Inhaltsverzeichnis

In case of technical questions

Please contact

Jochen Stärk
jstaerk@usegroup.de

or

Martin Wachtveitl
mwachtveitl@usegroup.de

Server access

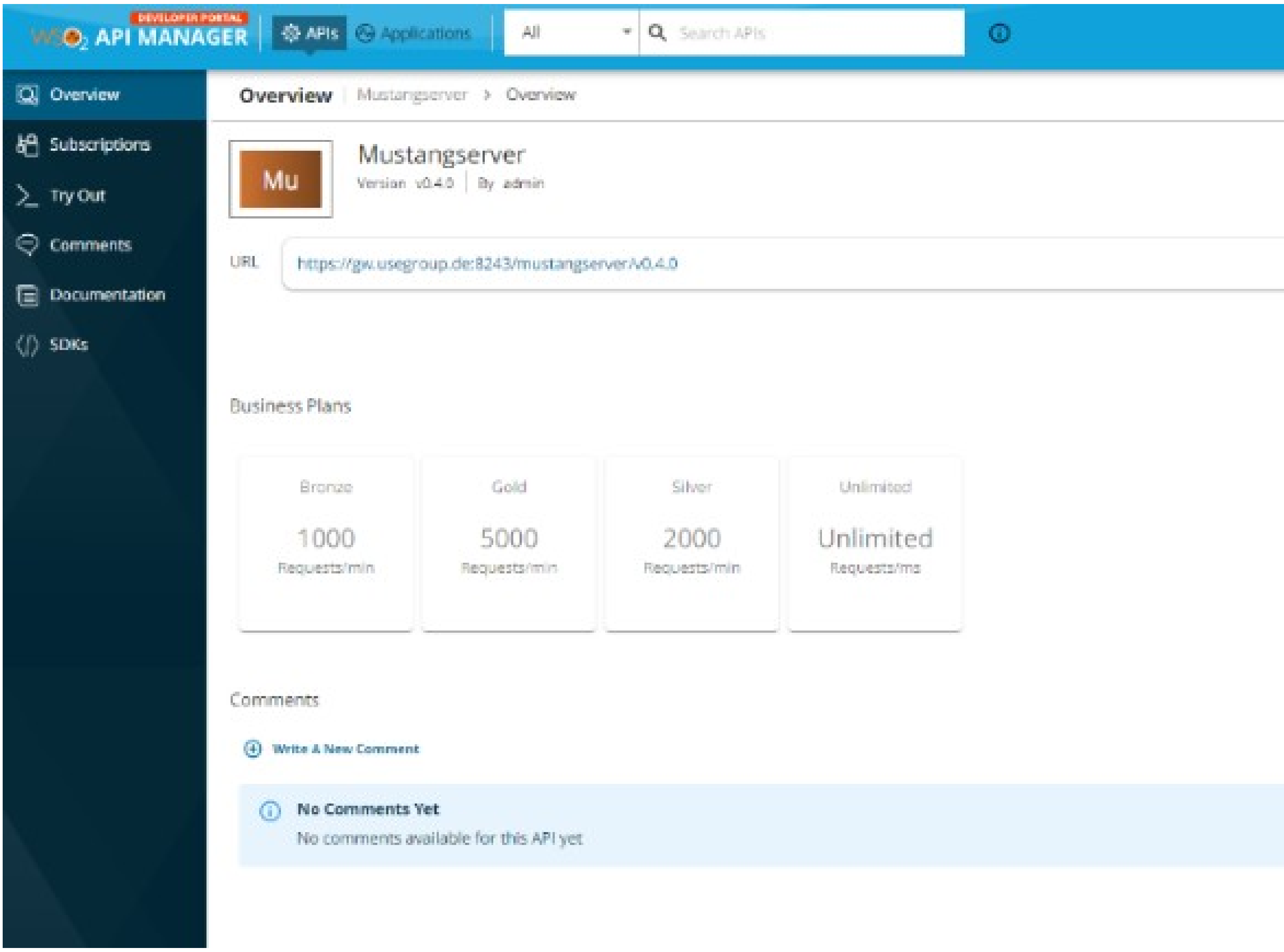
User access to the web interface is possible via https://api.usegroup.de/. Terms of service are listed in the chapter Terms of service on page 33.

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 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.



Mustangserver
Version v0.4.0 | By admin

URL https://gw.usegroup.de:8243/mustangserver/v0.4.0

Business Plans

Bronze	Gold	Silver	Unlimited
1000 Requests/min	5000 Requests/min	2000 Requests/min	Unlimited Requests/ms

Comments

No Comments Yet
No comments available for this API 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 a 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

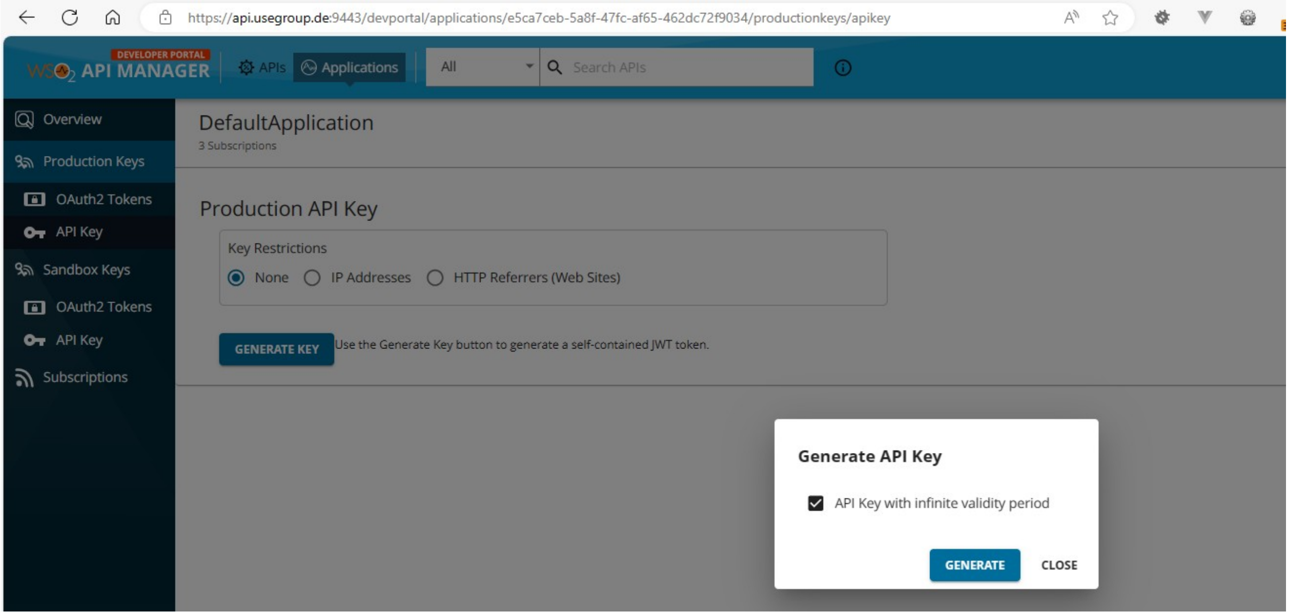
First of all you will have to subscribe to the API. You can manage your subscriptions in the left navigation of the devportal but it’s often easiest to subscribe via overview|try out, which also allows you to try it.

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 detailled in the PHP Client chapter „Allowing Client Credentials“ on page 16 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.



https://api.usegroup.de:9443/devportal/applications/e5ca7ceb-5a8f-47fc-af65-462dc72f9034/productionkeys/apikey

DefaultApplication

3 Subscriptions

Production API Key

Key Restrictions

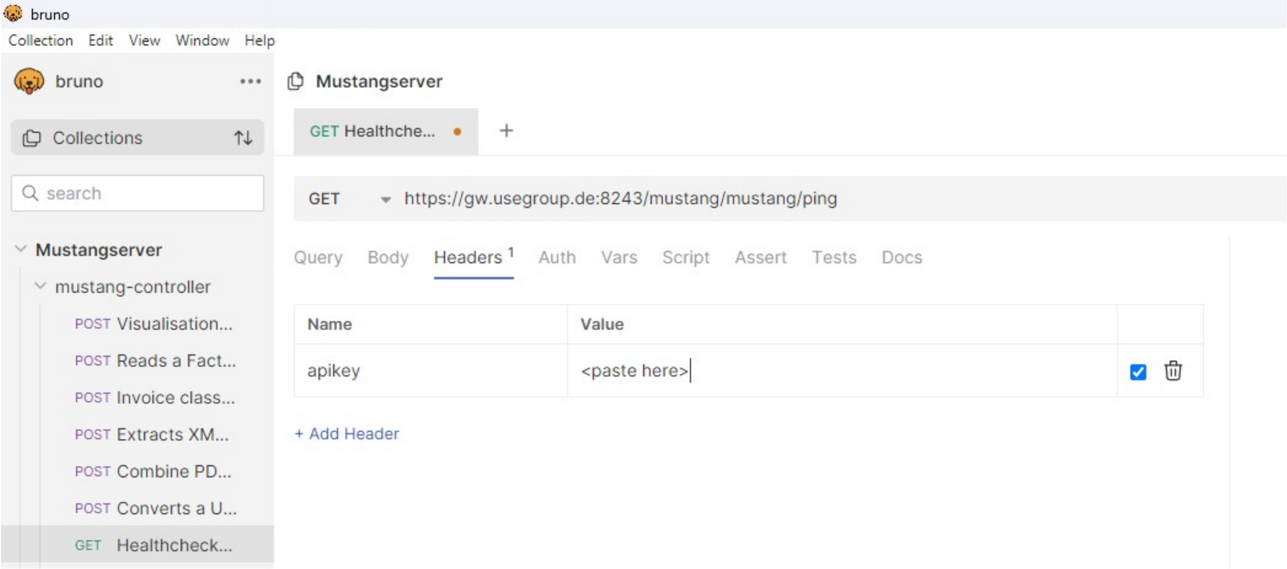
● None ○ IP Addresses ○ HTTP Referrers (Web Sites)

Use the Generate Key button to generate a self-contained JWT token.

Generate API Key

☑ API Key with infinite validity period

An API key can be used e.g. within Bruno (pp 27ff)



GET https://gw.usegroup.de:8243/mustang/mustang/ping

Name	Value
apikey	<paste here>

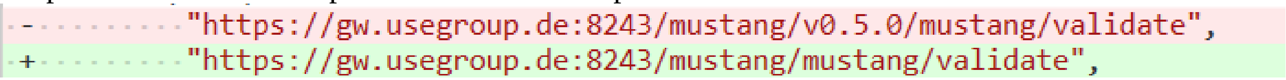
Within PHP you also might pass just as additional header attribute making your source code look like

```
 <?php

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

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

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

try {
    $result = $apiInstance->ping();
    print_r($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 .

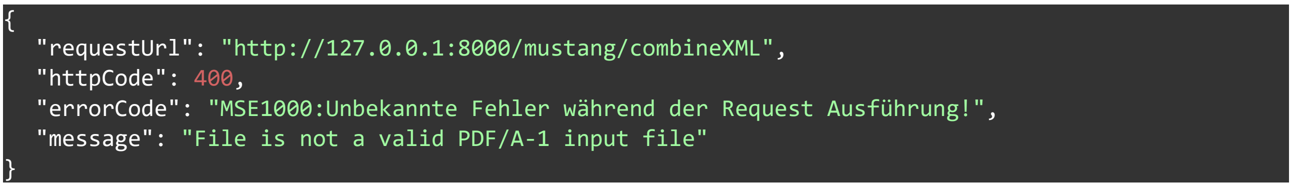
New versions may be retired as soon as six month after release of the successor. It is possible to always use the latest (more precisely: recommended) version by removing the version from the endpoint. This is an example of the validate endpoint:



```
-         "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 WSO² 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 WSO²‘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, parallely 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 diffrence 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 wrinting. 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.

Operations/Endpoints

Mustangserver’s available operations are

Ping

Just a test, always just returns „pong“. The only operation acessible 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:cii:1.2.1
de.xrechnung:cii:1.2.2
de.xrechnung:cii:2.0.0
de.xrechnung:cii:2.0.1
de.xrechnung:cii:2.1.1
de.xrechnung:cii:2.2.0
de.xrechnung:cii:2.3.1
de.xrechnung:cii:3.0.0
de.xrechnung:cii:3.0.1
de.xrechnung:cii:3.0.2
de.xrechnung:ubl-creditnote:1.2.0
de.xrechnung:ubl-creditnote:1.2.1
de.xrechnung:ubl-creditnote:1.2.2
de.xrechnung:ubl-creditnote:2.0.0
de.xrechnung:ubl-creditnote:2.0.1
de.xrechnung:ubl-creditnote:2.1.1
de.xrechnung:ubl-creditnote:2.2.0
de.xrechnung:ubl-creditnote:2.3.1
de.xrechnung:ubl-creditnote:3.0.0
de.xrechnung:ubl-creditnote:3.0.1
de.xrechnung:ubl-creditnote:3.0.2
de.xrechnung:ubl-invoice:1.2.0
de.xrechnung:ubl-invoice:1.2.1
de.xrechnung:ubl-invoice:1.2.2
de.xrechnung:ubl-invoice:2.0.0
de.xrechnung:ubl-invoice:2.0.1
de.xrechnung:ubl-invoice:2.1.1
de.xrechnung:ubl-invoice:2.2.0
de.xrechnung:ubl-invoice:2.3.1
de.xrechnung:ubl-invoice:3.0.0
de.xrechnung:ubl-invoice:3.0.1
de.xrechnung:ubl-invoice:3.0.2
es.gob:facturae:3.0.0
es.gob:facturae:3.1.0
es.gob:facturae:3.2.0
es.gob:facturae:3.2.1
es.gob:facturae:3.2.2
eu.cen.en16931:cii:1.0.0
eu.cen.en16931:cii:1.1.0
eu.cen.en16931:cii:1.2.0
eu.cen.en16931:cii:1.2.1
eu.cen.en16931:cii:1.2.3
eu.cen.en16931:cii:1.3.0
eu.cen.en16931:cii:1.3.1
eu.cen.en16931:cii:1.3.10
eu.cen.en16931:cii:1.3.11
eu.cen.en16931:cii:1.3.12
eu.cen.en16931:cii:1.3.2
eu.cen.en16931:cii:1.3.3
eu.cen.en16931:cii:1.3.4
eu.cen.en16931:cii:1.3.5
eu.cen.en16931:cii:1.3.6
eu.cen.en16931:cii:1.3.6; qualifier=a

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

eu.peppol.bis3:order-response-advanced:2024.5.0
eu.peppol.bis3:order-response:2023.11.0
eu.peppol.bis3:order-response:2023.5.0
eu.peppol.bis3:order-response:2024.5.0
eu.peppol.bis3:order:2023.11.0
eu.peppol.bis3:order:2023.5.0
eu.peppol.bis3:order:2024.5.0
eu.peppol.bis3:punch-out:2023.11.0
eu.peppol.bis3:punch-out:2023.5.0
eu.peppol.bis3:punch-out:2024.5.0
eu.peppol.directory:businesscard:1.0.0
eu.peppol.directory:businesscard:2.0.0
eu.peppol.directory:businesscard:3.0.0
eu.peppol.reporting:eusr:1.1.4
eu.peppol.reporting:eusr:1.1.5
eu.peppol.reporting:tsr:1.0.4
eu.peppol.reporting:tsr:1.0.5
it.fatturapa:invoice:1.2.0
it.fatturapa:invoice:1.2.1
it.fatturapa:invoice:1.2.2
org.peppol.jp.pint:credit-note:0.1.2
org.peppol.jp.pint:invoice:0.1.2
org.peppol.pint.my:creditnote-self-billing:1.0.0
org.peppol.pint.my:creditnote:1.0.0
org.peppol.pint.my:invoice-self-billing:1.0.0
org.peppol.pint.my:invoice:1.0.0
org.peppol.pint:credit-note:1.0.0
org.peppol.pint:credit-note:1.0.1
org.peppol.pint:invoice:1.0.0
org.peppol.pint:invoice:1.0.1

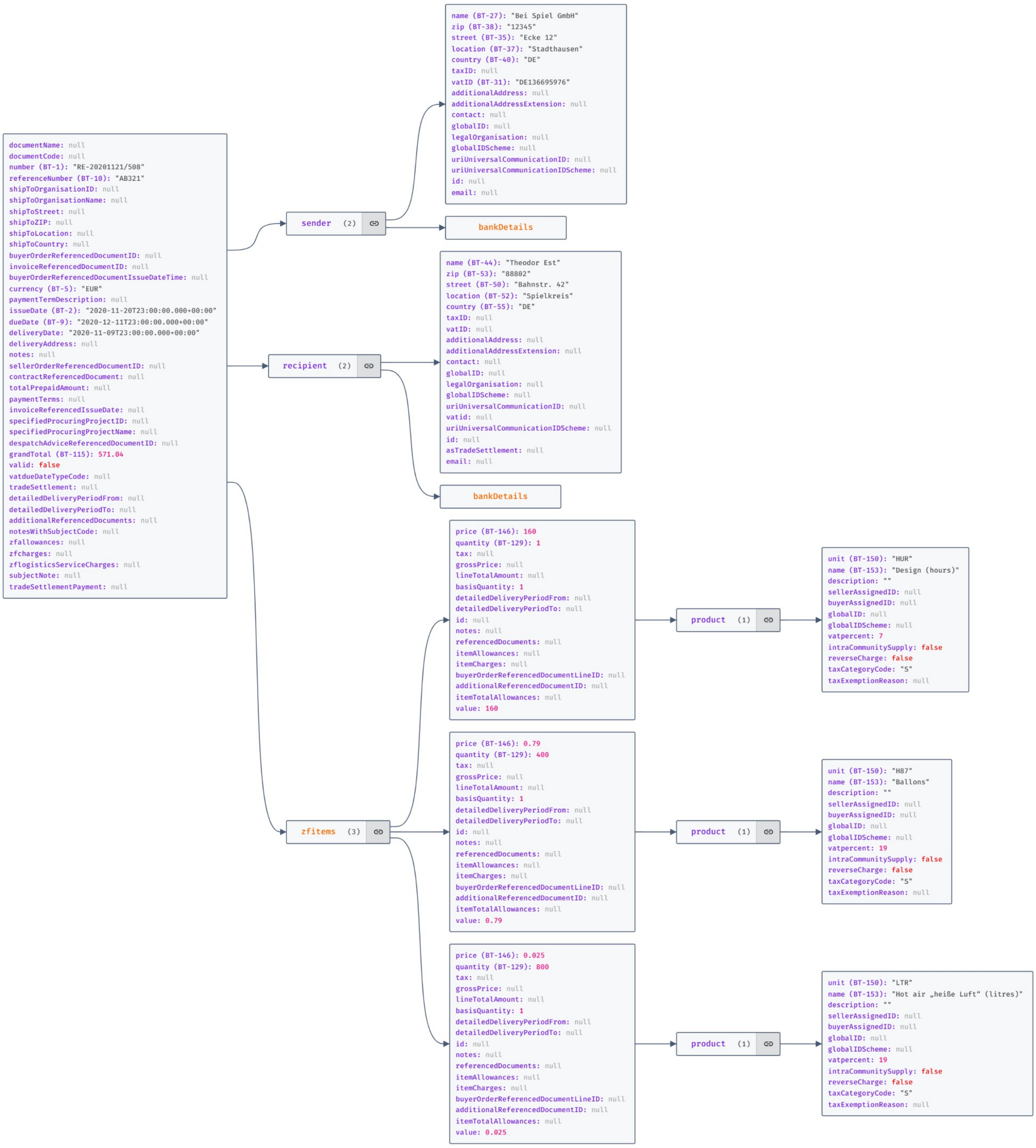
pdf

(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.

The object will look as follows:



```
documentName: null
documentCode: null
number (BT-1): "RE-20201121/508"
referenceNumber (BT-10): "AB321"
shipToOrganisationID: null
shipToOrganisationName: null
shipToStreet: null
shipToZIP: null
shipToLocation: null
shipToCountry: null
buyerOrderReferencedDocumentID: null
invoiceReferencedDocumentID: null
buyerOrderReferencedDocumentIssueDateTime: null
currency (BT-5): "EUR"
paymentTermDescription: null
issueDate (BT-2): "2020-11-20T23:00:00.000+00:00"
dueDate (BT-9): "2020-12-11T23:00:00.000+00:00"
deliveryDate: "2020-11-09T23:00:00.000+00:00"
deliveryAddress: null
notes: null
sellerOrderReferencedDocumentID: null
contractReferencedDocument: null
totalPrepaidAmount: null
paymentTerms: null
invoiceReferencedIssueDate: null
specifiedProcuringProjectID: null
specifiedProcuringProjectName: null
despatchAdviceReferencedDocumentID: null
grandTotal (BT-115): 571.04
valid: false
vatdueDateTypeCode: null
tradeSettlement: null
detailedDeliveryPeriodFrom: null
detailedDeliveryPeriodTo: null
additionalReferencedDocuments: null
notesWithSubjectCode: null
zfallowances: null
zfcharges: null
zflogisticsServiceCharges: null
subjectNote: null
tradeSettlementPayment: null
```

```
name (BT-27): "Bei Spiel GmbH"
zip (BT-38): "12345"
street (BT-35): "Ecke 12"
location (BT-37): "Stadthausen"
country (BT-40): "DE"
taxID: null
vatID (BT-31): "DE136695976"
additionalAddress: null
additionalAddressExtension: null
contact: null
globalID: null
legalOrganisation: null
globalIDScheme: null
uriUniversalCommunicationID: null
uriUniversalCommunicationIDScheme: null
id: null
email: null
```

```
name (BT-44): "Theodor Est"
zip (BT-53): "88802"
street (BT-50): "Bahnstr. 42"
location (BT-52): "Spielkreis"
country (BT-55): "DE"
taxID: null
vatID: null
additionalAddress: null
additionalAddressExtension: null
contact: null
globalID: null
legalOrganisation: null
globalIDScheme: null
uriUniversalCommunicationID: null
vatid: null
uriUniversalCommunicationIDScheme: null
id: null
asTradeSettlement: null
email: null
```

```
price (BT-146): 160
quantity (BT-129): 1
tax: null
grossPrice: null
lineTotalAmount: null
basisQuantity: 1
detailedDeliveryPeriodFrom: null
detailedDeliveryPeriodTo: null
id: null
notes: null
referencedDocuments: null
itemAllowances: null
itemCharges: null
buyerOrderReferencedDocumentLineID: null
additionalReferencedDocumentID: null
itemTotalAllowances: null
value: 160
```

```
unit (BT-150): "HUR"
name (BT-153): "Design (hours)"
description: ""
sellerAssignedID: null
buyerAssignedID: null
globalID: null
globalIDScheme: null
vatpercent: 7
intraCommunitySupply: false
reverseCharge: false
taxCategoryCode: "S"
taxExemptionReason: null
```

```
price (BT-146): 0.79
quantity (BT-129): 400
tax: null
grossPrice: null
lineTotalAmount: null
basisQuantity: 1
detailedDeliveryPeriodFrom: null
detailedDeliveryPeriodTo: null
id: null
notes: null
referencedDocuments: null
itemAllowances: null
itemCharges: null
buyerOrderReferencedDocumentLineID: null
additionalReferencedDocumentID: null
itemTotalAllowances: null
value: 0.79
```

```
unit (BT-150): "H87"
name (BT-153): "Ballons"
description: ""
sellerAssignedID: null
buyerAssignedID: null
globalID: null
globalIDScheme: null
vatpercent: 19
intraCommunitySupply: false
reverseCharge: false
taxCategoryCode: "S"
taxExemptionReason: null
```

```
price (BT-146): 0.025
quantity (BT-129): 800
tax: null
grossPrice: null
lineTotalAmount: null
basisQuantity: 1
detailedDeliveryPeriodFrom: null
detailedDeliveryPeriodTo: null
id: null
notes: null
referencedDocuments: null
itemAllowances: null
itemCharges: null
buyerOrderReferencedDocumentLineID: null
additionalReferencedDocumentID: null
itemTotalAllowances: null
value: 0.025
```

```
unit (BT-150): "LTR"
name (BT-153): "Hot air „heiße Luft“ (litres)"
description: ""
sellerAssignedID: null
buyerAssignedID: null
globalID: null
globalIDScheme: null
vatpercent: 19
intraCommunitySupply: false
reverseCharge: false
taxCategoryCode: "S"
taxExemptionReason: null
```

Schaubild 1: Sample read invoice JSON structure

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 á 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.

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".

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 factur-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":"iVBORw0KGgoAAAANSUhEUgAAAAgAAAC1CAQAAADIUnarAAAABGdBTUEAALGPC/
xhBQAAACBjSFJNAAB6JgAAgIQAAPoAAACA6AAAdTAAAOpgAAA6mAAAF3CculE8AAAAAmJLR0QA/
4ePzL8AAAAJcEhZcwAACxMAAAsTAQCanBgAAAAHdElNRQfkAQIXGQ0qsHJfAAAAo0lEQVRIx+3MoQrCUBjF8f+9dzBtgh
a72eAYLJgMQ4tvMREfQuw+wGA2mxarxTeYgi9gEwyDNaOfYTrM5u+kw4/
DcQAwi4PO+Q5gK2AsUVW+UEdBQUFBQUFBQUFB4W9IQhMRL3oABpKppLaLmEIm2cXMR6897c++cEM3WBPUBw2eVrzfR/
Gtd6Cs4SFHl1/
DFn18oDTLbOcgPwWFacpNVpstvAG3bSYVfhBdGAAAACV0RVh0ZGF0ZTpjcmVhdGUAMjAyMC0wMS0wMlQyMzoyNToxMysw
MDowMEN9AywAAAAldEVYdGRhdGU6bW9kaWZ5ADIwMjAtMDEtMDJUMjM6MjU6MTMrMDA6MDAyILuQAAAAAElFTkSuQmCC"
},{"sameFileB.png":"iVBORw0KGgoAAAANSUhEUgAAAAgAAAC1CAQAAADIUnarAAAABGdBTUEAALGPC/
xhBQAAACBjSFJNAAB6JgAAgIQAAPoAAACA6AAAdTAAAOpgAAA6mAAAF3CculE8AAAAAmJLR0QA/
4ePzL8AAAAJcEhZcwAACxMAAAsTAQCanBgAAAAHdElNRQfkAQIXGQ0qsHJfAAAAo0lEQVRIx+3MoQrCUBjF8f+9dzBtgh
a72eAYLJgMQ4tvMREfQuw+wGA2mxarxTeYgi9gEwyDNaOfYTrM5u+kw4/
DcQAwi4PO+Q5gK2AsUVW+UEdBQUFBQUFBQUFB4W9IQhMRL3oABpKppLaLmEIm2cXMR6897c++cEM3WBPUBw2eVrzfR/
Gtd6Cs4SFHl1/
DFn18oDTLbOcgPwWFacpNVpstvAG3bSYVfhBdGAAAACV0RVh0ZGF0ZTpjcmVhdGUAMjAyMC0wMS0wMlQyMzoyNToxMysw
MDowMEN9AywAAAAldEVYdGRhdGU6bW9kaWZ5ADIwMjAtMDEtMDJUMjM6MjU6MTMrMDA6MDAyILuQAAAAAElFTkSuQmCC"
}]
}
```

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 9) 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:

```
{
  "number": "471102",
  "currency": "EUR",
  "issueDate": "2018-03-04T23:00:00.000+00:00",
  "dueDate": "2018-03-04T23:00:00.000+00:00",
  "deliveryDate": "2018-03-04T23:00:00.000+00:00",
  "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"
      }
    }
  ]
}
```

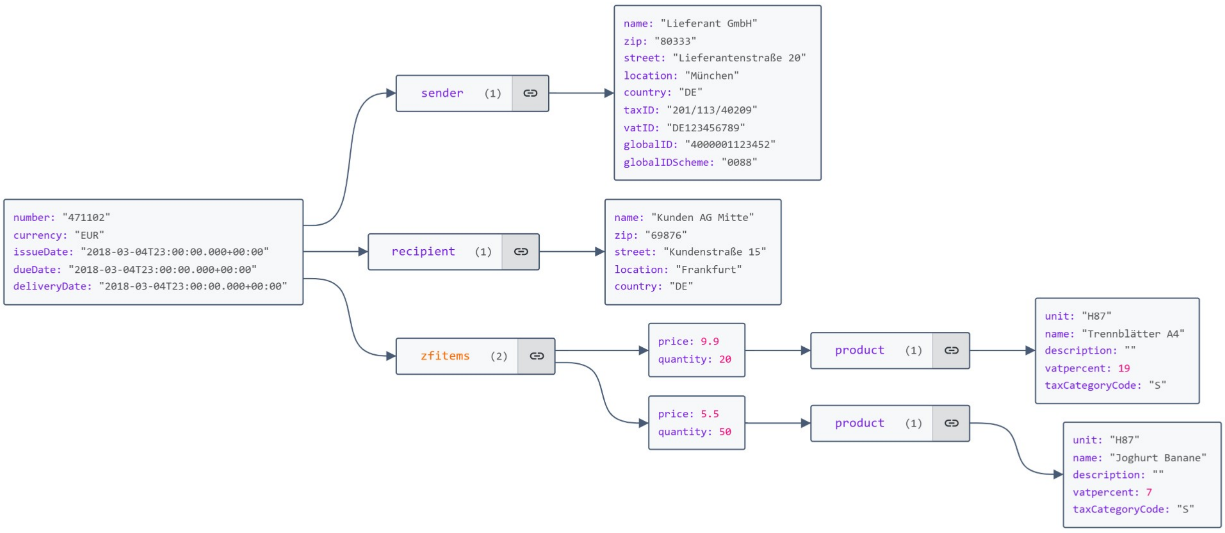


Schaubild 2: Sample JSON structure to write invoices

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 specfied 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

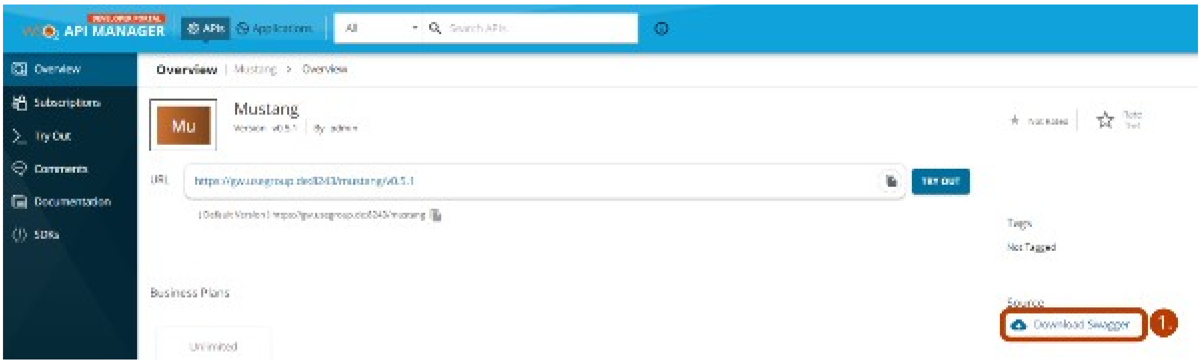
This example operates in a PHP context but https://editor.swagger.io/ also allows

C#, Dart, HTML, Go, Java, Javascript, Kotlin, Python, R, Ruby, Scala, Swift and Typescript.

Mustangserver Hello World

Preparations

Screenshot 1:

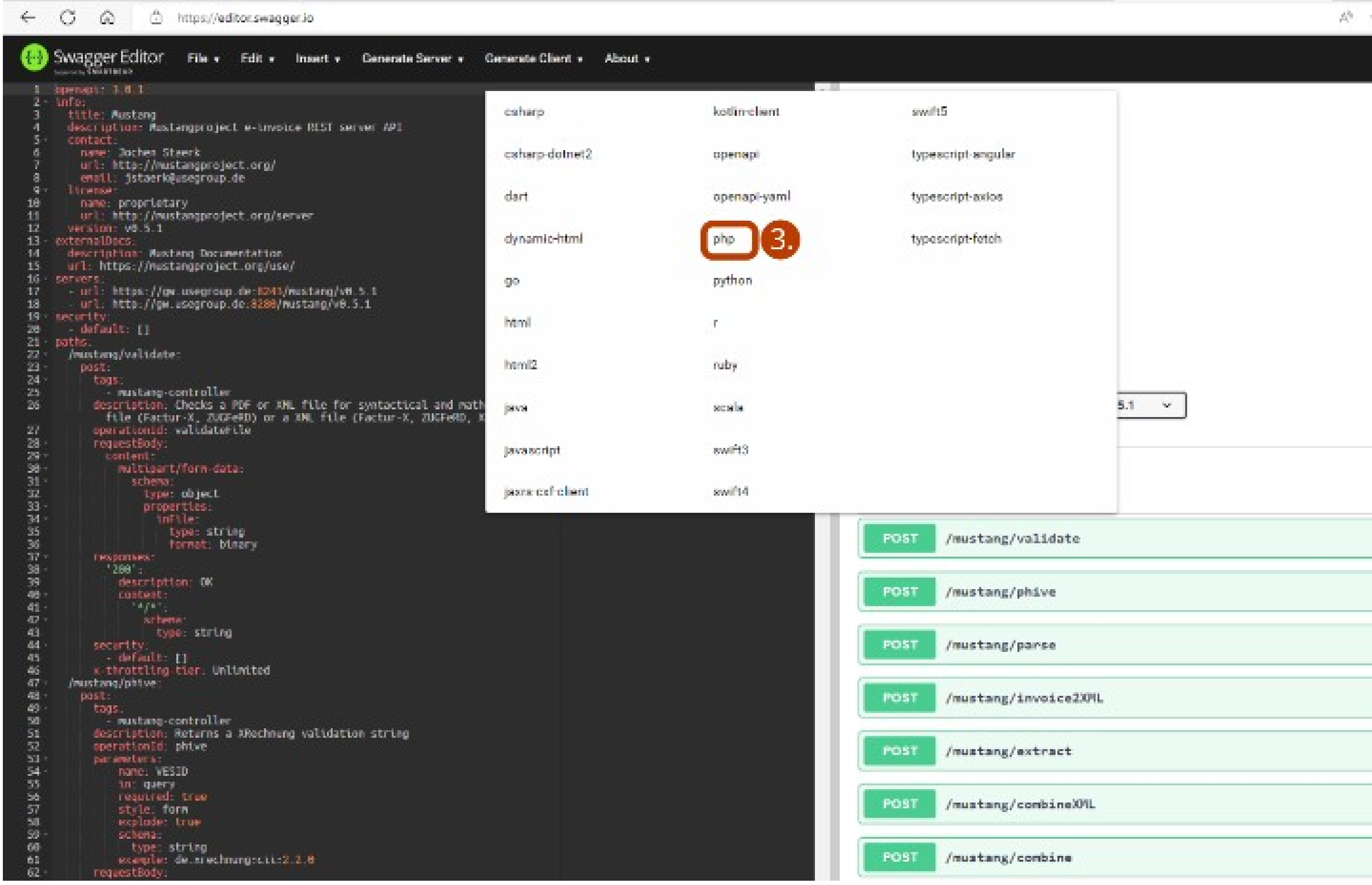


#1. Log in on https://api.usegroup.de/devportal/ , select the latest Mustangserver API and download the OpenAPI (=Swagger) definition of the API (->1.)
#2. Open the file in a text editor, select all and copy
#3. Go to editor.swagger.io, paste the definition and confirm conversion to yaml. Select Generate 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 bei 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.

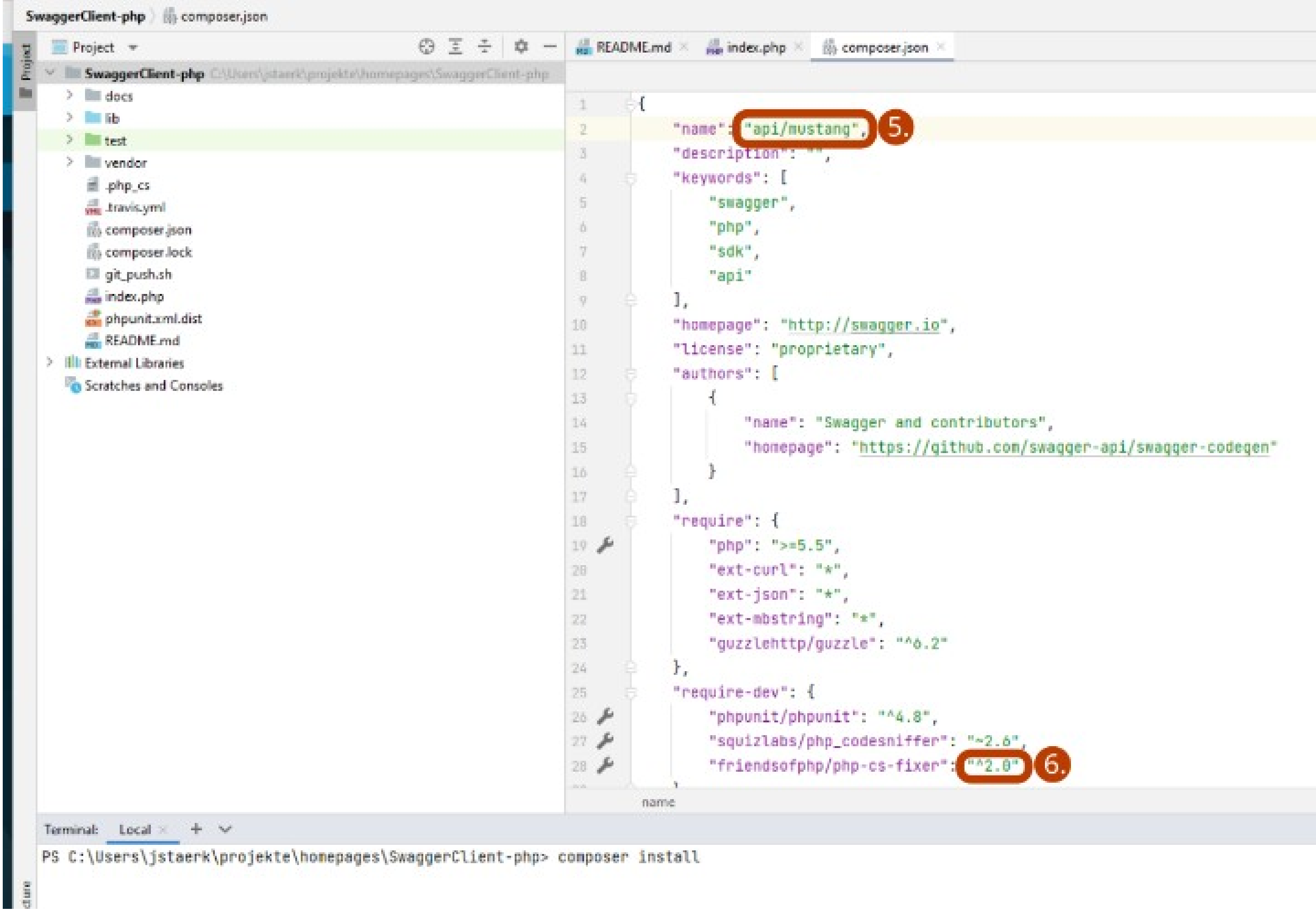
Screenshot 2:



```
openapi: 3.0.1
info:
  title: Mustang
  description: Mustangproject e-invoice REST server API
  contact:
    name: Jochen Staerk
    url: http://mustangproject.org/
    email: jstaerk@usegroup.de
  license:
    name: proprietary
    url: http://mustangproject.org/server
  version: v0.5.1
externalDocs:
  description: Mustang Documentation
  url: https://mustangproject.org/use/
servers:
  - url: https://gw.usegroup.de:8243/mustang/v0.5.1
  - url: http://gw.usegroup.de:8280/mustang/v0.5.1
security:
  - default: []
paths:
  /mustang/validate:
    post:
      tags:
        - mustang-controller
      description: Checks a PDF or XML file for syntactical and math
        file (Factur-X, ZUGFeRD) or a XML file (Factur-X, ZUGFeRD, X
      operationId: validateFile
      requestBody:
        content:
          multipart/form-data:
            schema:
              type: object
              properties:
                inFile:
                  type: string
                  format: binary
      responses:
        '200':
          description: OK
          content:
            '*/*':
              schema:
                type: string
      security:
        - default: []
      x-throttling-tier: Unlimited
  /mustang/phive:
    post:
      tags:
        - mustang-controller
      description: Returns a XRechnung validation string
      operationId: phive
      parameters:
          name: VESID
          in: query
          required: true
          style: form
          explode: true
          schema:
            type: string
          example: de.xrechnung:cii:2.2.0
      requestBody:
```

#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:



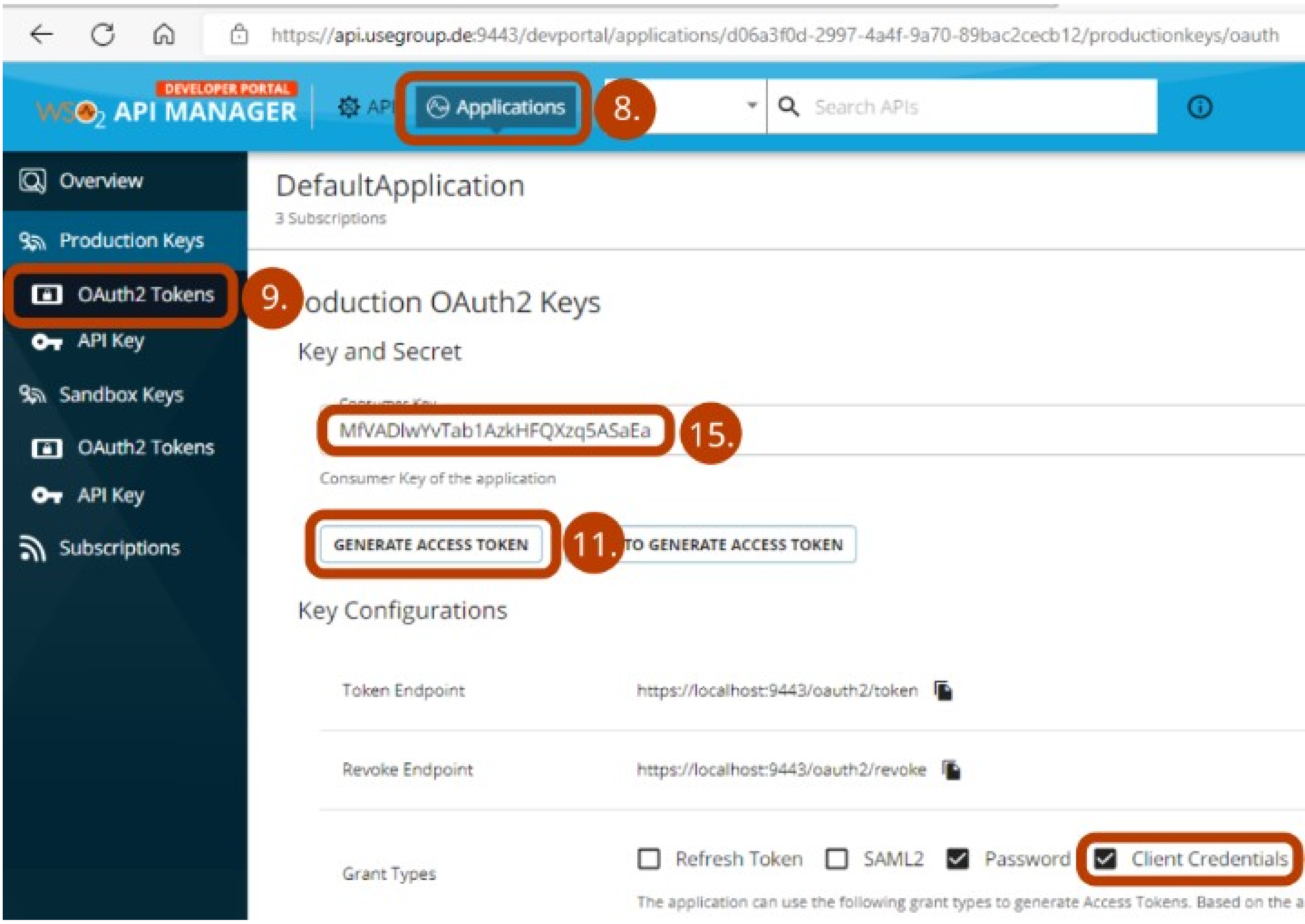
```
{
    "name": "api/mustang",
    "description": "",
    "keywords": [
        "swagger",
        "php",
        "sdk",
        "api"
    ],
    "homepage": "http://swagger.io",
    "license": "proprietary",
    "authors": [
        {
            "name": "Swagger and contributors",
            "homepage": "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"
```

```
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:



https://api.usegroup.de:9443/devportal/applications/d06a3f0d-2997-4a4f-9a70-89bac2cecb12/productionkeys/oauth

DefaultApplication
3 Subscriptions

Production OAuth2 Keys

Key and Secret

Consumer Key: MfVADlwYvTab1AzkHFQXzq5ASaEa
Consumer Key of the application

Key Configurations

Token Endpoint: https://localhost:9443/oauth2/token

Revoke Endpoint: https://localhost:9443/oauth2/revoke

Grant Types: □ Refresh Token □ SAML2 ☑ Password ☑ Client Credentials
The application can use the following grant types to generate Access Tokens. Based on the a

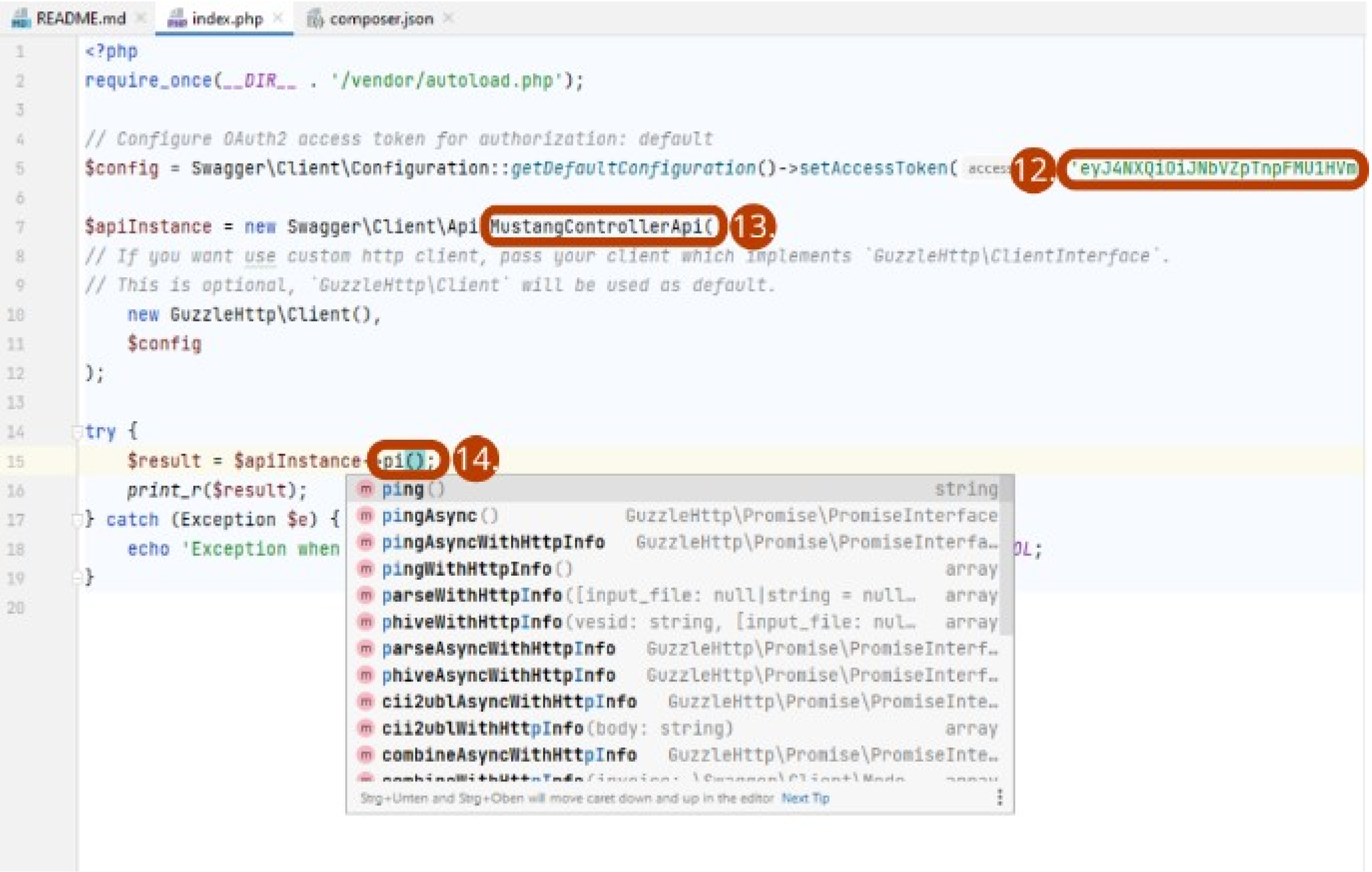
#8. Click on Applications (8.), Default Application,
#9. Production Keys/OAuth2 Token (9.).

#10. As preparation for authetication: 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:

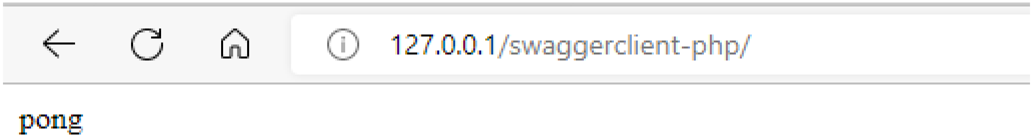


```php
<?php
require_once(__DIR__ . '/vendor/autoload.php');

// Configure OAuth2 access token for authorization: default
$config = Swagger\Client\Configuration::getDefaultConfiguration()->setAccessToken( access: 'eyJ4NXQiOiJNbVZpTnpFMU1HVm

$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 = $apiInstance->pi();
    print_r($result);
} catch (Exception $e) {
    echo 'Exception when                                                                      OL;
}
```

```
m ping()                                                    string
m pingAsync()              GuzzleHttp\Promise\PromiseInterface
m pingAsyncWithHttpInfo     GuzzleHttp\Promise\PromiseInterfa…
m pingWithHttpInfo()                                        array
m parseWithHttpInfo([input_file: null|string = null…        array
m phiveWithHttpInfo(vesid: string, [input_file: nul…        array
m parseAsyncWithHttpInfo    GuzzleHttp\Promise\PromiseInterf…
m phiveAsyncWithHttpInfo    GuzzleHttp\Promise\PromiseInterf…
m cii2ublAsyncWithHttpInfo   GuzzleHttp\Promise\PromiseInte…
m cii2ublWithHttpInfo(body: string)                         array
m combineAsyncWithHttpInfo   GuzzleHttp\Promise\PromiseInte…
m combineWithHttpInfo(invoice: \Swagger\Client\Mode…        array
Strg+Unten and Strg+Oben will move caret down and up in the editor  Next Tip
```

#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 processort in your browser, it should now look like Screenshot 6:



```
pong
```

OAuth2 Authentication

Back to Screenshot 4:

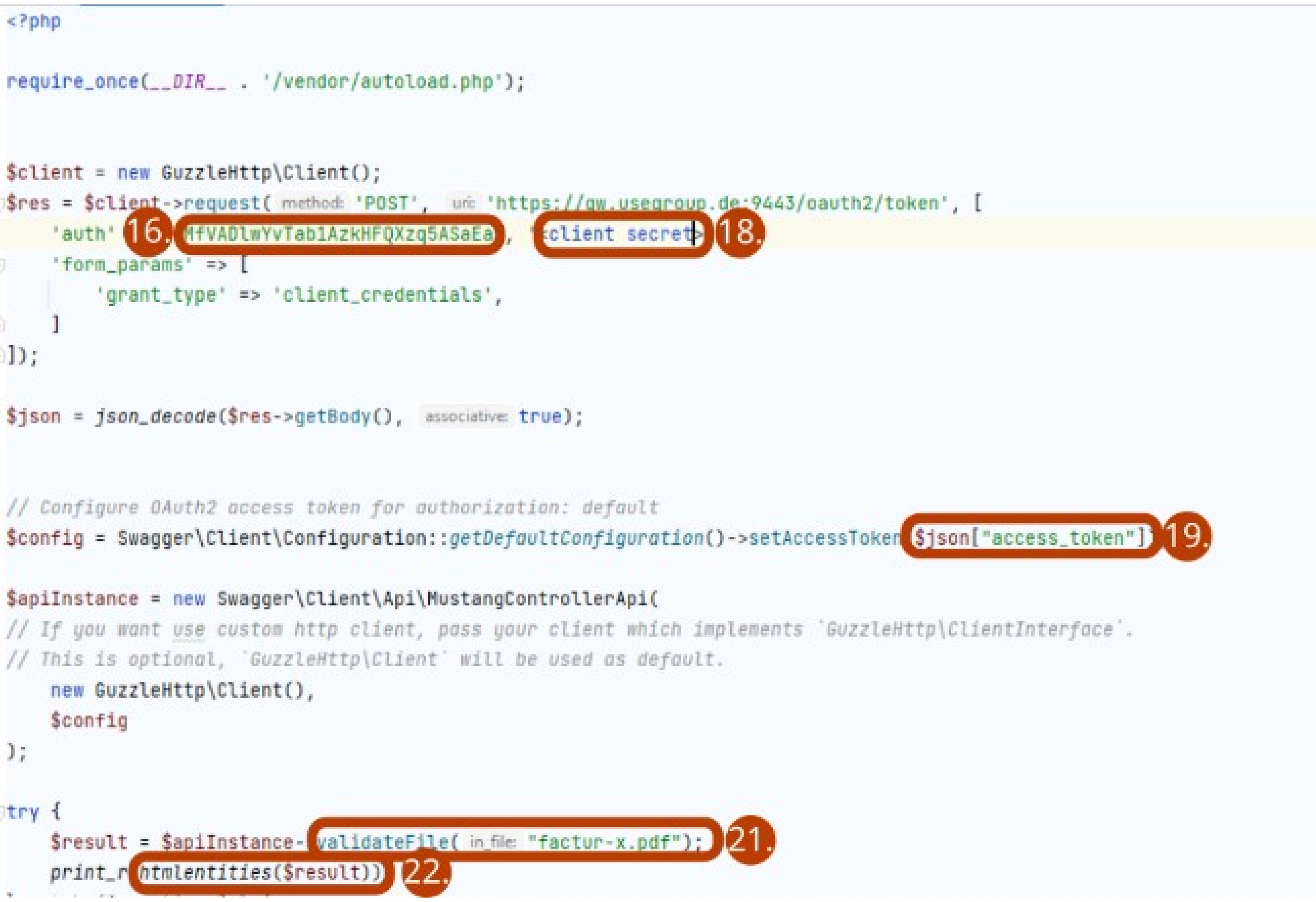
In index.php paste the following code

$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',  uri: 'https://gw.usegroup.de:9443/oauth2/token', [
    'auth'    MfVADlwYvTab1AzkHFQXzq5ASaEa ,  <client secret>
    '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"])

$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 = $apiInstance->validateFile( in_file: "factur-x.pdf");
    print_r(htmlentities($result));
```

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

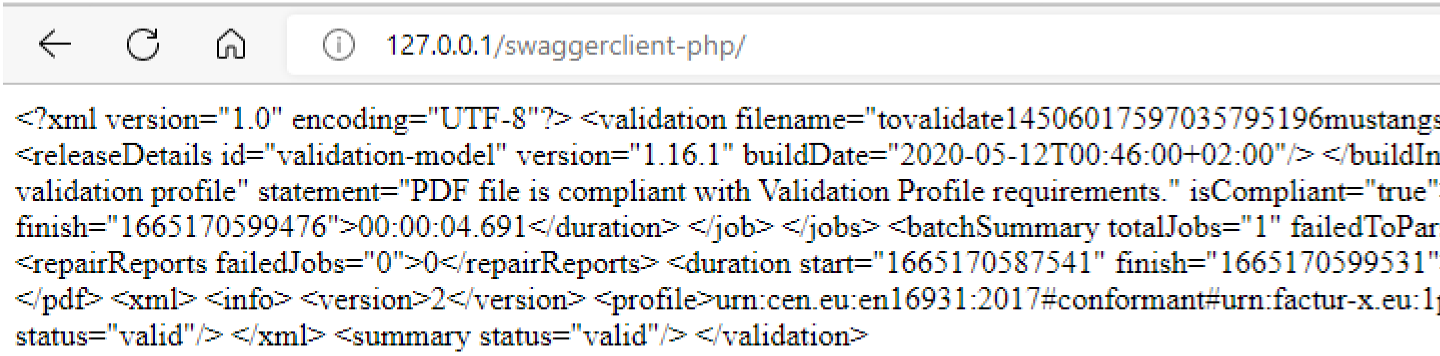
#17. reveal and copy Consumer Secret (18.).

#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 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:

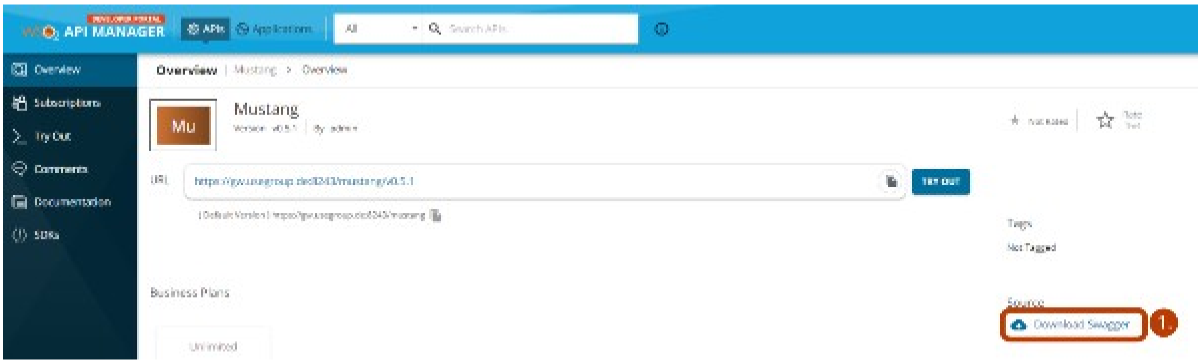


```
<?xml version="1.0" encoding="UTF-8"?> <validation filename="tovalidate14506017597035795196mustangs
<releaseDetails id="validation-model" version="1.16.1" buildDate="2020-05-12T00:46:00+02:00"/> </buildIn
validation profile" statement="PDF file is compliant with Validation Profile requirements." isCompliant="true"
finish="1665170599476">00:00:04.691</duration> </job> </jobs> <batchSummary totalJobs="1" failedToPar
<repairReports failedJobs="0">0</repairReports> <duration start="1665170587541" finish="1665170599531"
</pdf> <xml> <info> <version>2</version> <profile>urn:cen.eu:en16931:2017#conformant#urn:factur-x.eu:1
status="valid"/> </xml> <summary status="valid"/> </validation>
```

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

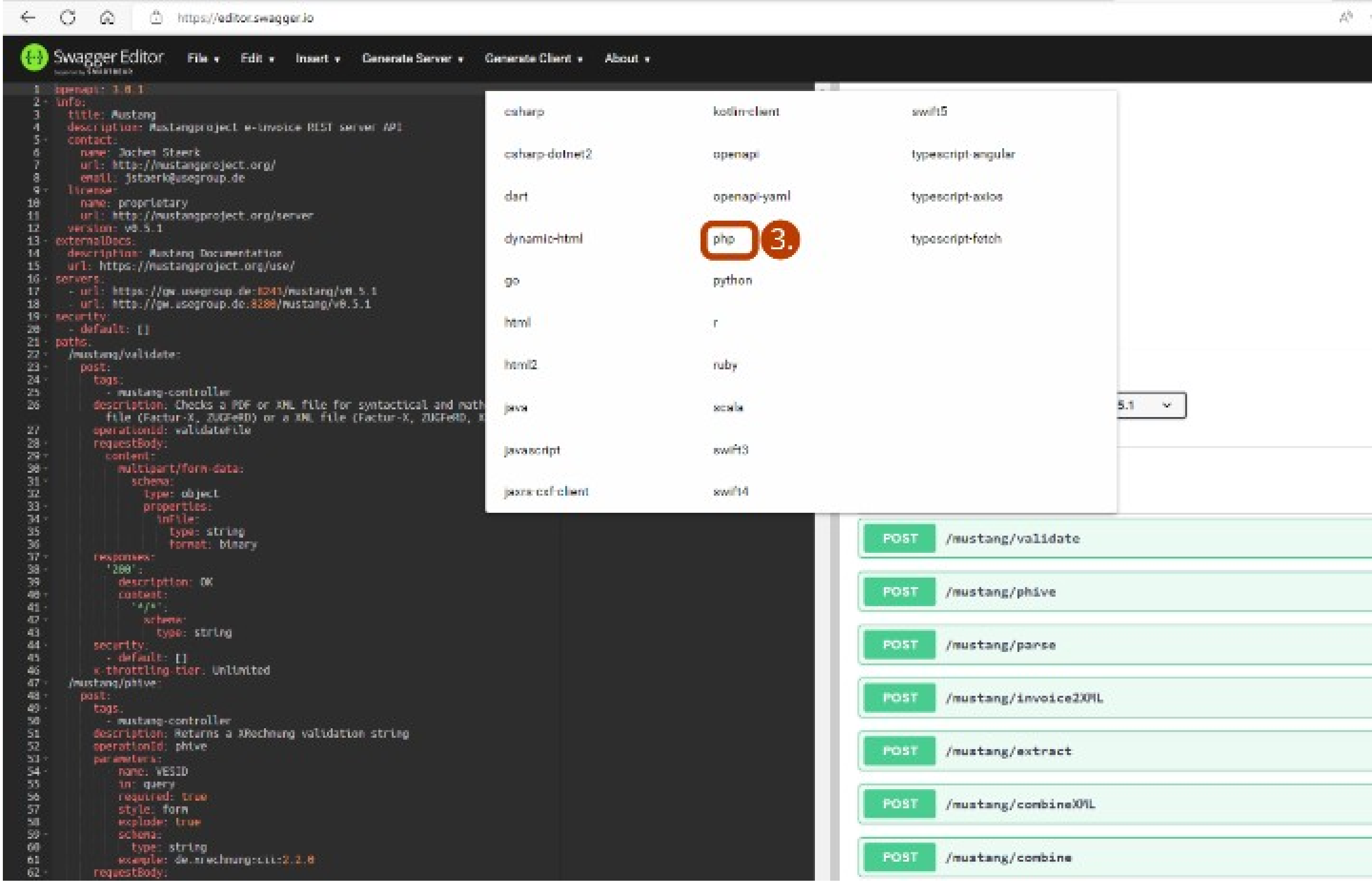
Feel free to also try the async functions.

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.)

Screenshot 2:

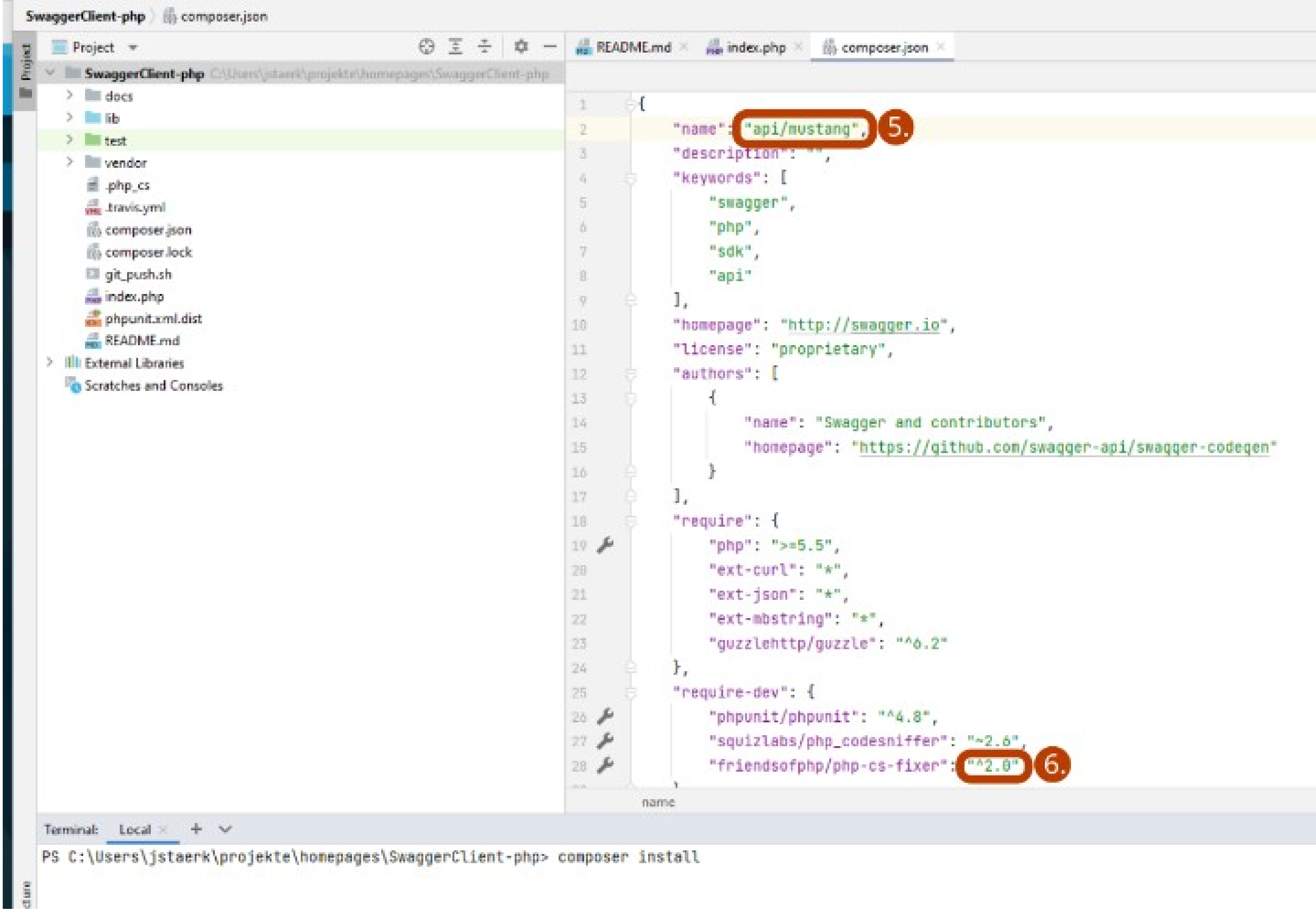


```
openapi: 3.0.1
info:
  title: Mustang
  description: Mustangproject e-invoice REST server API
  contact:
    name: Jochen Staerk
    url: http://mustangproject.org/
    email: jstaerk@usegroup.de
  license:
    name: proprietary
    url: http://mustangproject.org/server
  version: v0.5.1
externalDocs:
  description: Mustang Documentation
  url: https://mustangproject.org/use/
servers:
  - url: https://gw.usegroup.de:8243/mustang/v0.5.1
  - url: http://gw.usegroup.de:8280/mustang/v0.5.1
security:
  - default: []
paths:
  /mustang/validate:
    post:
      tags:
        - mustang-controller
      description: Checks a PDF or XML file for syntactical and math
        file (Factur-X, ZUGFeRD) or a XML file (Factur-X, ZUGFeRD, X
      operationId: validateFile
      requestBody:
        content:
          multipart/form-data:
            schema:
              type: object
              properties:
                inFile:
                  type: string
                  format: binary
      responses:
        '200':
          description: OK
          content:
            '*/*':
              schema:
                type: string
      security:
        - default: []
      x-throttling-tier: Unlimited
  /mustang/phive:
    post:
      tags:
        - mustang-controller
      description: Returns a XRechnung validation string
      operationId: phive
      parameters:
          name: VESID
          in: query
          required: true
          style: form
          explode: true
          schema:
            type: string
          example: de.xrechnung:cii:2.2.0
      requestBody:
```

#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:

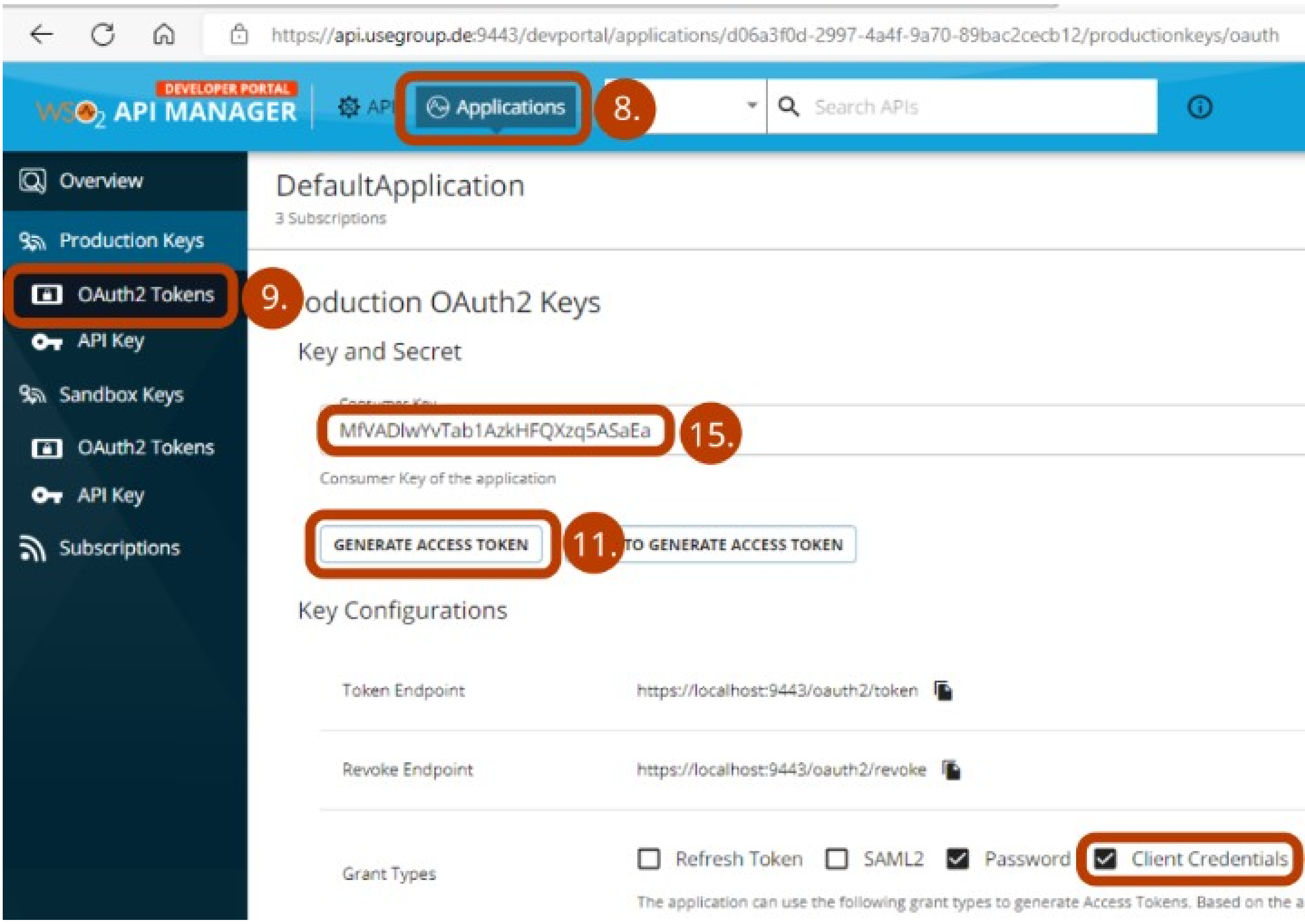


```
{
    "name": "api/mustang",
    "description": "",
    "keywords": [
        "swagger",
        "php",
        "sdk",
        "api"
    ],
    "homepage": "http://swagger.io",
    "license": "proprietary",
    "authors": [
        {
            "name": "Swagger and contributors",
            "homepage": "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"
```

```
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:



https://api.usegroup.de:9443/devportal/applications/d06a3f0d-2997-4a4f-9a70-89bac2cecb12/productionkeys/oauth

DefaultApplication
3 Subscriptions

oduction OAuth2 Keys

Key and Secret

Consumer Key: MfVADlwYvTab1AzkHFQXzq5ASaEa
Consumer Key of the application

Key Configurations

Token Endpoint https://localhost:9443/oauth2/token

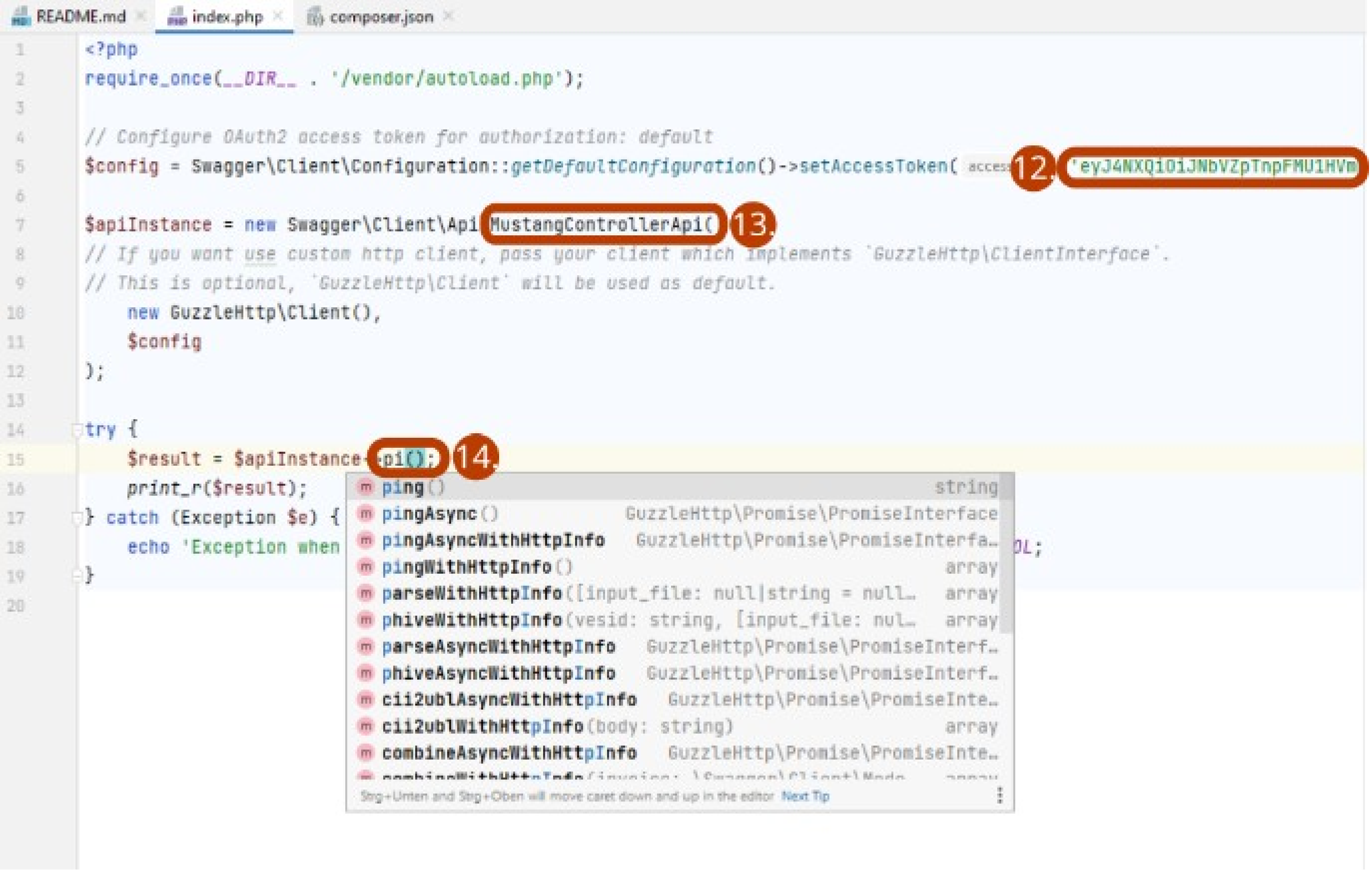
Revoke Endpoint https://localhost:9443/oauth2/revoke

Grant Types □ Refresh Token □ SAML2 ☑ Password ☑ Client Credentials
The application can use the following grant types to generate Access Tokens. Based on the a

#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:

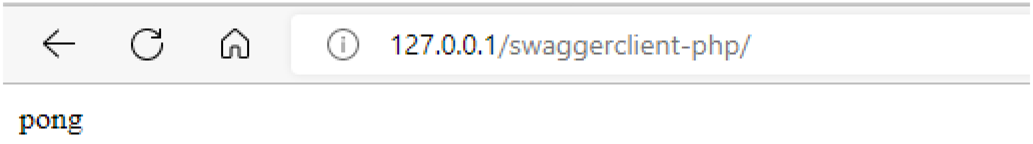


README.md
index.php
composer.json
12.
13.
14.
ping() string
pingAsync() GuzzleHttp\Promise\PromiseInterface
pingAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterfa…
pingWithHttpInfo() array
parseWithHttpInfo([input_file: null|string = null… array
phiveWithHttpInfo(vesid: string, [input_file: nul… array
parseAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterf…
phiveAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInterf…
cii2ublAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInte…
cii2ublWithHttpInfo(body: string) array
combineAsyncWithHttpInfo GuzzleHttp\Promise\PromiseInte…
Strg+Unten and Strg+Oben will move caret down and up in the editor Next Tip

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

// Configure OAuth2 access token for authorization: default
$config = Swagger\Client\Configuration::getDefaultConfiguration()->setAccessToken( access 'eyJ4NXQiOiJNbVZpTnpFMU1HVm

$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 = $apiInstance->pi();
    print_r($result);
} catch (Exception $e) {
    echo 'Exception when                                                                      OL;
}
```

#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 processort in your browser, it should now look like Screenshot 6:



127.0.0.1/swaggerclient-php/

pong

Validation of electronic invoices

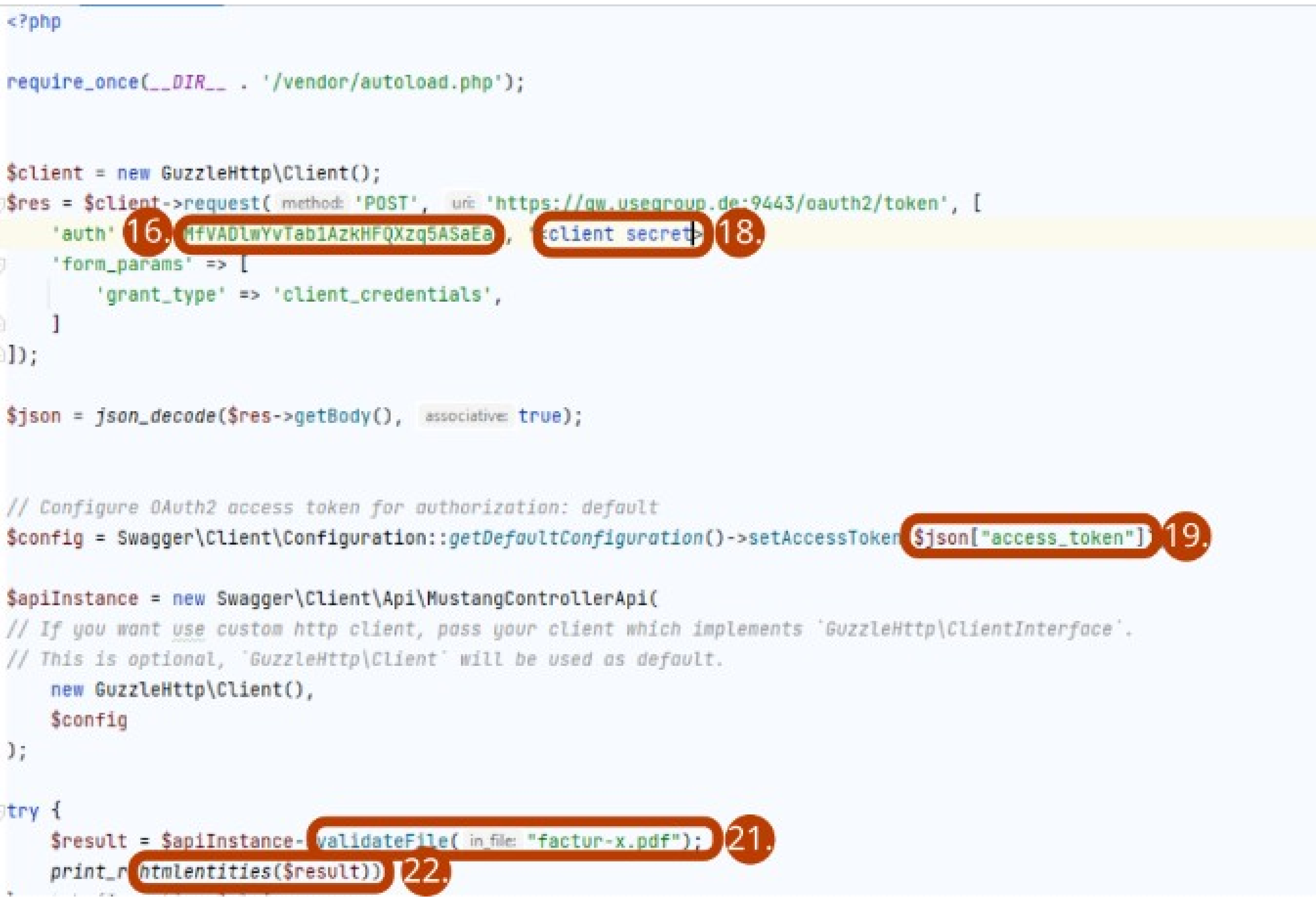
Concerning Screenshot 4:
In index.php paste the following code

$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( method: 'POST',  uri: 'https://gw.usegroup.de:9443/oauth2/token', [
    'auth'  MfVADlwYvTab1AzkHFQXzq5ASaEa,  <client secret>
    '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"]

$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 = $apiInstance- validateFile( in_file: "factur-x.pdf");
    print_r htmlentities($result))
```

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

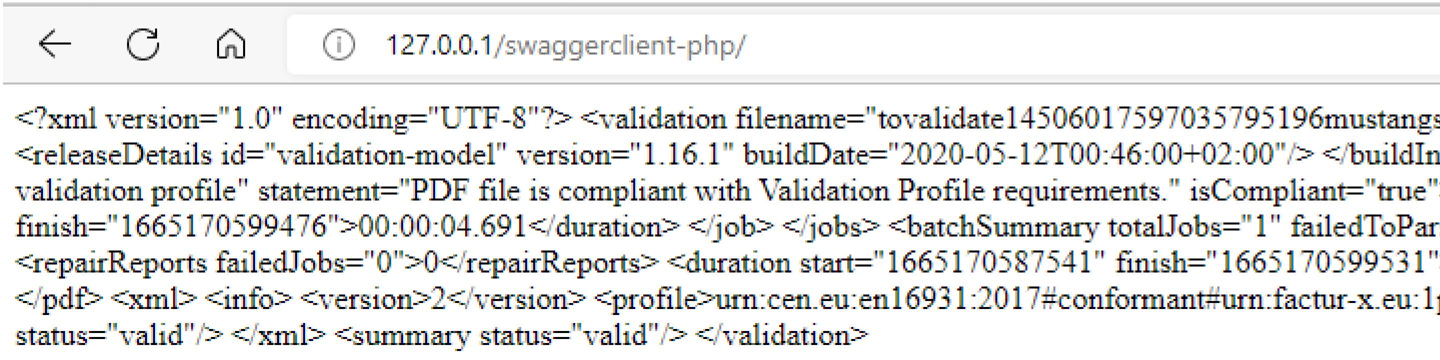
#17. reveal and copy Consumer Secret (18.).

#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 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="tovalidate14506017597035795196mustangs
<releaseDetails id="validation-model" version="1.16.1" buildDate="2020-05-12T00:46:00+02:00"/> </buildIn
validation profile" statement="PDF file is compliant with Validation Profile requirements." isCompliant="true"
finish="1665170599476">00:00:04.691</duration> </job> </jobs> <batchSummary totalJobs="1" failedToPar
<repairReports failedJobs="0">0</repairReports> <duration start="1665170587541" finish="1665170599531"
</pdf> <xml> <info> <version>2</version> <profile>urn:cen.eu:en16931:2017#conformant#urn:factur-x.eu:1
status="valid"/> </xml> <summary status="valid"/> </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 16

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.

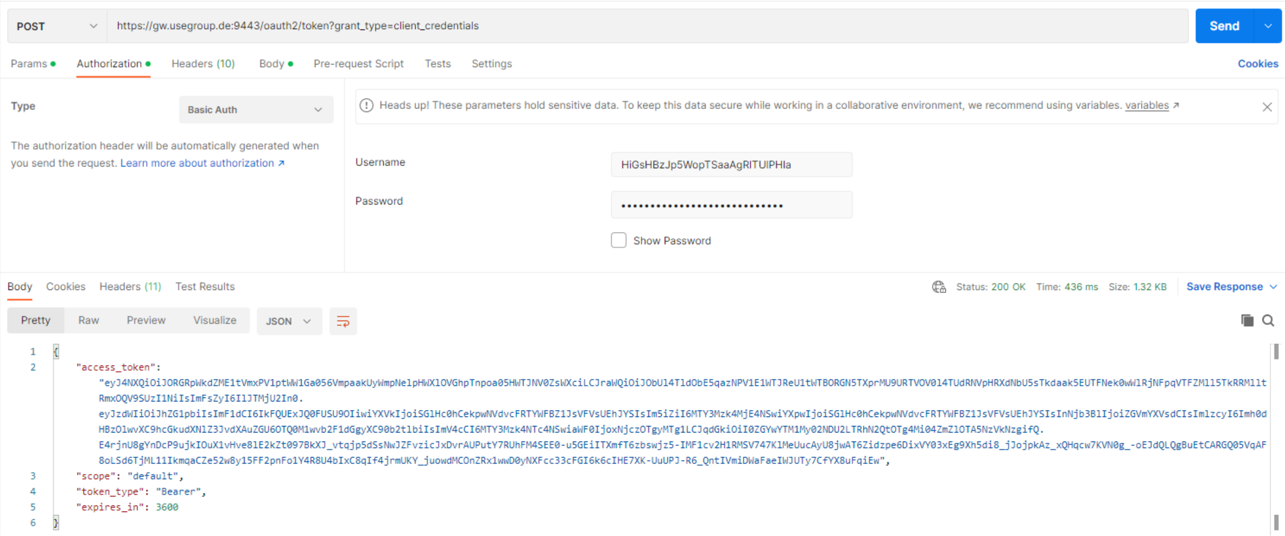
This example is based on Postman, you will need enabled client credentials as described on page 16 in Allowing Client Credentials.

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

Add a request to 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.usegroup.de:9443/oauth2/token?grant_type=client_credentials

Type: Basic Auth

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

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

Username: HiGsHBzJp5WopTSaaAgRITUIPHIa

Password: ••••••••••••••••••••••••••••

□ Show Password

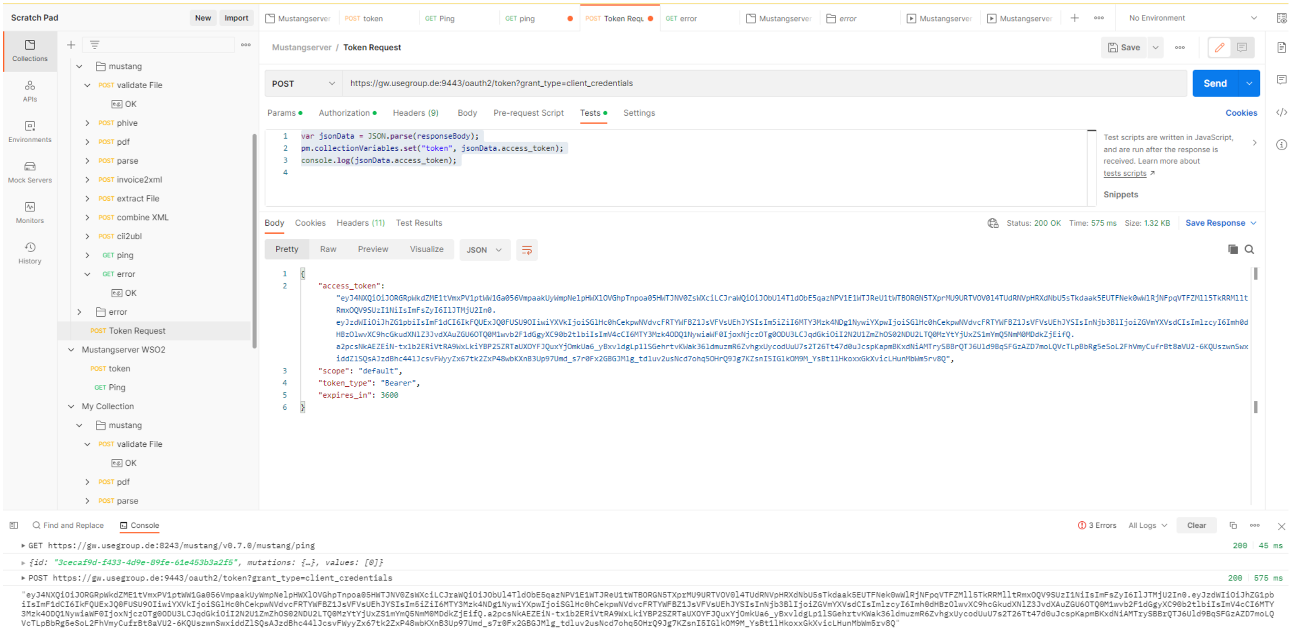
Status: 200 OK Time: 436 ms Size: 1.32 KB

```
{
    "access_token":
        "eyJ4NXQiOiJORGRpWkdZME1tVmxPV1ptWW1Ga056VmpaakUyWmpNelpHWXlOVGhpTnpoa05HWTJNV0ZsWXciLCJraWQiOiJObUl4TldObE5qazNPV1E1WTJReU1tWTBORGN5TXprMU9URTVOV0l4TUdRNVpHRXdNbU5sTkdaak5EUTFNek0wWlRjNFpqVTFZMll5TkRRMllt
        RmxOQV9SUzI1NiIsImFsZyI6IlJTMjU2In0.
        eyJzdWIiOiJhZG1pbiIsImF1dCI6IkFQUExJQ0FUSU9OIiwiYXVkIjoiSGlHc0hCekpwNVdvcFRTYWFBZ1JsVFVsUEhJYSIsIm5iZiI6MTY3Mzk4MjE4NSwiYXpwIjoiSGlHc0hCekpwNVdvcFRTYWFBZ1JsVFVsUEhJYSIsInNjb3BlIjoiZGVmYXVsdCIsImlzcyI6Imh0d
        HBzOlwvXC9hcGkudXNlZ3JvdXAuZGU6OTQ0M1wvb2F1dGgyXC90b2tlbiIsImV4cCI6MTY3Mzk4NTc4NSwiaWF0IjoxNjczOTgyMTg1LCJqdGkiOiI0ZGYwYTM1My02NDU2LTRhN2QtOTg4Mi04ZmZlOTA5NzVkNzgifQ.
        E4rjnU8gYnDcP9ujkIOuX1vHve8lE2kZt097BkXJ_vtqjp5dSsNwJZFvzicJxDvrAUPutY7RUhFM4SEE0-u5GEiITXmfT6zbswjz5-IMF1cv2H1RMSV747KlMeUucAyU8jwAT6Zidzpe6DixVY03xEg9Xh5di8_jJojpkAz_xQHqcw7KVN0g_-oEJdQLQgBuEtCARGQ05VqAF
        8oLSd6TjML11IkmqaCZe52w8y15FF2pnFo1Y4R8U4bIxC8qIf4jrmUKY_juowdMCOnZRx1wwD0yNXFcc33cFGI6k6cIHE7XK-UuUPJ-R6_QntIVmiDWaFaeIWJUTy7CfYX8uFqiEw",
    "scope": "default",
    "token_type": "Bearer",
    "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:



POST https://gw.usegroup.de:9443/oauth2/token?grant_type=client_credentials

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

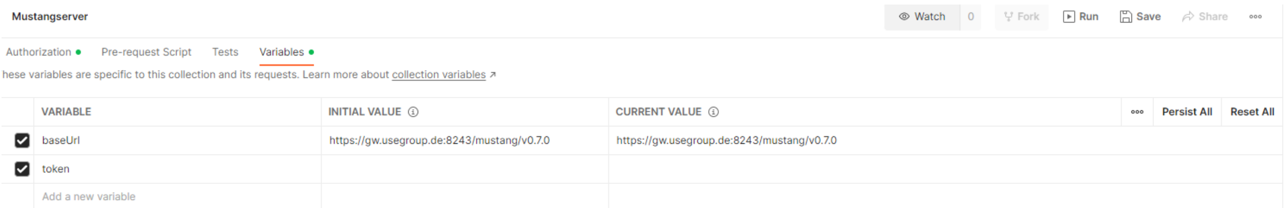
Test scripts are written in JavaScript, and are run after the response is received. Learn more about tests scripts ↗

Status: 200 OK Time: 575 ms Size: 1.32 KB

```
{
    "access_token":
        "eyJ4NXQiOiJORGRpWkdZME1tVmxPV1ptWW1Ga056VmpaakUyWmpNelpHWXlOVGhpTnpoa05HWTJNV0ZsWXciLCJraWQiOiJObUl4TldObE5qazNPV1E1WTJReU1tWTBORGN5TXprMU9URTVOV0l4TUdRNVpHRXdNbU5sTkdaak5EUTFNek0wWlRjNFpqVTFZMll5TkRRMllt
        RmxOQV9SUzI1NiIsImFsZyI6IlJTMjU2In0.
        eyJzdWIiOiJhZG1pbiIsImF1dCI6IkFQUExJQ0FUSU9OIiwiYXVkIjoiSGlHc0hCekpwNVdvcFRTYWFBZ1JsVFVsUEhJYSIsIm5iZiI6MTY3Mzk4NDg1NywiYXpwIjoiSGlHc0hCekpwNVdvcFRTYWFBZ1JsVFVsUEhJYSIsInNjb3BlIjoiZGVmYXVsdCIsImlzcyI6Imh0d
        HBzOlwvXC9hcGkudXNlZ3JvdXAuZGU6OTQ0M1wvb2F1dGgyXC90b2tlbiIsImV4cCI6MTY3Mzk4ODQ1NywiaWF0IjoxNjczOTg0ODU3LCJqdGkiOiI2N2U1ZmZhOS02NDU2LTQ0MzYtYjUxZS1mYmQ5NmM0MDdkZjEifQ.
        a2pcsNkAEZEiN-tx1b2ERiVtRA9WxLkiYBP2SZRTaUXOYFJQuxYjOmkUa6_yBxvldgLp1lSGehrtvKWak36ldmuzmR6ZvhgxUycodUuU7s2T26Tt47d0uJcspKapmBKxdNiAMTrySBBrQTJ6Uld9BqSFGzAZD7moLQVcTLpBbRg5eSoL2FhVmyCufrBt8aVU2-6KQUszwnSwx
        iddZlSQsAJzdBhc44lJcsvFWyyZx67tk2ZxP48wbKXnB3Up97Umd_s7r0Fx2GBGJMlg_tdluv2usNcd7ohq5OHrQ9Jg7KZsnI5IGlkOM9M_YsBt1lHkoxxGkXvicLHunMbWm5rv8Q",
    "scope": "default",
    "token_type": "Bearer",
    "expires_in": 3600
}
```

```
▸ GET https://gw.usegroup.de:8243/mustang/v0.7.0/mustang/ping                                200 | 45 ms
▸ {id: "3cecaf9d-f433-4d9e-89fe-61e453b3a2f5", mutations: {…}, values: [0]}
▸ POST https://gw.usegroup.de:9443/oauth2/token?grant_type=client_credentials               200 | 575 ms
"eyJ4NXQiOiJORGRpWkdZME1tVmxPV1ptWW1Ga056VmpaakUyWmpNelpHWXlOVGhpTnpoa05HWTJNV0ZsWXciLCJraWQiOiJObUl4TldObE5qazNPV1E1WTJReU1tWTBORGN5TXprMU9URTVOV0l4TUdRNVpHRXdNbU5sTkdaak5EUTFNek0wWlRjNFpqVTFZMll5TkRRMlltRmxOQV9SUzI1NiIsImFsZyI6IlJTMjU2In0.eyJzdWIiOiJhZG1pb
iIsImF1dCI6IkFQUExJQ0FUSU9OIiwiYXVkIjoiSGlHc0hCekpwNVdvcFRTYWFBZ1JsVFVsUEhJYSIsIm5iZiI6MTY3Mzk4NDg1NywiYXpwIjoiSGlHc0hCekpwNVdvcFRTYWFBZ1JsVFVsUEhJYSIsInNjb3BlIjoiZGVmYXVsdCIsImlzcyI6Imh0dHBzOlwvXC9hcGkudXNlZ3JvdXAuZGU6OTQ0M1wvb2F1dGgyXC90b2tlbiIsImV4cCI6MTY
3Mzk4ODQ1NywiaWF0IjoxNjczOTg0ODU3LCJqdGkiOiI2N2U1ZmZhOS02NDU2LTQ0MzYtYjUxZS1mYmQ5NmM0MDdkZjEifQ.a2pcsNkAEZEiN-tx1b2ERiVtRA9WxLkiYBP2SZRTaUXOYFJQuxYjOmkUa6_yBxvldgLp1lSGehrtvKWak36ldmuzmR6ZvhgxUycodUuU7s2T26Tt47d0uJcspKapmBKxdNiAMTrySBBrQTJ6Uld9BqSFGzAZD7moLQ
VcTLpBbRg5eSoL2FhVmyCufrBt8aVU2-6KQUszwnSwxiddZlSQsAJzdBhc44lJcsvFWyyZx67tk2ZxP48wbKXnB3Up97Umd_s7r0Fx2GBGJMlg_tdluv2usNcd7ohq5OHrQ9Jg7KZsnI5IGlkOM9M_YsBt1lHkoxxGkXvicLHunMbWm5rv8Q"
```

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

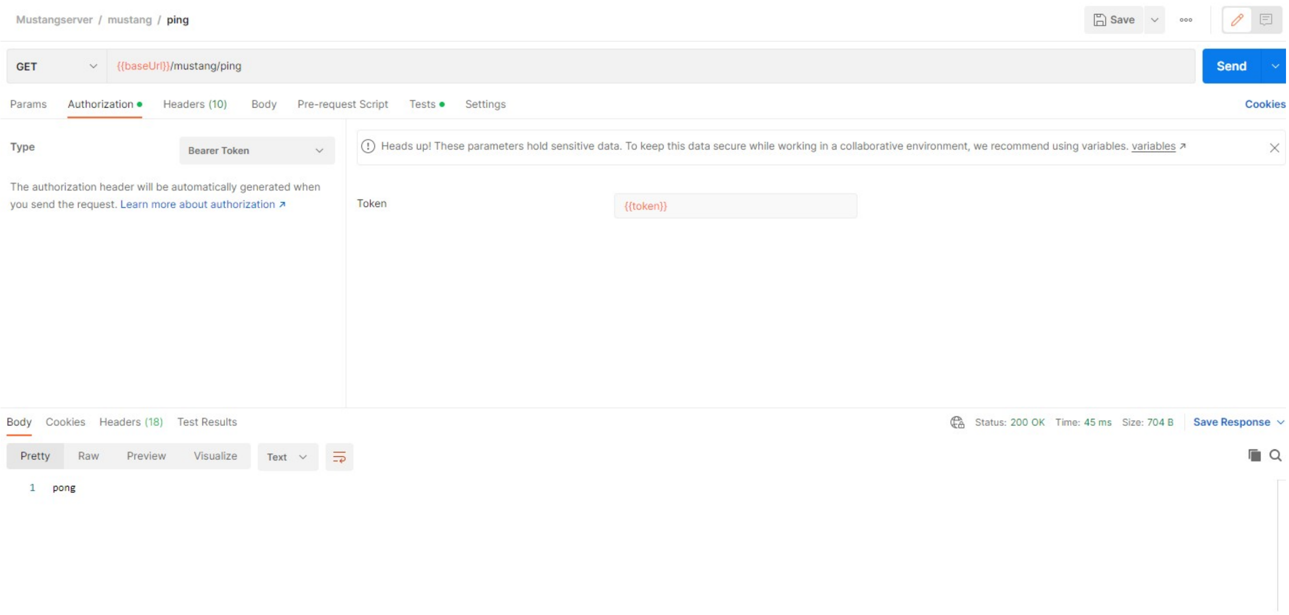


Mustangserver

hese variables are specific to this collection and its requests. Learn more about collection variables ↗

	VARIABLE	INITIAL VALUE	CURRENT VALUE
☑	baseUrl	https://gw.usegroup.de:8243/mustang/v0.7.0	https://gw.usegroup.de:8243/mustang/v0.7.0
☑	token		
	Add a new 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

GET {{baseUrl}}/mustang/ping

Type: Bearer Token

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

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

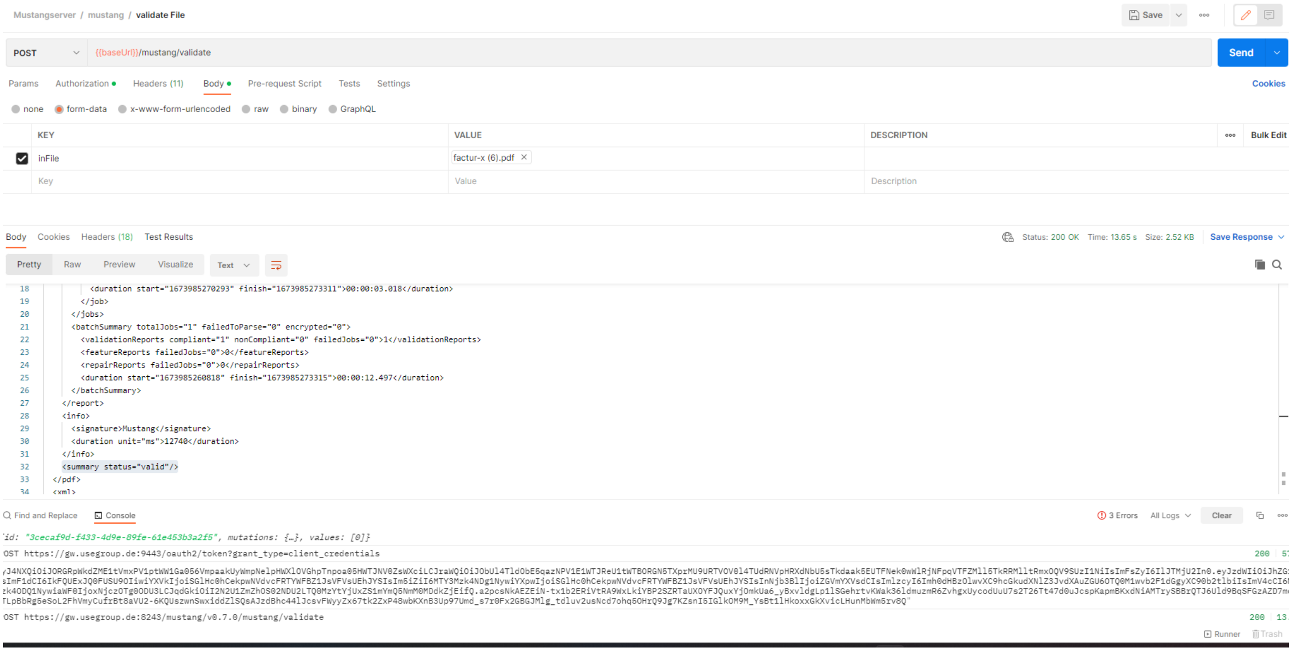
Token: {{token}}

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

```
pong
```

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 factur-x:



Mustangserver / mustang / validate File

POST {{baseUrl}}/mustang/validate

○ none ● form-data ○ x-www-form-urlencoded ○ raw ○ binary ○ GraphQL

	KEY	VALUE	DESCRIPTION
☑	inFile	factur-x (6).pdf ×	
	Key	Value	Description

Status: 200 OK Time: 13.65 s Size: 2.52 KB

```
        <duration start="1673985270293" finish="1673985273311">00:00:03.018</duration>
      </job>
    </jobs>
    <batchSummary totalJobs="1" failedToParse="0" encrypted="0">
      <validationReports compliant="1" nonCompliant="0" failedJobs="0">1</validationReports>
      <featureReports failedJobs="0">0</featureReports>
      <repairReports failedJobs="0">0</repairReports>
      <duration start="1673985260818" finish="1673985273315">00:00:12.497</duration>
    </batchSummary>
  </report>
  <info>
    <signature>Mustang</signature>
    <duration unit="ms">12740</duration>
  </info>
  <summary status="valid"/>
</pdf>
<xml>
```

```
id: "3cecaf9d-f433-4d9e-89fe-61e453b3a2f5", mutations: {…}, values: [0]}
OST https://gw.usegroup.de:9443/oauth2/token?grant_type=client_credentials                    200 | 5
yJ4NXQiOiJORGRpWkdZME1tVmxPV1ptWW1Ga056VmpaakUyWmpNelpHWXlOVGhpTnpoa05HWTJNV0ZsWXciLCJraWQiOiJObUl4TldObE5qazNPV1E1WTJReU1tWTBORGN5TXprMU9URTVOV0l4TUdRNVpHRXdNbU5sTkdaak5EUTFNek0wWlRjNFpqVTFZMll5TkRRMlltRmxOQV9SUzI1NiIsImFsZyI6IlJTMjU2In0.eyJzdWIiOiJhZG
sImF1dCI6IkFQUExJQ0FUSU9OIiwiYXVkIjoiSGlHc0hCekpwNVdvcFRTYWFBZ1JsVFVsUEhJYSIsIm5iZiI6MTY3Mzk4NDg1NywiYXpwIjoiSGlHc0hCekpwNVdvcFRTYWFBZ1JsVFVsUEhJYSIsInNjb3BlIjoiZGVmYXVsdCIsImlzcyI6Imh0dHBzOlwvXC9hcGkudXNlZ3JvdXAuZGU6OTQ0M1wvb2F1dGgyXC90b2tlbiIsImV4cCI6
zk4ODQ1NywiaWF0IjoxNjczOTg0ODU3LCJqdGkiOiI2N2U1ZmZhOS02NDU2LTQ0MzYtYjUxZS1mYmQ5NmM0MDdkZjEifQ.a2pcsNkAEZEiN-tx1b2ERiVtRA9WxLkiYBP2SZRTaUXOYFJQuxYjOmkUa6_yBxvldgLp1lSGehrtvKWak36ldmuzmR6ZvhgxUycodUuU7s2T26Tt47d0uJcspKapmBKxdNiAMTrySBBrQTJ6Uld9BqSFGzAZD7m
TLpBbRg5eSoL2FhVmyCufrBt8aVU2-6KQUszwnSwxiddZlSQsAJzdBhc44lJcsvFWyyZx67tk2ZxP48wbKXnB3Up97Umd_s7r0Fx2GBGJMlg_tdluv2usNcd7ohq5OHrQ9Jg7KZsnI5IGlkOM9M_YsBt1lHkoxxGkXvicLHunMbWm5rv8Q"
OST https://gw.usegroup.de:8243/mustang/v0.7.0/mustang/validate                               200 | 13
```

Performance Tests with Jmeter

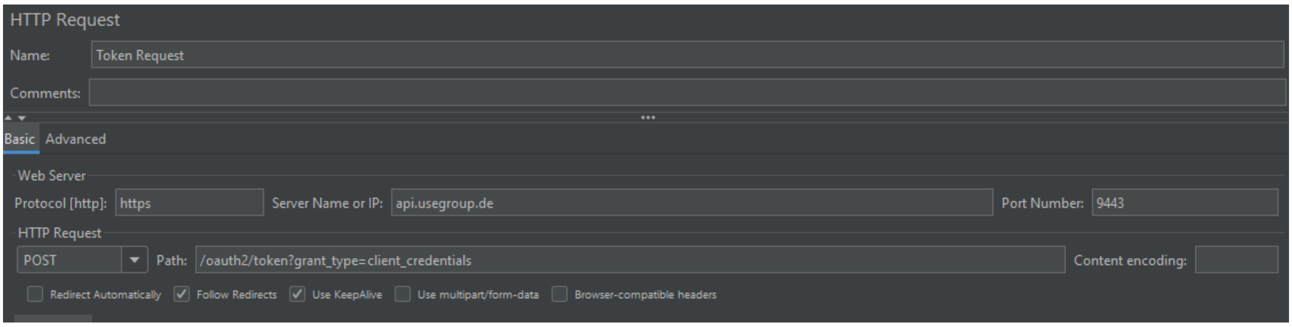
Jmeter is a generic load testing tool and load generator.

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 16 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:



HTTP Request

Name: Token Request

Comments:

Web Server

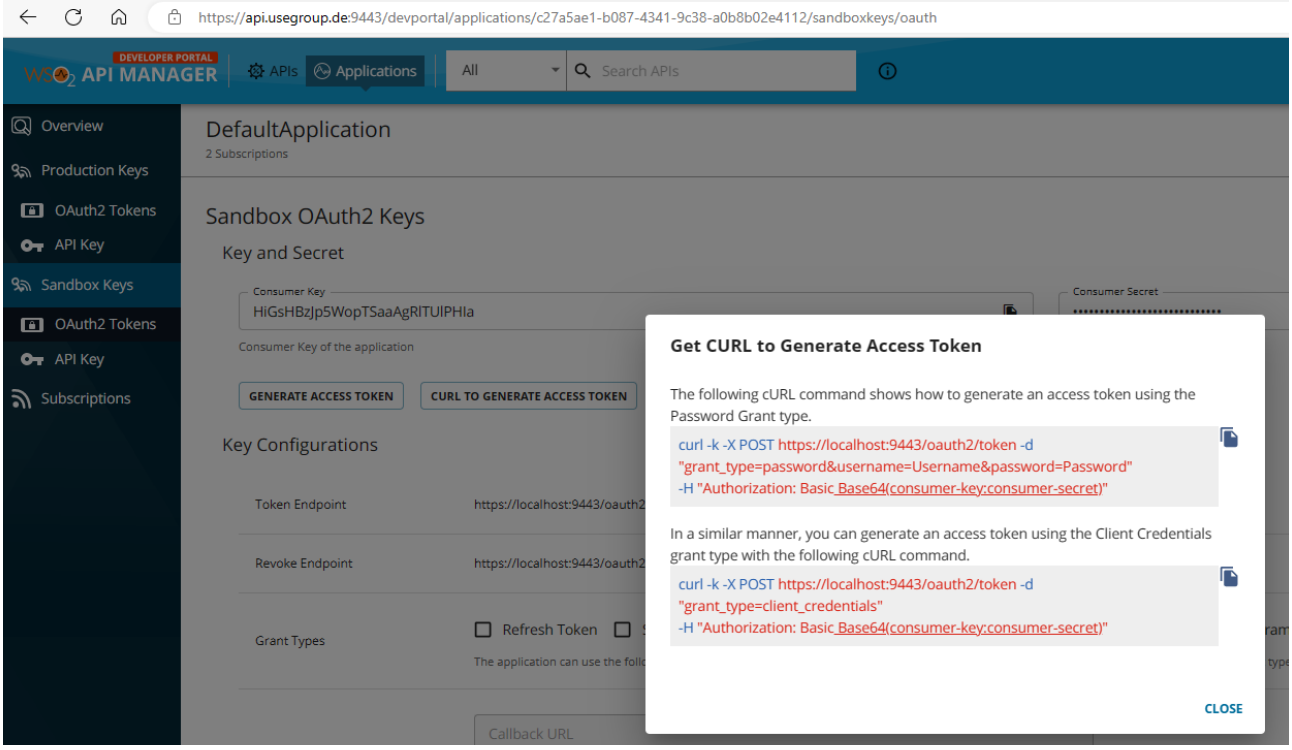
Protocol [http]: https Server Name or IP: api.usegroup.de Port Number: 9443

HTTP Request

POST Path: /oauth2/token?grant_type=client_credentials Content encoding:

□ Redirect Automatically ☑ Follow Redirects ☑ Use KeepAlive □ Use multipart/form-data □ Browser-compatible headers

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



Consumer Key: HiGsHBzJp5WopTSaaAgRITUlPHla

Consumer Key of the application

Consumer Secret: ••••••••••••••••••••••••••••

Token Endpoint: https://localhost:9443/oauth2

Revoke Endpoint: https://localhost:9443/oauth2

Grant Types: □ Refresh Token □

The application can use the follo

Get CURL to Generate Access Token

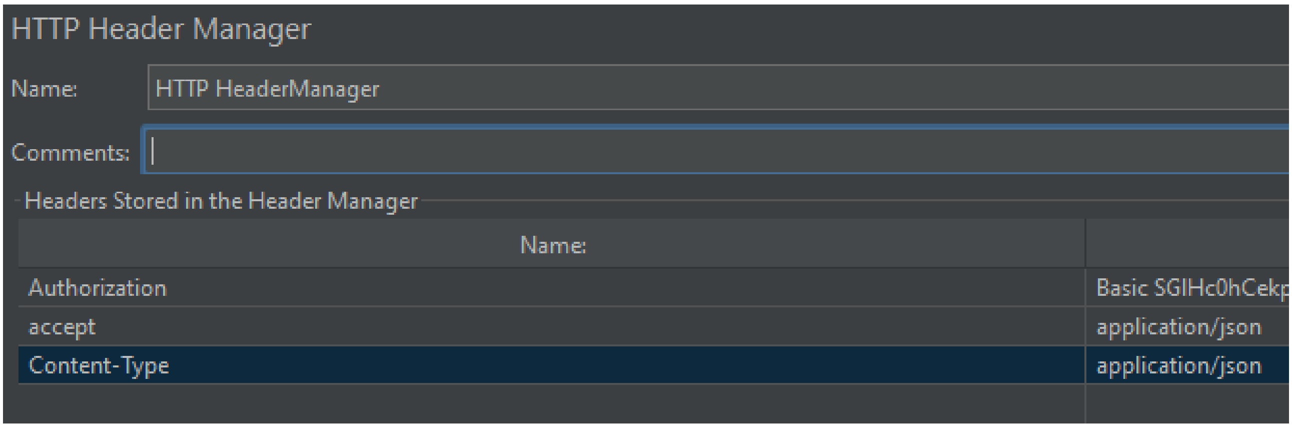
The following cURL command shows how to generate an access token using the Password Grant type.

```
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)"
```

In a similar manner, you can generate an access token using the Client Credentials grant type with the following cURL command.

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

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



HTTP Header Manager

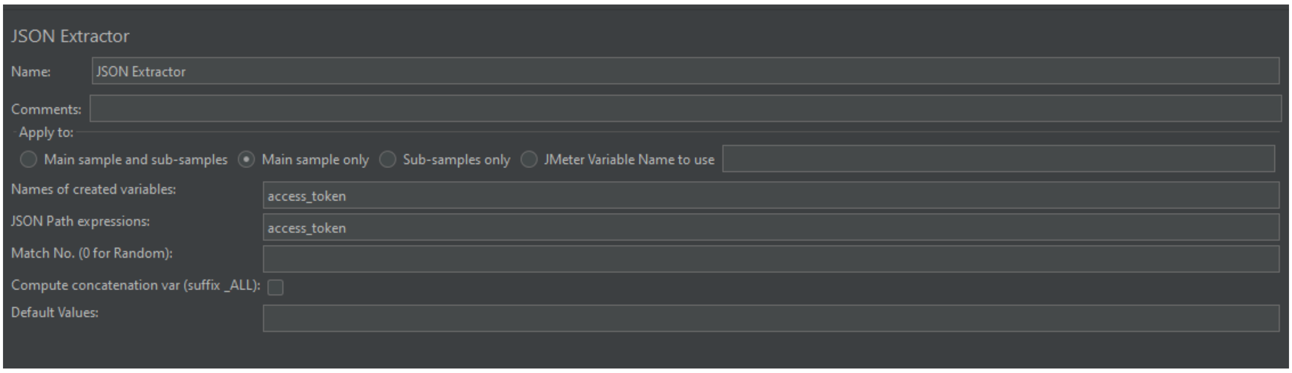
Name: HTTP HeaderManager

Comments:

Headers Stored in the Header Manager

Name:	
Authorization	Basic SGIHc0hCekp
accept	application/json
Content-Type	application/json

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



JSON Extractor

Name: JSON Extractor

Comments:

Apply to:

○ Main sample and sub-samples ● Main sample only ○ Sub-samples only ○ JMeter Variable Name to use

Names of created variables: access_token

JSON Path expressions: access_token

Match No. (0 for Random):

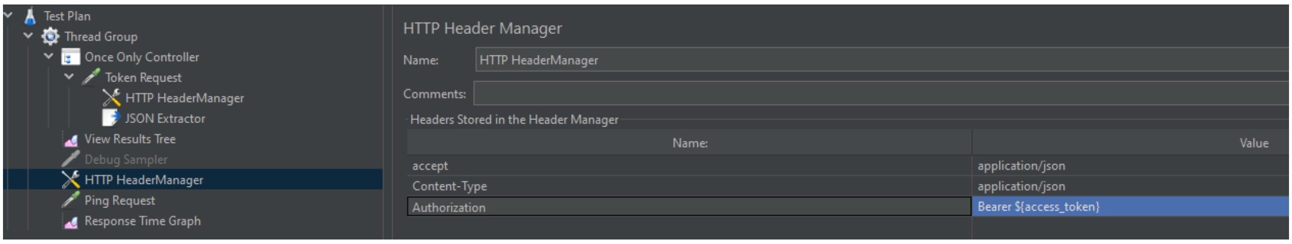
Compute concatenation var (suffix _ALL): ☐

Default Values:

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}



```
Test Plan
  Thread Group
    Once Only Controller
      Token Request
        HTTP HeaderManager
        JSON Extractor
    View Results Tree
    Debug Sampler
    HTTP HeaderManager
    Ping Request
    Response Time Graph
```

HTTP Header Manager

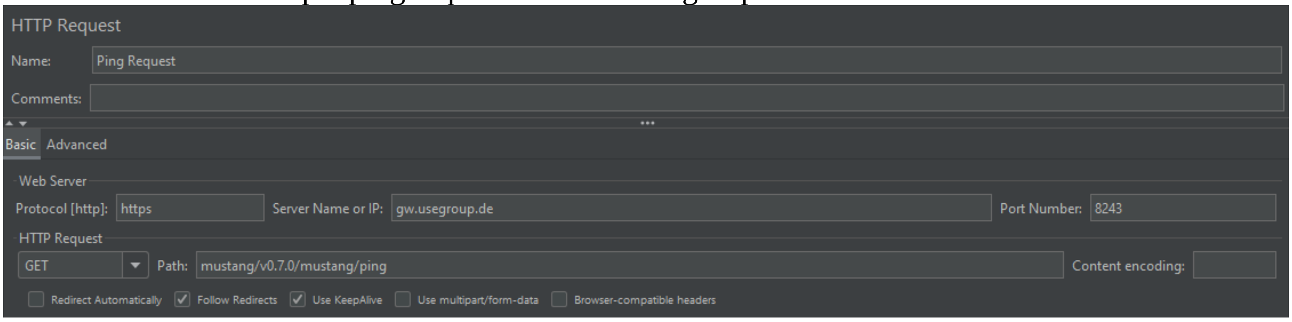
Name: HTTP HeaderManager

Comments:

Headers Stored in the Header Manager

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



HTTP Request

Name: Ping Request

Comments:

Basic Advanced

Web Server

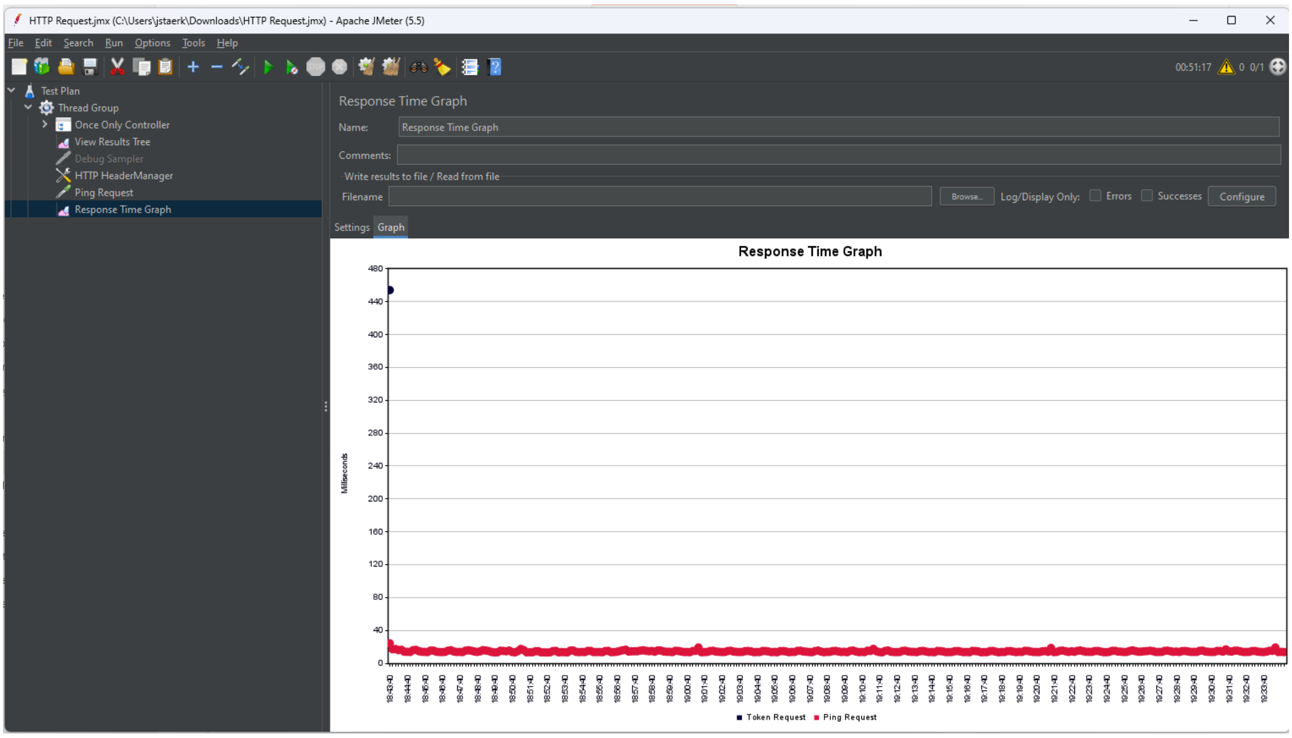
Protocol [http]: https Server Name or IP: gw.usegroup.de Port Number: 8243

HTTP Request

GET Path: mustang/v0.7.0/mustang/ping Content encoding:

☐ Redirect Automatically ☑ Follow Redirects ☑ Use KeepAlive ☐ Use multipart/form-data ☐ Browser-compatible headers

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.



Response Time Graph

Name: Response Time Graph

Comments:

Write results to file / Read from file

Filename

Log/Display Only: □ Errors □ Successes

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) 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

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

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.