

API

Workshop 05.08.2025

e-on

- Ziel: technisch tiefergreifendes Verständnis für HTTP API's mit JSON
- Methoden der Anfrage
- Aufbau von Anfragen (Requests)
- Analyse von Request / Responses im Browser
- Entwicklung eines kleinen PHP-Servers
- Abfragen an den PHP-Server mit eigenem Client
- Aufbau JSON in Verbindung mit OpenAPI
- Umgang mit Langläufern

The logo for 'e-on' is displayed in a bold, red, stylized font. The 'e' is lowercase and has a thick, rounded shape. The dot is a solid red circle. The 'on' is also lowercase, with the 'o' being a thick, rounded shape and the 'n' having a thick, rounded shape with a slight curve at the end.

Voraussetzungen

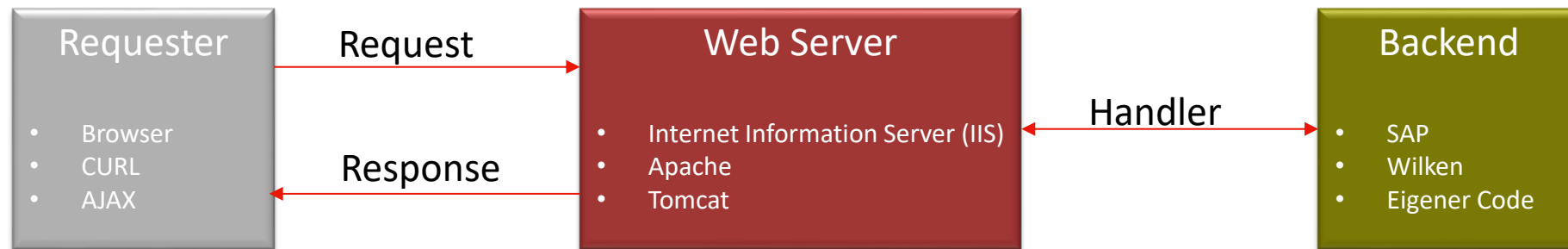
Ziel des Workshops ist es, dass jeder Teilnehmer in der Lage ist, selbständig einen eigenen WEB-Dienst mittels PHP zu entwickeln, um API's en Detail zu verstehen

- Der Web-Dienst soll zur Demonstration dienen
- Es ist kein Programmierkurs – sondern soll bestimmte Voraussetzungen und Zusammenhänge für API's verdeutlichen
- Laden Sie bitte von <https://makohub.de> im Menü Doku die PHP Installation herunter und entpacken Sie diese in ein von Ihnen gewähltes Verzeichnis. (Login: VornameNachname, PW: makohub, z. B. EdgarJung)
- Testen Sie bitte die Lauffähigkeit des PHP Interpreters durch Eingabe (Windows Eingabeaufforderung): php -version (Sie sollten keinen Fehler sondern eine Versionsinformation erhalten)

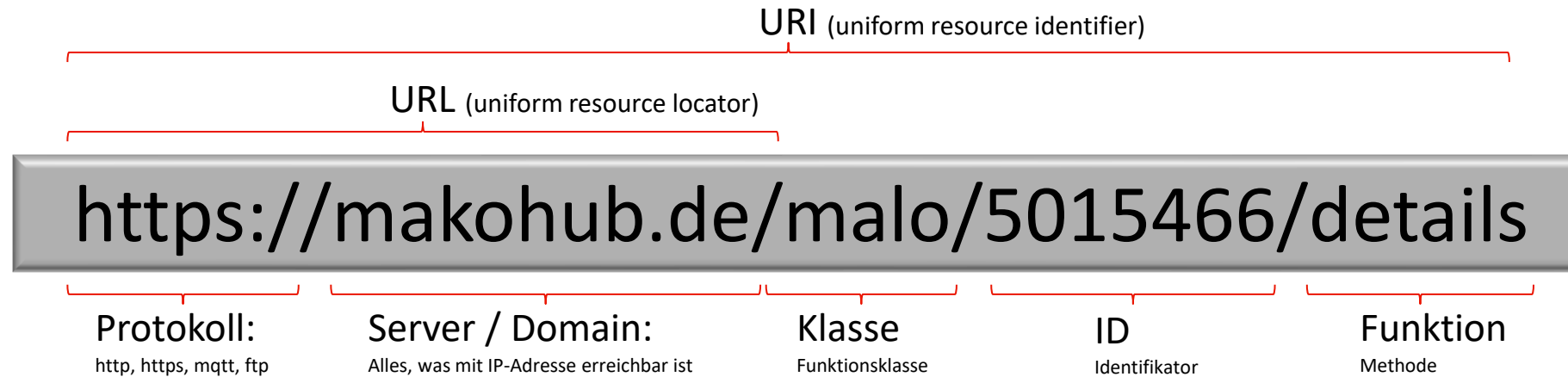
```
C:\Users\EdgarJung>php -version
PHP 8.3.19 (cli) (built: Mar 12 2025 14:03:36) (ZTS Visual C++ 2019 x64)
Copyright (c) The PHP Group
Zend Engine v4.3.19, Copyright (c) Zend Technologies
    with Zend OPcache v8.3.19, Copyright (c), by Zend Technologies
    with Xdebug v3.4.2, Copyright (c) 2002-2025, by Derick Rethans
```

- Installieren Sie den Visual Studio Code Editor ([Download Visual Studio Code - Mac, Linux, Windows](#)). Sollten Sie keine Berechtigung für die Installation von Software haben, können Sie alternativ den vorhandenen notepad Editor oder die editor APP von Windows verwenden.

Grundsätzlicher Aufbau von Web API Anwendungen. Strikte Trennung von WebServer (Transportschicht) und Backend (Verarbeitungsschicht)



Aufbau von Requests – hier der URI



Beispiel	Bewertung	Info
https://makohub.de/malo/v1/ident	DON'T	Moderne Schnittstellen verwenden auf keinen Fall die Version in der URI. Die Version wird im HTTP Header mitgeführt. (Gute Schnittstellen ändern keine Links – siehe TU Dortmund URL-Segment - TYPO3-Doku - TU Dortmund)
https://makohub.de/rest.php?malo=5015466	DON'T	Keine Abstraktion von der dahinterliegenden Implementierung. Es wird in der SSt PHP erwartet. Sie API kann nicht einfach ausgetauscht werden. Die Verwendung von ? Ist technisch möglich, jedoch veraltet
https://makohub.de/malo/list	OK	Hier möchte man eine Liste von Ergebnissen erhalten.
https://makohub.de/malo/5015466	OK	OK, wenn die Requestmethode sauber ausgeführt wurde (PUT, GET, POST, DELETE)

Request Methoden

Methode	Wirkung	Info
GET	Daten lesen	Nur Daten lesen, keine Verarbeitung. Browser Standard. Nur Request URI, Body technisch möglich, könnte jedoch abgelehnt werden. Daher keine Body Daten für GET verwenden
POST	Neue Daten anlegen	Nahezu keine Größenbeschränkung. Grundsätzlich Standardmethode in API's
PUT	UPDATE komplett	Update von kompletten Datensätzen bzw. UPLOAD von Dateien. Achtung gesonderte Freischaltung am Webserver
PATCH	UPDATE Teilweise	Anders wie bei PUT, können hier Teile von Daten ersetzt werden ohne einen kompletten Datenversand erneut zu initiieren. Achtung gesonderte Freischaltung am Webserver
DELETE	LÖSCHEN	Löschen von Daten. Muss für REST gesondert freigeschaltet werden.
CONNECT	Verbindungsaufbau	Login und Initiierung einer Session. Gesonderte Freischaltung am Server

Analyse eines Request (IE, Taste F12)

<https://makohub.de/services/alive>

URI

ANALYSE

RESPONSE

Laufzeit

Methode

Response Header

Request Header

Request Methode

Request Methode

Response Body

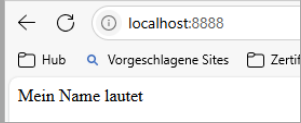
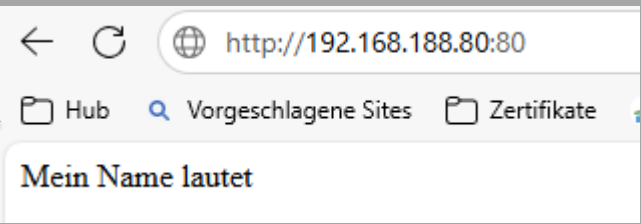
The screenshot shows the Chrome DevTools Network tab. The selected request is a GET request to `https://makohub.de/services/alive`. The status is 200 OK. The response headers and body are visible. The response body is a JSON object: `{ "IP": "194.164.56.107", "host": "util-exchange", "webserver": "Microsoft-IIS/10.0", "requestmethod": "GET", "serverinstance": "HUB", "alive": true }`.

Header	Value
Access-Control-Allow-Credentials	true
Access-Control-Allow-Headers	X-Requested-With, Content-Type, Origin, Cache-Control, Pragma, Authorization, Accept, Accept-Encoding, Session
Access-Control-Allow-Methods	PUT, POST, GET, OPTIONS, DELETE, HEAD
Access-Control-Allow-Origin	*
Cache-Control	no-store, no-cache, must-revalidate
Content-Length	144
Content-Type	application/json; charset=UTF-8
Date	Wed, 09 Jul 2025 14:39:50 GMT
Expires	Thu, 10 Jul 2025 14:39:50 GMT
Hub-Documentation	https://app.swaggerhub.com/apis/util-exchange
Hub-Id	1
Hub-Marketpartner	9907399000009
Hub-Openapi	3.0
Hub-Requestid	1311239
Hub-Serverversion	0.1
Pragma	no-cache
Server	Microsoft-IIS/10.0
User-Agent	Energy Data Hub 0.1
X-Powered-By	PHP/8.3.7
X-Powered-By	ASP.NET

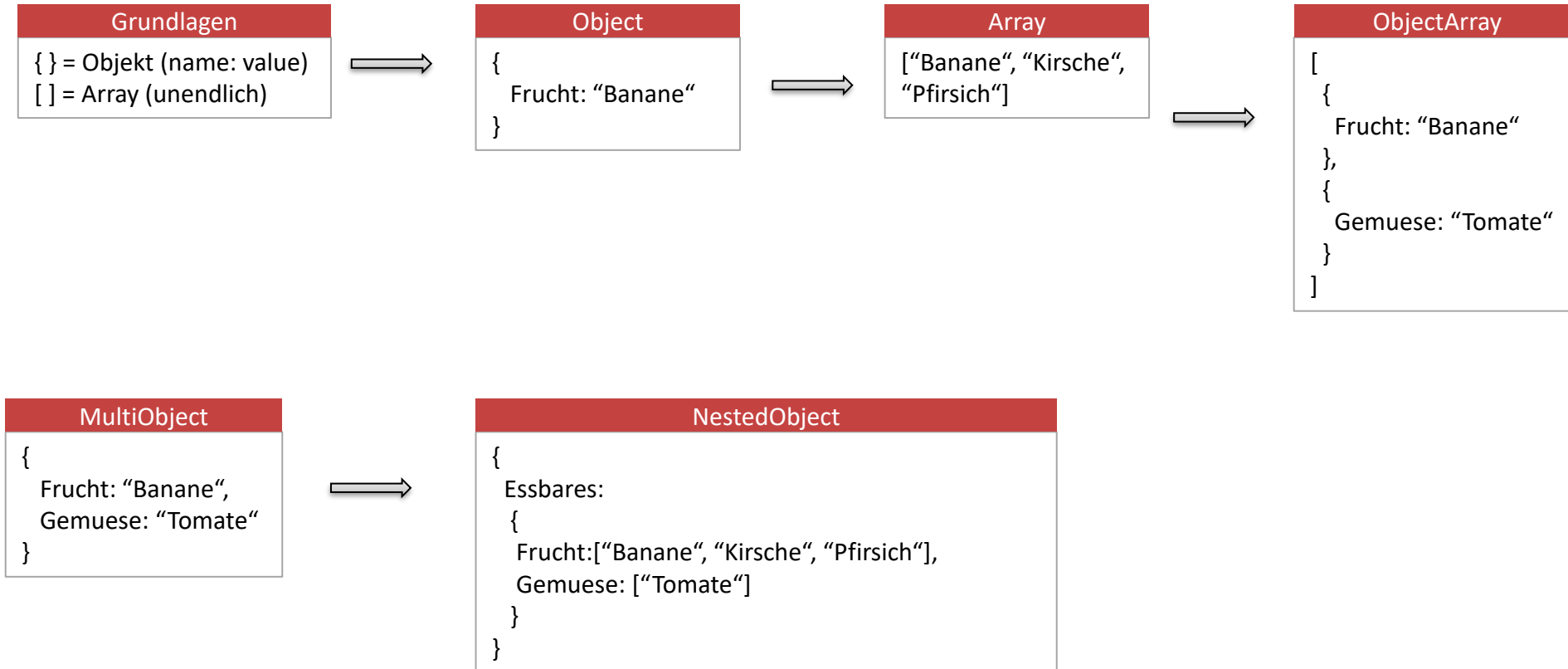
Fehlercodes / http Status Codes

CODE	Info
1X	Anfrage läuft noch – wird kaum verwendet
2X	Anfrage war erfolgreich
3X	Anfrage wurde weitergeleitet – hier müssen wir etwas unternehmen, um anschließend einen 2X zu erhalten. Kaum verwendet
<u>4X</u>	Der Client hat einen Fehler gemacht (Autorisierung fehlt, Syntax beim Request falsch, oder bei sofortiger Semantikprüfung)
5X	Der Server hat einen Fehler ausgelöst (Meist Programmier- oder Ressourcenfehler [Timeout, Speicher])

Entwicklung des ersten eigenen API-Servers mit PHP

Task	Info
1	Schreiben Sie Ihren Code und speichern Sie diesen in das PHP Verzeichnis in die Datei demoServer.php <pre>1 <?php 2 echo "Mein Name lautet"; 3 ?></pre>
2	Starten Sie den Server mit php -S localhost:80 demoServer.php <pre>C:\develop\Hub>php -S localhost:8888 demoServer.php [Wed Jul 9 17:20:59 2025] PHP 8.3.19 Development Server (http://localhost:8888) started [Wed Jul 9 17:21:03 2025] [::1]:55504 Accepted [Wed Jul 9 17:21:03 2025] [::1]:55505 Accepted [Wed Jul 9 17:21:03 2025] [::1]:55504 Closing [Wed Jul 9 17:21:03 2025] [::1]:55505 Closing [Wed Jul 9 17:21:04 2025] [::1]:55508 Accepted [Wed Jul 9 17:21:04 2025] [::1]:55508 Closing</pre>
3	Rufen Sie im Browser auf: http://localhost:80 
4	Lassen Sie sich die IP-Adresse von Ihrem Nachbarn geben (geben Sie ipconfig auf in der Eingabeaufforderung ein). Versuchen Sie den Server Ihres Kollegen zu erreichen: http://192.168.188.80:80 <pre>C:\Users\EdgarJung>ipconfig Windows-IP-Konfiguration Ethernet-Adapter Ethernet 3: Medienstatus : Medium getrennt Verbindungsspezifisches DNS-Suffix: Ethernet-Adapter Ethernet 2: Verbindungsspezifisches DNS-Suffix: fritz.box IPv6-Adresse. : 2003:e8:370b:1700:fede:c01e:5d9a:6add IPv6-Adresse. : fdd6:ee81:ff48:0:47c0:1060:342:eeaa Temporäre IPv6-Adresse. . . . : 2003:e8:370b:1700:d93a:577f:926:5132 Temporäre IPv6-Adresse. . . . : fdd6:ee81:ff48:0:d93a:577f:926:5132 Verbindungslokale IPv6-Adresse : fe80::cdef:26e5:85af:ab9f%11 IPv4-Adresse : 192.168.188.80 Subnetzmaske : 255.255.255.0 Standardgateway : fe80::6b4:feff:fe9b:75a1%11 192.168.188.1</pre> 

Strukturierung von Request/Response Werten (Body) in JSON (Java Script Object Notation)



Strukturierung von Request/Response Werten (Body) in JSON (Java Script Object Notation)

Object

```
{
  Frucht: "Banane"
}
```

```
header("Content-Type: application/json; charset=UTF-8");
$object1["Frucht"] = "Banane";
echo "object: " . json_encode($object1). "<br>".PHP_EOL;
```

ObjectArray

```
[
  {
    Frucht: "Banane"
  },
  {
    Gemuese: "Tomate"
  }
]
```

```
header("Content-Type: application/json; charset=UTF-8");
$object1["Frucht"] = "Banane";
$object2["Gemuese"] = "Tomate";
$objectArray[] = $object1;
$objectArray[] = $object2;

echo "objectArray: " . json_encode($objectArray). "<br>".PHP_EOL;
```

Array

```
["Banane", "Kirsche", "Pfirsich"]
```

```
header("Content-Type: application/json; charset=UTF-8");
$array = array("Banane", "Kirsche", "Pfirsich");
echo "array: " . json_encode($array). "<br>".PHP_EOL;
```

NestedObject

```
{
  Essbares:
  {
    Frucht: ["Banane", "Kirsche", "Pfirsich"],
    Gemuese: ["Tomate"]
  }
}
```

```
header("Content-Type: application/json; charset=UTF-8");

$object1["Frucht"] = "Banane";
$object2["Gemuese"] = "Tomate";

$array = array("Banane", "Kirsche", "Pfirsich");

$nested1["Frucht"] = $array;
$nested2 = $object2;
$nestedObject["Essbares"] = $nested1+$nested2;

echo "nestedObject: " . json_encode($nestedObject). "<br>".PHP_EOL;
```

Kompletter Server Code

```
<?php
header("Content-Type: application/json; charset=UTF-8");
echo "Mein Name lautet Stefan Seidel<br>";

$object1["Frucht"] = "Banane";
$object2["Gemuese"] = "Tomate";

$array = array("Banane", "Kirsche", "Pfirsich");

$objectArray[] = $object1;
$objectArray[] = $object2;

$multiObject = $object1+$object2;

$nested1["Frucht"] = $array;
$nested2 = $object2;
$nestedObject["Essbares"] = $nested1+$nested2;

echo "object: " . json_encode($object1). "<br>".PHP_EOL;
echo "array: " . json_encode($array). "<br>".PHP_EOL;
echo "objectArray: " . json_encode($objectArray). "<br>".PHP_EOL;
echo "multiObject: " . json_encode($multiObject). "<br>".PHP_EOL;
echo "nestedObject: " . json_encode($nestedObject). "<br>".PHP_EOL;
?>
```

Ausführen mit: `php -S localhost:8080 demoServer.php`

Werte anstatt im Browser, in einem Programm empfangen. Request + Response

Kompletter Server Code (demoCurlClient.php)

```
<?php

// Initialize cURL session
$ch = curl_init();

// Set the URL to fetch
curl_setopt($ch, CURLOPT_URL, "http://localhost:8080");

// Return the transfer as a string instead of outputting it
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

// Execute the cURL session
$response = curl_exec($ch);

// Check for errors
if (curl_errno($ch)) {
    echo 'cURL Error: ' . curl_error($ch);
} else {
    // Output the response
    echo $response;
}

// Close the cURL session
curl_close($ch);
?>
```

Reiner API Code (ohne Beschreibung, demoServer.php)

```
[
  {
    Frucht: "Banane"
  },
  {
    Gemuese: "Tomate"
  }
]

header("Content-Type: application/json; charset=UTF-8");
$object1["Frucht"] = "Banane";
$object2["Gemuese"] = "Tomate";
$objectArray[] = $object1;
$objectArray[] = $object2;

echo json_encode($objectArray);
```

Ausführen mit: `php demoCURLClient.php`

Ausführen mit: `php -S localhost:8080 demoServer.php`

Einschränkungen, Limits

Erweiterte Einstellungen

(Allgemein)

Verhalten

Ablaufverfolgung für Anforderungsfehler

Aktivierte Protokolle: http

HSTS

Limits

Max. URL-Segmente	32
Maximale Anzahl gleichzeitiger Verbindungen	4294967295
Maximale Bandbreite (Bytes/Sekunde)	4294967295
Verbindungstimeout (Sekunden)	120

(Allgemein)

OK Abbrechen

Einstellungen für die Anforderungsfilterung bearbeit...

Allgemein

☒ Nicht aufgelistete Dateinamenerweiterungen zulassen

☒ Nicht aufgelistete Verben zulassen

☒ High-Bit-Zeichen zulassen

☐ Doppelte Escapezeichen zulassen

Anforderungslimits

Maximal zulässige Inhaltslänge (Bytes): 30000000

Maximale URL-Länge (Bytes): 4096

Maximale Länge einer Abfragezeichenfolge (Bytes): 2048

OK Abbrechen

PHP Einstellungen

Konfiguriere alle Einstellung der Datei php.ini. Diese Einstellungen steuern alle Aspekte

Filter: Start Alle anzeigen Gruppieren nach: Sekt

Name	Wert	Sektion
cgi.fix_pathinfo	1	PHP
cgi.force_redirect	0	PHP
default_charset	"UTF-8"	PHP
default_mimetype	"text/html"	PHP
default_socket_timeout	60	PHP
disable_classes		PHP
disable_functions		PHP
display_errors	Off	PHP
display_startup_errors	Off	PHP
doc_root		PHP
enable_dl	Off	PHP
engine	On	PHP
error_log	"C:\Windows\Temp\php-8.3...."	PHP
error_reporting	E_ALL & ~E_DEPRECATED	PHP
expose_php	On	PHP
extension_dir	"C:\Program Files\PHP\PHP8...."	PHP
fastcgi.impersonate	1	PHP
fastcgi.logging	0	PHP
file_uploads	On	PHP
html_errors	Off	PHP
ignore_repeated_errors	Off	PHP
ignore_repeated_source	Off	PHP
implicit_flush	Off	PHP
include_path	".;./;../;./;./"	PHP
log_errors	On	PHP
max_execution_time	800	PHP
max_file_uploads	20	PHP
max_input_time	60	PHP
memory_limit	128M	PHP
output_buffering	4096	PHP
post_max_size	16M	PHP
precision	14	PHP
register_argc_argv	Off	PHP
report_memleaks	On	PHP
request_order	"GP"	PHP
serialize_precision	-1	PHP
short_open_tag	Off	PHP
unserialize_callback_func		PHP
upload_max_filesize	2M	PHP
upload_tmp_dir	"C:\Windows\Temp\"	PHP
user_dir		PHP
variables_order	"GPCS"	PHP
zend.enable_gc	On	PHP
zend.exception_ignore_args	On	PHP
zend.exception_string_param_...	0	PHP
zlib.output_compression	Off	PHP

Problem der Fehlerbehandlung – zu wenige Status Codes. Keine Aussagekraft

Response Envelope und Data

```
{
  Status: "ERROR",
  Message: "MaLo did not match",
  Reference: "See EDI Process Document N513.7",
  Backend:
    {
      Runtime: "Development",
      EDIVersion: "0.1",
      Module: "Semantic Check",
    },
  Data:
    {
      ...
    },
}
```

Json Schema

Entire set of control and content

Status string

required

Even if we've got 2X from API, there might be a nested Error to handle at client

Allowed values: `ERROR` `WARNING` `OK`

Example: `ERROR`

Message string

required

Message in Envelope Header

Example: `MaLo did not match`

Reference string

Description, what happend for improved error analysis

Example: `See EDI Process Document N513.7`

▼ **Backend** object

— **Runtime** string

required

Type of System

Allowed values: `Development` `Quality` `Production`

Example: `Development`

— **EDIVersion** string

required

Version of Interface, consulted by EDI@ENERGY

Match pattern: `\\d\\d\\d\\d`

— **Module** string

required

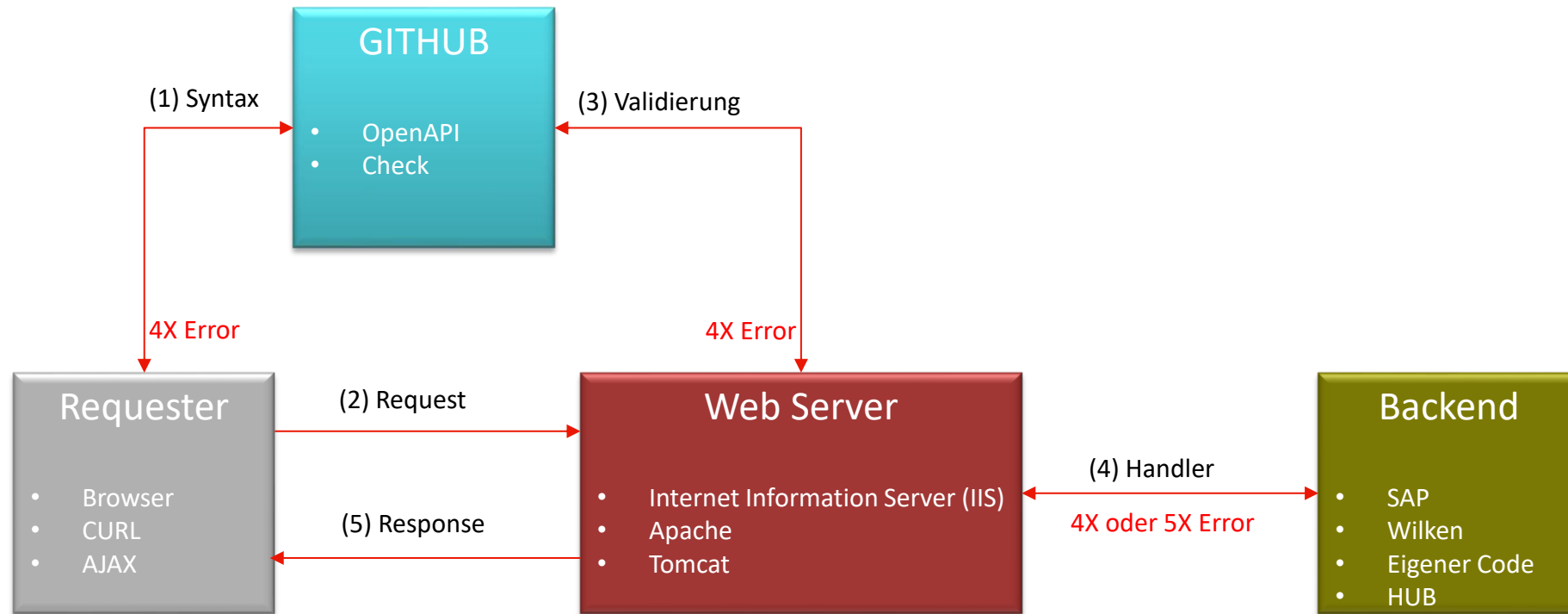
Hint, which Module is affected

Data object

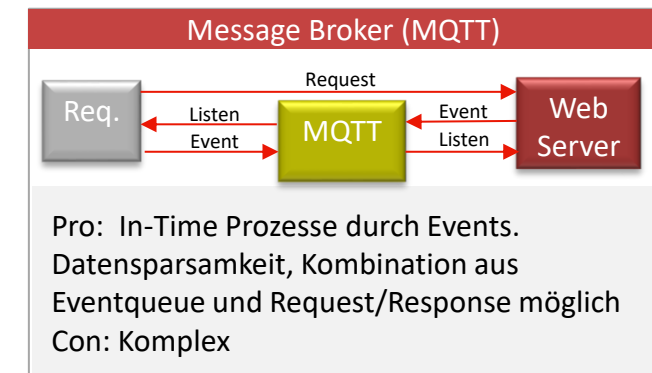
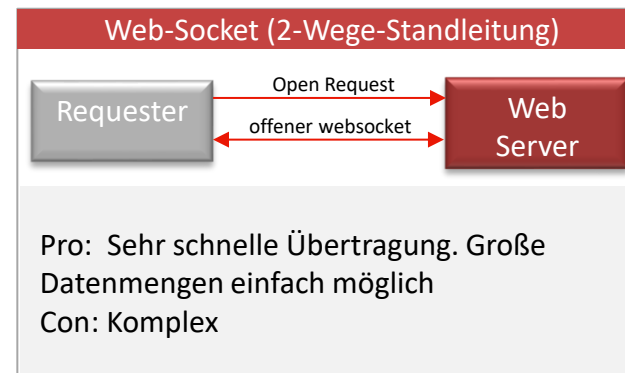
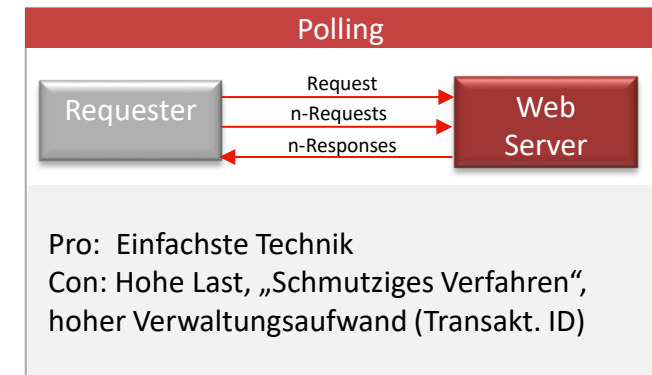
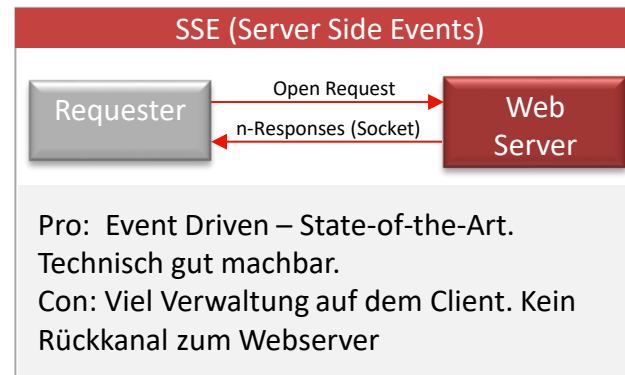
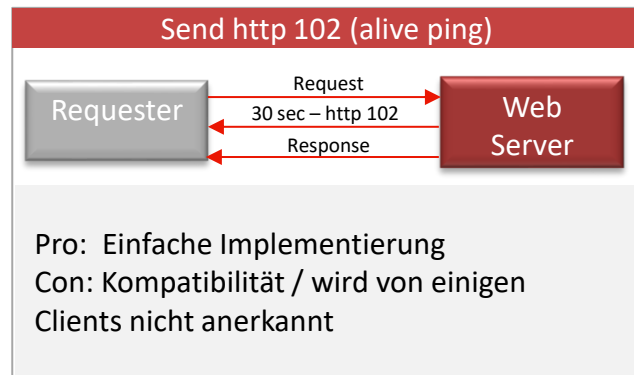
Real Data to transport

```
type: object
properties:
  Status:
    $ref: Status.yaml
  Message:
    $ref: Message.yaml
  Reference:
    $ref: Reference.yaml
  Backend:
    $ref: Backend.yaml
  Data:
    type: object
    properties: {}
    description: Real Data to transport
required:
  - Status
  - Message
description: Entire set of control and content
```

Art der Fehlerbehandlung (Syntax, Semantik)



Umgang mit Langläufern (grundsätzlich 30 sec bis 120 sec zwischen Request und Response, sonst Timeout 408, 504)



Vielen Dank für Ihre
Aufmerksamkeit