (15., from screen 4) to the beginning of index.php (16.).

#17. reveal and copy Consumer Secret (18.).

#19 replace the static access token which will become invalid by \$json["access\_token"]

#20 Create or download a invoice to be validated, e.g.

[https://www.mustangproject.org/files/MustangGnuaccountingBeispielRE-20201121\\_508.pdf](https://www.mustangproject.org/files/MustangGnuaccountingBeispielRE-20201121_508.pdf) and save it as factur-x.pdf

#21 Change the method to validateFile and

#22 html escape the validation result, so that the result in the browser looks like screenshot 8:



```
<?xml version="1.0" encoding="UTF-8"?> <validation filename="tovalidate14506017597035795196mustang.pdf">
<releaseDetails id="validation-model" version="1.16.1" buildDate="2020-05-12T00:46:00+02:00"/> </buildIn>
<validationProfile statement="PDF file is compliant with Validation Profile requirements." isCompliant="true" finish="1665170599476">00:00:04.691</duration> </job> </jobs> <batchSummary totalJobs="1" failedToParse="0">
<repairReports failedJobs="0">0</repairReports> <duration start="1665170587541" finish="1665170599531">00:00:04.691</duration> <pdf> <xml> <info> <version>2</version> <profile>urn:cen.eu:en16931:2017#conformant#urn:fatur-x.eu:1.0</profile> <summary status="valid"/> </xml> </pdf> </batchSummary> </validationProfile>
</validation>
```

That's it. Instead of displaying the XML you can now parse it.

Feel free to also try the async functions.

## Example C# client

With node.js installed use

```
npm install @openapitools/openapi-generator-cli -g
```

then

```
openapi-generator-cli generate -i "swagger.json" -g csharp -o "csharpproject"
```

This will create a Library which uses RestSharp to access Mustang. Then use Microsoft Visual Studio Community or higher (not Visual Studio Code) to open Org.OpenAPITools.sln .

Get yourself an Api Key(see page 4)

Uncomment e.g. the ping test in src\Org.OpenAPITools.Test\Api\MustangControllerApiTests.cs

And change the constructor to

```
public MustangControllerApiTests()
{
    Configuration c = new Configuration();
    c.DefaultHeader.Add("apikey", "<your api key>");
    instance = new MustangControllerApi(c);
}
```

Alternatively, to use oauth,

see „Allowing Client Credentials“ on page 19

use

```

public MustangControllerApiTests()
{
    Configuration c = new Configuration();
    c OAuthFlow = OAuthFlow.APPLICATION;
    c OAuthTokenUrl = "https://api.usegroup.de:9443/oauth2/token";
    c OAuthClientId = "<your client id>";
    c OAuthClientSecret = "<your client secret>";

    instance = new MustangControllerApi(c);
}

```

If you want to use an API Key. With CTRL+E, T you can see the test explorer and with CTRL+R, A you can run all tests (of which only pingTest will be enabled).

## Converting PDF

Converting from PDF to PDF/A, fixing faulty PDF/A and removing file attachments (like Factur-X) are all done with the same Endpoint .../mustang-doc/v1.0.0/mustang/pdf. Please note that Mustangserver-docs is a separate API which needs to be separately be subscribed to.

## Interactive testing using Postman/Bruno

[Bruno](#) is an open source alternative to Postman, graphical user interfaces to create/execute requests on REST APIs.

This example is based on [Postman](#), you will need enabled client credentials as described on page 19 in Allowing Client Credentials.

Use Import|File|Upload Files to upload your Openapi.yaml file into a Mustangserver collection.

Add a request to [https://gw.usegroup.de:9443/oauth2/token?grant\\_type=client\\_credentials](https://gw.usegroup.de:9443/oauth2/token?grant_type=client_credentials) and call it Token Request. Postman will auto-detect the Parameter in the URL. Change the type to POST.

In Headers, add a new field Content-Type with application/json as its value.

The tab Authorization should be Basic auth with your client id and secret as username and password. Once you click Send, an according access token should be submitted:

POST https://gw.usgroup.de:9443/oauth2/token?grant\_type=client\_credentials

Params • Authorization • Headers (10) Body • Pre-request Script Tests Settings Cookies

Type Basic Auth

The authorization header will be automatically generated when you send the request. [Learn more about authorization ↗](#)

Username: HiGshBzJp5WopTSaaAgRiTUIPhla

Password:  Show Password

Body Cookies Headers (11) Test Results

Status: 200 OK Time: 436 ms Size: 1.32 KB Save Response ↗

Pretty Raw Preview Visualize JSON

```

1  "access_token": "eyJ4NkQ1Oj10jORgRpWkdZHE1tVmPx1ptN10a056VmpakUyimpelPhMx10VGrpTnpoa05HfTJNv0ZslXc1lCjraikQ1o1jObu14T1d0bE5qazNPV1E1wTjReU1tMTBORGNSTprM9URTV014TuDRNvphRXdNbUsTkdaek5EUTfNeK0w1RjNfpqVTfZm115tkRm11tRmQ0q95u211n111s1nfz161117hjU2In0."
2  eyJzdif101jhZG1pbilsImf1dC16IkFQUEjQ0FSU901iw1YXK1joiSG1Hc0hCeKpwNvDvcFRTyWFBZ1jsVFVsUEhjYS1sIn5iZ1i6MTY3MzK4MjE4N5w1YXpwIjo1SG1h0hCeKpwNvDvcFRTyWFBZ1jsVFVsUEhjYS1sInNjbj3B1jjo1zGVmYXsdC1sIm1zcy16Imhd0H201lwXChcGku0x11233xdAuZLG06TQmLwvb2f1dGgjXC9082t1bl1sIm4C16MTY3MzK4Nt4NsW1af01joxNjczTgjMtgLCqdgk101b2GyvTM1jy02NDU2LTrNz1QtG4Ml042m210TA5nVknNgifQ.
3  E4rjnU8gYvDcP9ujkIoUx1vhve1E2k2t0978Kx_jvtqjdsNmNj2FviziJxvraUPuYTRuNF45EE0-U5GE1i1TxmfT6zswjz5-1MF1cv2HJRNSV747K1MeUuAyU8jwAT62idzpe6DixVY03xEg5Xnsd18_jJojpkaZ_xQhqcw7KVn0g_oEjdQLqBueEtCARGQ05VqAF
4  80Lsd673M111k1nqaCze52w8y15FF2pnFoy4R8U4bIxC8qIf4jrmUKy_juowdMCOn2Rx1wn0Dby/NxFcc33cFG16k6c1mE7XK-UuUPJ-86_QntIVmDwFaef1n0JUty7CFYX8uFq1ew",
5  "scope": "default",
6  "token_type": "Bearer",
7  "expires_in": 3600

```

## Add

```

var jsonData = JSON.parse(responseBody);
pm.collectionVariables.set("token", jsonData.access_token);
console.log(jsonData.access_token);

```

In the „Tests“ tab and View|Show Postman Console (Alt+CTRL+C) . Once you click Send again you should be able to see the access token also in the Console:

Scratch Pad New Import Mustangserver POST token GET Ping GET ping POST Token Req. GET error Mustangserver Mustangserver + \*\*\* No Environment

Collections mustang POST validate File OK POST phive POST parse POST invoice+xml POST extract File POST combine XML POST curlUtil GET ping GET error OK POST Token Request Mustangserver / Token Request

POST https://gw.usgroup.de:9443/oauth2/token?grant\_type=client\_credentials

Params • Authorization • Headers (8) Body Pre-request Script Tests Settings

```

1 var jsonData = JSON.parse(responseBody);
2 pm.collectionVariables.set("token", jsonData.access_token);
3 console.log(jsonData.access_token);
4

```

Body Cookies Headers (11) Test Results

Status: 200 OK Time: 575 ms Size: 1.32 KB Save Response ↗

Pretty Raw Preview Visualize JSON

```

1  "access_token": "eyJ4NkQ1Oj10jORgRpWkdZHE1tVmPx1ptN10a056VmpakUyimpelPhMx10VGrpTnpoa05HfTJNv0ZslXc1lCjraikQ1o1jObu14T1d0bE5qazNPV1E1wTjReU1tMTBORGNSTprM9URTV014TuDRNvphRXdNbUsTkdaek5EUTfNeK0w1RjNfpqVTfZm115tkRm11tRmQ0q95u211n111s1nfz161117hjU2In0."
2  eyJzdif101jhZG1pbilsImf1d161kf0UxjQ0FSU901iw1YXK1joiSG1Hc0hCeKpwNvDvcFRTyWFBZ1jsVFVsUEhjYS1sIn5iZ1i6MTY3MzK4MjE4N5w1YXpwIjo1SG1h0hCeKpwNvDvcFRTyWFBZ1jsVFVsUEhjYS1sInNjbj3B1jjo1zGVmYXsdC1sIm1zcy16Imhd0H201lwXChcGku0x11233xdAuZLG06TQmLwvb2f1dGgjXC9082t1bl1sIm4C16MTY3MzK4Nt4NsW1af01joxNjczTgjMtgLCqdgk101b2GyvTM1jy02NDU2LTrNz1QtG4Ml042m210TA5nVknNgifQ.
3  E4rjnU8gYvDcP9ujkIoUx1vhve1E2k2t0978Kx_jvtqjdsNmNj2FviziJxvraUPuYTRuNF45EE0-U5GE1i1TxmfT6zswjz5-1MF1cv2HJRNSV747K1MeUuAyU8jwAT62idzpe6DixVY03xEg5Xnsd18_jJojpkaZ_xQhqcw7KVn0g_oEjdQLqBueEtCARGQ05VqAF
4  80Lsd673M111k1nqaCze52w8y15FF2pnFoy4R8U4bIxC8qIf4jrmUKy_juowdMCOn2Rx1wn0Dby/NxFcc33cFG16k6c1mE7XK-UuUPJ-86_QntIVmDwFaef1n0JUty7CFYX8uFq1ew",
5  "scope": "default",
6  "token_type": "Bearer",
7  "expires_in": 3600

```

Find and Replace Console 3 Errors All Logs Clear 208 45 ms

```

+ GET https://gw.usgroup.de:9443/mustang/v0.7.0/mustang/ping
+ {id: "Sceca9fd-f433-4d9e-89fe-61e453b3a2e5", mutations: {}, values: {}}
+ POST https://gw.usgroup.de:9443/oauth2/token?grant_type=client_credentials
+ "eyJ4NkQ1Oj10jORgRpWkdZHE1tVmPx1ptN10a056VmpakUyimpelPhMx10VGrpTnpoa05HfTJNv0ZslXc1lCjraikQ1o1jObu14T1d0bE5qazNPV1E1wTjReU1tMTBORGNSTprM9URTV014TuDRNvphRXdNbUsTkdaek5EUTfNeK0w1RjNfpqVTfZm115tkRm11tRmQ0q95u211n111s1nfz161117hjU2In0."
+ eyJzdif101jhZG1pbilsImf1d161kf0UxjQ0FSU901iw1YXK1joiSG1Hc0hCeKpwNvDvcFRTyWFBZ1jsVFVsUEhjYS1sIn5iZ1i6MTY3MzK4MjE4N5w1YXpwIjo1SG1h0hCeKpwNvDvcFRTyWFBZ1jsVFVsUEhjYS1sInNjbj3B1jjo1zGVmYXsdC1sIm1zcy16Imhd0H201lwXChcGku0x11233xdAuZLG06TQmLwvb2f1dGgjXC9082t1bl1sIm4C16MTY3MzK4Nt4NsW1af01joxNjczTgjMtgLCqdgk101b2GyvTM1jy02NDU2LTrNz1QtG4Ml042m210TA5nVknNgifQ.
+ E4rjnU8gYvDcP9ujkIoUx1vhve1E2k2t0978Kx_jvtqjdsNmNj2FviziJxvraUPuYTRuNF45EE0-U5GE1i1TxmfT6zswjz5-1MF1cv2HJRNSV747K1MeUuAyU8jwAT62idzpe6DixVY03xEg5Xnsd18_jJojpkaZ_xQhqcw7KVn0g_oEjdQLqBueEtCARGQ05VqAF
+ 80Lsd673M111k1nqaCze52w8y15FF2pnFoy4R8U4bIxC8qIf4jrmUKy_juowdMCOn2Rx1wn0Dby/NxFcc33cFG16k6c1mE7XK-UuUPJ-86_QntIVmDwFaef1n0JUty7CFYX8uFq1ew",
+ "scope": "default",
+ "token_type": "Bearer",
+ "expires_in": 3600

```

In the collection, click on the variables tab and add „token“ as a collection variable.

You can now add the variable {{token}} as authorization to any request. The variable will be available and the requests work after you click on „Send“ of the Token Request for the first time.

Mustangserver / mustang / ping

Save ⌘S

Send

GET {{baseUrl}}/mustang/ping

Params Authorization Headers (10) Body Pre-request Script Tests Settings Cookies

Type Bearer Token

The authorization header will be automatically generated when you send the request. Learn more about authorization ↗

① Heads up! These parameters hold sensitive data. To keep this data secure while working in a collaborative environment, we recommend using variables. [variables](#) ↗

Token {{token}}

Body Cookies Headers (18) Test Results

Pretty Raw Preview Visualize Text

1 pong

Status: 200 OK Time: 45 ms Size: 704 B Save Response

Please note that ping was answered by pong

Where appropriate, e.g. in the validation endpoint, Postman will allow you to select files, this being a valid factor-x:

## Performance Tests with Jmeter

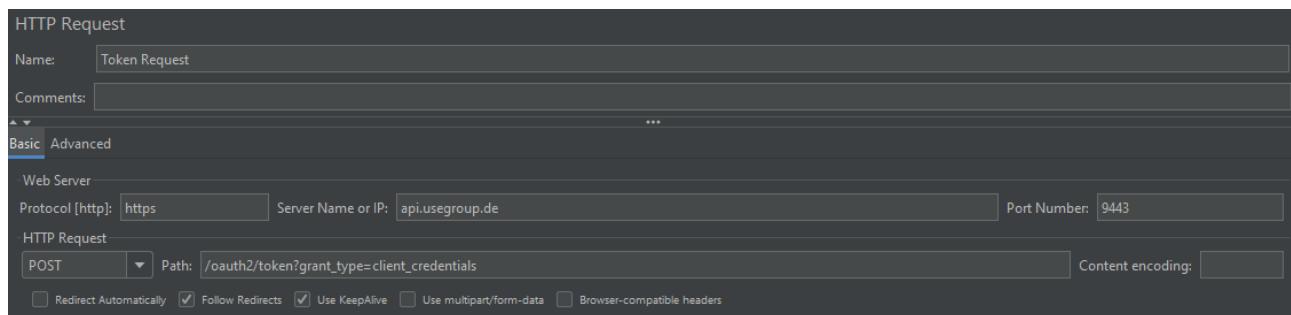
[Jmeter](#) is a generic load testing tool and load generator.

If you want to perform load tests, in order not to affect our production servers, we will happily grant you access to our mirror infrastructure, i.e. we will guarantee that the hardware, software and settings are identical.

<https://openapi-generator.tech/> supports a Jmeter export but that does not handle authentication so here we describe how to set up some Jmeter performance test manually.

For this example, you will need enabled client credentials as described on page 19 in Allowing Client Credentials.

Right click your Test plan, add a Thread Group with a Once Only controller. Below that, add a HTTP request sampler, we'll call it Token Request. This is how it is defined: Change protocol to https, method to POST, add server name and port number, and add the path:



Base64encode your <consumer key>:<consumer secret> as described on the applications page of the API management:

The screenshot shows the WSO2 API Manager Developer Portal. In the left sidebar, under 'Sandbox Keys', the 'OAuth2 Tokens' option is selected. On the main page, under 'DefaultApplication', the 'Sandbox OAuth2 Keys' section is shown. It includes fields for 'Consumer Key' (value: HiGsHBzJp5WopTSaaAgrITUIPhla) and 'Consumer Secret' (value: redacted). Below these are sections for 'Key Configurations' (Token Endpoint: https://localhost:9443/oauth2, Revoke Endpoint: https://localhost:9443/oauth2/revoke), 'Grant Types' (Refresh Token, S), and 'Callback URL'. A modal window titled 'Get CURL to Generate Access Token' is open, containing two code snippets:

```
curl -k -X POST https://localhost:9443/oauth2/token -d
"grant_type=password&username=Username&password=Password"
-H "Authorization: Basic Base64(consumer-key:consumer-secret)"
```

```
curl -k -X POST https://localhost:9443/oauth2/token -d
"grant_type=client_credentials"
-H "Authorization: Basic Base64(consumer-key:consumer-secret)"
```

**CLOSE**

In Jmeter, below the Token request, add a HeaderManager with Basic Authorization as described:

And from the response, also below Token Request, extract the JSON value access token into a Jmeter Variable access token using a JSON Extractor:

Name:	JSON Extractor
Comments:	(empty)
Apply to:	<input checked="" type="radio"/> Main sample only <input type="radio"/> Main sample and sub-samples <input type="radio"/> Sub-samples only <input type="radio"/> JMeter Variable Name to use (empty)
Names of created variables:	access_token
JSON Path expressions:	access_token
Match No. (0 for Random):	(empty)
Compute concatenation var (suffix _ALL):	<input type="checkbox"/>
Default Values:	(empty)
<b>Authorization</b>	
accept	application/json
Content-Type	application/json

Add a sampler View Result Tree to confirm the results and a debug sampler if you like (in the results tree you will then be able to e.g. see the current variables when you click on the results of the debug sampler).

Run|Start should give you green entries in the results tree.

Now we will set the ordinary authentication as header: add another Header Manager outside of the Token request and add the variable as token, i.e. Authorization being Bearer \${access\_token}

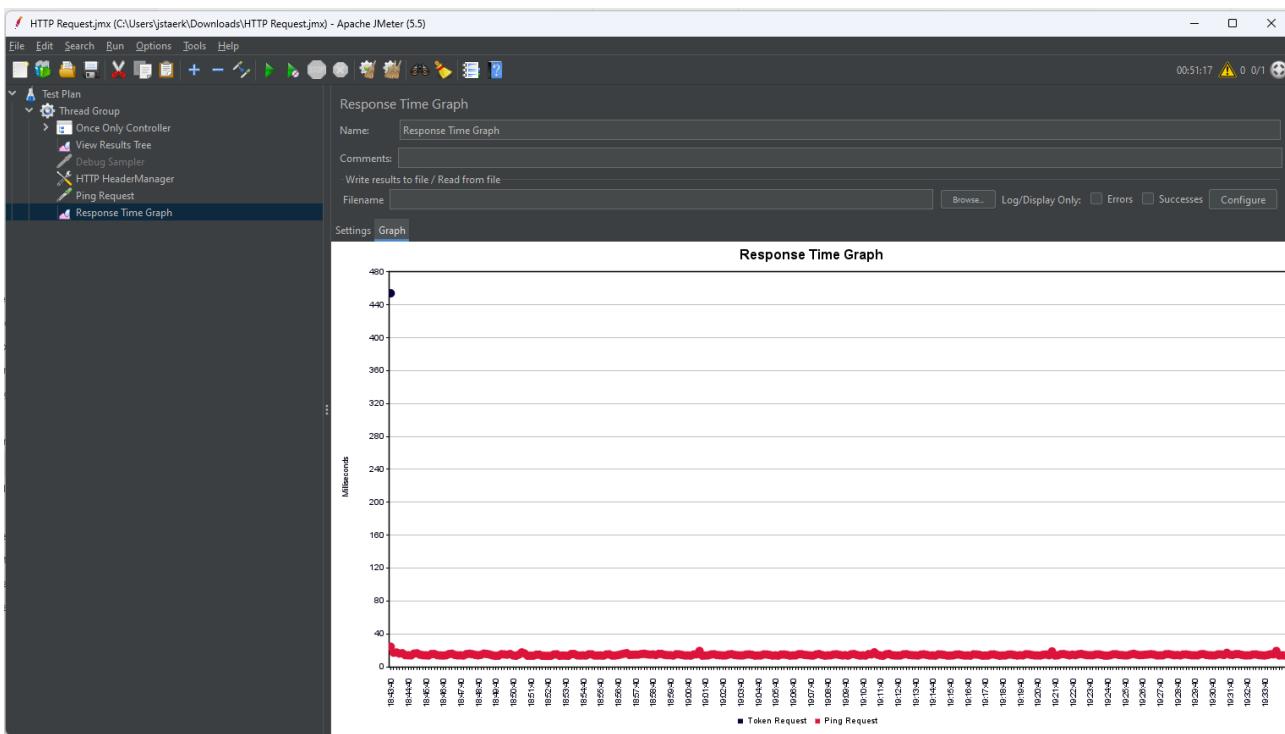
The screenshot shows the JMeter Test Plan interface. On the left, under the Thread Group, there is a 'Once Only Controller' containing a 'Token Request' sampler. Below the 'Once Only Controller' is an 'HTTP HeaderManager' sampler. To the right, the 'HTTP Header Manager' configuration window is open. It has fields for 'Name' (HTTP HeaderManager) and 'Comments'. Under 'Headers Stored in the Header Manager', there is a table:

Name:	Value
accept	application/json
Content-Type	application/json
Authorization	Bearer \${access_token}

You can then add a simple ping request in the thread group

The screenshot shows the configuration for an 'HTTP Request' sampler named 'Ping Request'. The 'Basic' tab is selected. Under 'Web Server', the protocol is set to 'https', the server name or IP is 'gw.usegroup.de', and the port number is '8243'. Under 'HTTP Request', the method is 'GET' and the path is 'mustang/v0.7.0/mustang/ping'. Other options like 'Content encoding' and various checkboxes are also visible.

and e.g. a Response time graph. You can then set the Thread group loop count to infinite, start the sampling and check the results tree. After a while the response time graph will look like this, indicating the initial login took ~440ms and the usual response time to our „ping“ is ~20ms.



## Terms of service

### Test terms

To test and evaluate the service a valid email address has to be provided. Unless otherwise agreed ([info@usegroup.de](mailto:info@usegroup.de)) test access is restricted to one account per legal entity, i.e. usually company. This email address will also be used to send availability, information about the roadmap, development and status with an expected maximum volume of one per week. You can terminate your test phase by unsubscribing from the announcements newsletter list. After the signup, access can then happen free of charge, with a limit of 1,000 operations/month, unless access is revoked by usegroup. You are not allowed to share personal data (e.g. real invoice recipient's names, addresses, email addresses, bank credentials or real invoice contents). Access may be revoked because the general test phase has ended, the test phase is over for a certain customer, or due to other terms which do not need to be disclosed. Under this test terms we also do not guarantee the availability nor the correctness of the service.

[https://api.usegroup.de:9443/authenticationendpoint/privacy\\_policy.do](https://api.usegroup.de:9443/authenticationendpoint/privacy_policy.do)

### Production terms

To access Mustangserver productively including a data processing agreement a Mustang Pro license is required. Further info can be obtained at <https://www.mustangproject.org/pro/>

## Troubleshooting

- The ping endpoint is intentionally simple and can be used to check basic functionality
- A HTTP response code 400 Post method not allowed on methods which do allow post may (temporarily) indicate a throttled user or subscription

## Inhouse variant

If you bought the inhouse version: the container registry is dev.usegroup.de:5050 and the default port the server starts on is 8000

```
docker login dev.usegroup.de:5050 -u <username> -p <token>
docker run -e MUSTANG_SERVER_VERSION=1.4.0 -dp 8000:8000 dev.usegroup.de:5050/internal/mustangserver
```

Afterwards you should be able to access

<http://<ip>:8000/swagger-ui/index.html>

and e.g. perform a ping like described above. You can leave username empty. The correct response to ping is „pong“.

The screenshot shows a Swagger UI interface for a 'mustang/ping' endpoint. The endpoint is described as a 'Healthcheck. Just request a ping, will respond with a 'pong''. The 'Parameters' section shows a single parameter 'USERNAME' of type 'string' with a '(header)' constraint. Below the parameters are 'Execute' and 'Clear' buttons. The 'Responses' section includes a 'Curl' block with the command:

```
curl -X 'GET' \
  'http://127.0.0.1:8000/mustang/ping' \
  -H 'accept: */*'
```

It also shows a 'Request URL' block with the URL <http://127.0.0.1:8000/mustang/ping>. The 'Server response' section shows a 200 status code with a 'pong' response body. There are download and copy icons next to the response body.

The screenshot shows the OpenAPI HTML Client interface. At the top, it displays a GET request to '/mustang/ping' with a description: 'Healthcheck. Just request a ping, will respond with a 'pong''. Below this is a 'Parameters' section with a single entry: 'USERNAME' (header) set to 'string'. There are 'Execute' and 'Clear' buttons. The 'Responses' section shows a 'curl' command, a 'Request URL' of 'http://127.0.0.1:8000/mustang/ping', and a 'Server response' with status code 200 and body 'pong'. There are 'Download' and 'Copy' buttons for the response.

By specifying additional -e key=value pairs (e.g. docker run -e MUSTANG\_SERVER\_VERSION=1.4.0 -e mustang.allowedIPs=123.45.67.89 -p 8888:8000 dev.usegroup.de:5050/internal/mustangserver:latest) you can set config variables.

Available vars are

Config	Default	Description
server.port	8000	The port the OpenAPI HTML Client and backend operate on
mustang.allowedIPs		If specified, a comma-separated list of IPs who will be allowed to connect
mustang.oAuth	false	Off by default
mustang.additionalLog		Additional entries for the log line of the request, can e.g. be set to the instance's ID to allow a consolidated. Logs are stored in /opt/mustangserver/log and in a future version there may be a filebeat to be configured
keycloak.enabled	false	Almost never to be set to true, even if you connect to a keycloak server, if true it will start an keycloak server on its own
spring.security.oauth2.client.registration.login-app keycloak.client-id		Oauth2 specific setting (if enabled)
spring.security.oauth2.client.registration.authorizatio keycloak.authorization-grant-type	n_code	Oauth2 specific setting (if enabled)

spring.security.oauth2.client.registration.openid_scope	keycloak.scope	Oauth2 specific setting (if enabled)
spring.security.oauth2.client.provider.keycloak.issuer-uri	http://localhost:8080/realm/	Oauth2 specific setting (if enabled)
spring.security.oauth2.client.provider.keycloak.user-name-attribute	preferred_username	Oauth2 specific setting (if enabled)
spring.security.oauth2.resourceserver.jwt.issuer-uri	http://localhost:8080/realm/	Oauth2 specific setting (if enabled)
message-from-application-properties	Die Anwendung wird in der Default Environment gestartet!	Will just be shown on the console
server.servlet.context-path		Usually not set at all but could be set to e.g. /api/v1
springdoc.api-docs.path	/api-docs	Where the openapi spec file will be found
springdoc.swagger-ui.path	/swagger-ui	Path of the HTML GUI
logging.config	classpath:logback-spring.xml	

# Version history

Of this document:

0.7.0 on 2023-01-19 by Jochen.

0.8.0 on 2023-02-25 by Jochen: Added Mustangserver 0.8.0 (=Order-X)

0.8.1 on 2023-02-26 by Jochen: added C++-Client, PDF/A-param to PDF endpoint

1.0.0 on 2023-09-26 by Jochen: most recent endpoint, neccessity to subscribe, split between mustangserver and mustangserver-docs, updated list of Ves-IDs, added change password url

1.1.0 added username field, exception handling, invoice2XML parameter standard was renamed to format. Better Exception handling. CombineXML now also allows PDF/A-3 input.

1.2.0 on 2024-01-31 /combineXML /parse and /invoice2XML now support XRechnung 3.0.1

1.3.0 on 2024-02-29 Endpoint /combine is now available again (combine JSON and PDF to fx), /phive has been updated, now auto-detects Ves-IDs if none specified and now also supports e.g. XR 3.0.1 (from 135 to 143 VesIDs). /xmltohtml and API keys documented. Mentioned Bruno.

1.3.1 on 2024-07-29 JSON structure documented, added Troubleshooting

1.4.0 added description of detach and xmltopdf endpoint and of C# sample client. Added section classes.

1.4.1 added chapter on Docker config for in-house server. Corrected timezone in example. Added BT Mapping and invoice class description.