Creativ Software AG
Unterdorfstrasse 83
CH-9443 Widnau
www.creativ.ch

creati</> SOFTWARE

Dokumentation

Dokumentation
OM REST Web-API

Letzte Änderung: 12.08.2024
Version: 15.3
Klassifizierung: zur Herausgabe für externe IT-Dienstleister

Inhaltsverzeichnis

OM REST Web-API 3
Beschreibung 3
Versionenverzeichnis 3
1. Externe Schnittstellenbeschreibung OM REST Web-API 4
1.1. Authentisierung 4
1.2. Beispiel-Code für Erstellung Authorization-Header (C#) 5
1.3. Beispiel Post-Request (Postman) 5
1.4. Meta-Daten-Abfrage 6
1.5. Methoden-Beschreibung 6
1.6. http-Status-Codes von «OM REST Web-API» 9

OM REST Web-API

Beschreibung

Diese Dokumentation dient der applikatorischen Anbindung an die Schnittstelle. Die URL zur API ist von Kunde zu Kunde unterschiedlich und wird separat kommuniziert.

Versionenverzeichnis

Version	Mutationsgrund	Autor	Datum
1.0	Erstellung Dokumentation	Daniel Méndez (CSA)	10.04.2018
1.1	Timestamp-Unterstützung für GET	Daniel Méndez (CSA)	23.05.2018
1.2	- Versionenverzeichnis hinzugefügt - PATCH Methoden-Beschreibung hinzugefügt - DELETE Methoden-Beschreibung ergänzt - POST Methode unterstützt ID-Vergabe auf Consumer-Seite	Daniel Kobe (CSA)	30.05.2018
1.3	- Ergänzung bezüglich ID-Vergabe auf Consumer-Seite - Trennung der beiden Dokumentationsteile (Installationsanleitung und Dokumentation) - Ergänzung Fehlercode 409	Daniel Méndez (CSA)	11.12.2018
1.4	- Support für Url-Query für GET Liste (wenn keine ID gesendet hinzugefügt) - Metadaten-Abfrage für Query-Parameter (/params) - Neue Absicherung wg. ObjectID's (502er Fehler: OM-Error occurred (no ObjectID returned for one or more objects) - GUID-Format-Toleranz im Objekt	Daniel Méndez (CSA)	31.01.2019
1.5	Timestamp-Unterstützung für GET nicht mehr generell	Daniel Méndez (CSA)	06.02.2019
1.6	Einführung 9xx Status Codes für Hersteller spezifische Fälle - 901 – Ressource not available	Jonas Graf (CSA)	14.08.2019
1.7	Änderung der Status Codes 9xx. Hersteller spezifische Fälle sind neu ab Status Code 550 +	Jonas Graf (CSA)	29.08.2019
1.8	Code-Sample C# hinzugefügt für Authorization Header	Daniel Méndez (CSA)	12.11.2019
1.9	~~502 network to OM failed~~ Diesen Fehlercode gibt es nicht mehr, aufgrund einer technischen Änderung.	Daniel Méndez (CSA)	22.01.2019
1.10	410 Response-Code hinzugefügt	Daniel Méndez (CSA)	24.10.2020
1.11	550 Response-Code hinzugefügt	Jonas Graf (CSA)	18.05.2021
1.12	Die übrigen 502-Fehler wurden geändert in 500. Diesen Fehlercode (502) gibt es nicht mehr, aufgrund einer technischen Änderung.	Daniel (CSA)	29.07.2021
1.13	206 Partial Content hinzugefügt.	Jonas Graf (CSA)	08.11.2021
1.14	Fehler korrigiert PUT gibt 204 zurück (dokumentiert war bisher 200)	Daniel Méndez (CSA)	26.10.2022
1.15	«503 - OM is in maintenance mode» dokumentiert	Daniel Méndez (CSA)	03.11.2022
15.2	Diverse kleinere Abweichungen in der reason-phrase korrigiert	Daniel Méndez (CSA)	12.08.2024
15.2	POST gibt JSON des erstellten Objekts zurück (bisher no-content)	Daniel Méndez (CSA)	12.08.2024

1. Externe Schnittstellenbeschreibung OM REST Web-API

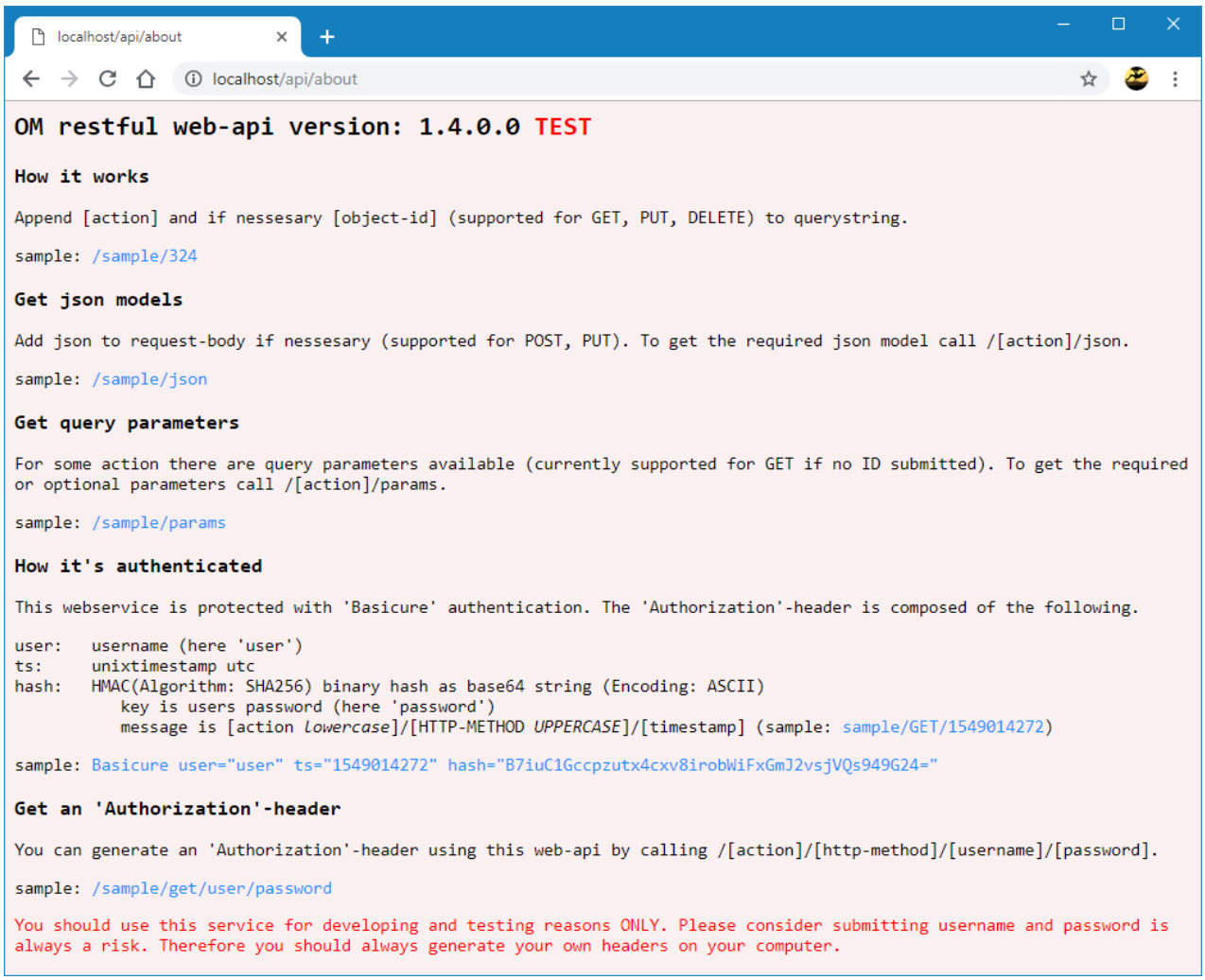
Die Schnittstelle «OM REST Web-API» wurde nach REST-Standard implementiert. Unterstützt werden die http-Methoden «GET», «POST», «PUT», «PATCH» und «DELETE».

1.1. Authentisierung

Alle Methoden der Webservice-Schnittstelle sind durch «Basicure»-Authentisierung geschützt. Folgende Methoden sind generell ohne Authentisierung erreichbar:

- /about
- /[action]/json (siehe 1.3)
- /[action]/params (siehe 1.3)

Das JSON-Modell kann also ohne Authentisierung abgefragt werden. Ebenfalls können die verfügbaren Query-Parameter pro «action» ohne Authentisierung abgefragt werden. Dies soll die Anbindung der Schnittstelle erleichtern.
Möglicherweise gibt es einzelne «actions», welche ohne Authentisierung ausgeführt werden können; Aber grundsätzlich gilt, dass alle «actions» nur mit Authentisierung ausgeführt werden können. Alles Weitere zur Authentisierung ist detailliert beschrieben auf der Startseite des Webservice /api/about. Die URL des Webservice, sowie der Benutzername und das Passwort zur Authentisierung werden separat kommuniziert.

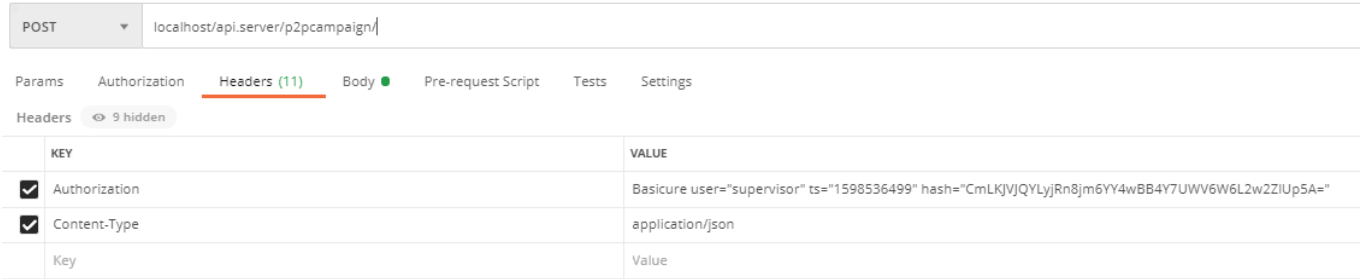


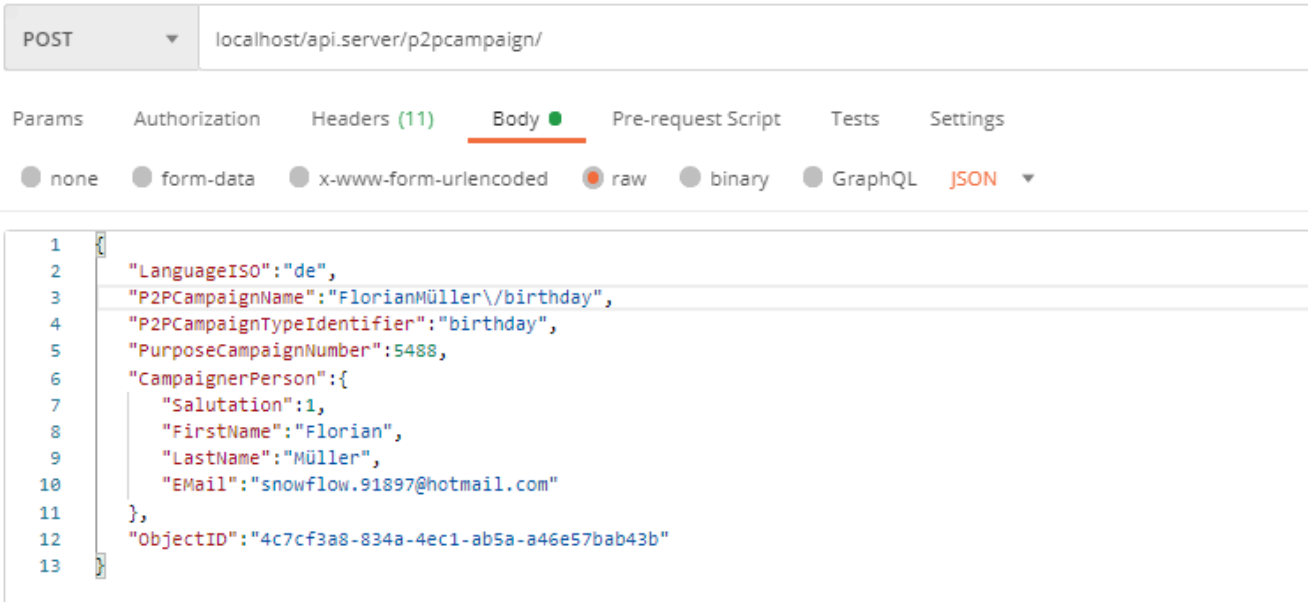
1.2. Beispiel-Code für Erstellung Authorization-Header (C#)

```
using System;
using System.Security.Cryptography;
using System.Text;

namespace Authorization
{
public static class BasicureHelper
{
    /// <summary>
    /// Creates Authorization-header for basicure-authenticated rest-api
    /// </summary>
    /// <param name="action">api-controller action (samples: contact, membership, campaign…)</param>
    /// <param name="httpMethod">http-method using in the current request (samples: get, put, post…)</param>
    /// <param name="username">your username you have received from the api-administrator (sample: api_user1)</param>
    /// <param name="password">your password you have received from the api-administrator (sample: Test123$)</param>
    /// <returns>Authorization-header ready to put in the current http-request-headers at index "Authorization"</returns>
    public static string CreateBasicureAuthorizationHeader(string action, string httpMethod,
                                                            string username, string password)
    {
        // get utc now unix timestamp in ms
        var unixTimeStart1970 = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc);
        var utcTimeStamp = (long)(TimeZoneInfo.ConvertTimeToUtc(DateTime.UtcNow) - unixTimeStart1970).TotalSeconds;

        // for TESTING your Code can overwrite the utcTimeStamp variable to have a constant value
        /* utcTimeStamp = 1573546842; */

        // concatenate authorization string from api-action, HTTP-method, utc timestamp
        var authorizationString = $"{action?.ToLower()}/{httpMethod?.ToUpper()}/{utcTimeStamp}";

        // compute base64 HMAC-SHA256 authorization-hash from password (secret)
        string base64AuthorizationHash;
        using (var hashGenerator = new HMACSHA256(Encoding.ASCII.GetBytes(password)))
        {
            var authorizationHash = hashGenerator.ComputeHash(Encoding.ASCII.GetBytes(authorizationString));
            base64AuthorizationHash = Convert.ToBase64String(authorizationHash);
        }

        // concatenate authorization header string
        var authorizationHeader = $"Basicure user=\"{username}\" ts=\"{utcTimeStamp}\" hash=\"{base64AuthorizationHash}\"";

        return authorizationHeader;
    }

    // TEST YOUR CODE:
    // With utcTimeStamp fixed to:
    // [1573546842]
    // return value of the call:
    // [Authorization.BasicureHelper.CreateBasicureAuthorizationHeader("sample", "get", "user", "password");]
    // results in:
    // [Basicure user="user" ts="1573546842" hash="quo9q8wYU4re8L9t6y3By+bWqfeOvKrLP+k0JdUi3bs="]
}
}
```

1.3. Beispiel Post-Request (Postman)

Übermittlung einer P2Pcampaign via Postman an den REST-Webservice von localhost.





Status Response, wenn der REST-Service den Request akzeptiert hat:

Status Response bei ungültigen Werten:

1.4. Meta-Daten-Abfrage

/[action]/json

Bei jeder «action» kann unter dem Pfad /json das json-Objekt-Modell angefragt werden.

/[action]/params

Bei jeder «action» können unter dem Pfad /params die verfügbaren query-parameter abgefragt werden. Es ist ersichtlich, ob der betreffende Parameter zwingend ist, welchen Namen der Parameter hat und von welchem Typ, dass er ist. Die Parameter sind nur für die GET-Methode verfügbar und werden anstelle einer ID verwendet. Sprich Sie können NICHT in Kombination mit einer ID verwendet werden. Beispiel der Parameter-Information:

```
[{
   “Name”: “searchTerm”,
   “Type”: “string”,
   “IsMandatory”: true
},
{
   “Name”: “ts”,
   “Type”: “long”,
   “IsMandatory”: false
}]
```

1.5. Methoden-Beschreibung

Im Folgenden sind die unterstützten Methoden detailliert beschreiben. Die verschiedenen «actions» sind in diesem Dokument nicht beschrieben, da sie laufend erweitert werden. Diese werden separat kommuniziert. Sämtliche «actions» funktionieren mit den beschriebenen Methoden. Die Methoden werden jeweils anhand der Beispiel-«action» «**address**» beschrieben.

GET

Über die Methode GET kann entweder ein einzelner Datensatz oder eine Liste von Datensätzen abgerufen werden.

Methode	Request	Erfolgreicher Response
Einzelner Datensatz abrufen	/[action]/[ID] z.B. /address/64805c09-7048-4f94-a3ab-c2a7826fdb3a Die ID muss eine GUID sein.	Code: 200 Reason-Phrase: «request processed by OM» JSON-Object im Response-Body: `{ "LanguageISO": "de", "Salutation": 1,…}`
Liste von Datensätzen abrufen	/[action] z.B. /address?ts= 1549034169	Code: 200; Reason-Phrase: «request processed by OM» JSON-Object-Array im Response-Body: `[{ "LanguageISO": "de", "Salutation": 1,…},{…}…]`

POST

Über die Methode POST können einzelne Datensätze erstellt werden.

Wichtiger Hinweis: Wir unterstützen in der POST-Methode die ID-Vergabe auf Consumer-Seite. Auf diese Weise können doppelte Datenlieferungen verhindert werden.

Methode	Request	Erfolgreicher Response
Einzelner Datensatz erstellen	/[action] z.B. /address Body: `{ "LanguageISO": "de", "Salutation": 1, ObjectID: "`**`64805C09-7048-4F94-A3AB-C2A7826FDB3A`**`" …}`	Code: 201 Reason-Phrase: «object [address] created by OM (64805C09-7048-4F94-A3AB-C2A7826FDB3A)». Die ObjectID wird in diesem Beispiel vorgegeben. Dies ist jedoch optional. Location-Header: …/address/64805C09-7048-4F94-A3AB-C2A7826FDB3A Die neue ID lautet in diesem Beispiel: **64805C09-7048-4F94-A3AB-C2A7826FDB3A** JSON-Object im Response-Body: `{ "LanguageISO": "de", "Salutation": 1,…}`

PUT

Über die Methode PUT können einzelne Datensätze mutiert werden.

Methode	Request	Erfolgreicher Response
Einzelner Datensatz mutieren	/[action]/[ID] z.B. /address/64805C09-7048-4F94-A3AB-C2A7826FDB3A Die ID muss eine GUID sein. Body: `{ «LanguageISO»: «de», «Salutation»: 1,…}`	Code: 204 Reason-Phrase: «object [address] updated by OM (64805C09-7048-4F94-A3AB-C2A7826FDB3A)»

PATCH

Über die Methode PATCH können bestimmte Eigenschaften von einzelnen Datensätzen mutiert werden.

Methode	Request	Erfolgreicher Response
Einzelner Datensatz mutieren	/[action]/[ID] z.B. /address/64805C09-7048-4F94-A3AB-C2A7826FDB3A Die ID muss eine GUID sein. Body: `{ «LanguageISO»: «de», «Salutation»: 1 }`	Code: 204 Reason-Phrase: «object [address] updated by OM (64805C09-7048-4F94-A3AB-C2A7826FDB3A)»

DELETE

Über die Methode DELETE können einzelne Datensätze gelöscht werden.

Methode	Request	Erfolgreicher Response
Einzelner Datensatz löschen	/[action]/[ID] z.B. /address/64805C09-7048-4F94-A3AB-C2A7826FDB3A Die ID muss grundsätzlich eine GUID sein. Falls eine bestimmte Action eine andere ID unterstützt, ist dies in der jeweiligen Action-Spezifikation explizit erwähnt.	Code: 200 Reason-Phrase: «request processed by OM»

1.6. http-Status-Codes von «OM REST Web-API»

Im Folgenden sind alle Responses und deren Bedeutung deklariert, welche von «OM REST Web-API» an die aufrufende Instanz übermittelt werden können. Alle Platzhalter in den «Reason-Phrases» sind *kursiv*.

200 – 299 Erfolgreiche Operationen

Code	Reason-Phrase	Beschreibung
200	request processed by OM	Die Anfrage wurde erfolgreich von OM verarbeitet (für «GET», «PUT», «DELETE»)
201	object [*action*] created by OM (*newid*)	Das übermittelte Objekt wurde von OM erstellt (z.B. für «POST»). Es wird ein «location»-header gesetzt, welcher den Link zum neuen Objekt (inkl. ID) enthält.
204	object [*action*] updated by OM (*newid*)	Das übermittelte Objekt wurde von OM aktualisiert (z.B. für «PATCH»).
206	Partial Content. The request exceeded the maximum number of objects returned. Number of objects returned: *number-of-objects -returned*	Es wurde nur ein Teilresultat geliefert da das gesamte Resultat zu gross ist. Die maximale Anzahl zurückgelieferte Objekte wird im Response Status Text angegeben. Wird nicht im Zusammenhang mit Pagination (Offset und numberOfRows) verwendet.

400 – 499 Client-Fehler

Code	Reason-Phrase	Beschreibung
400	id is missing	Es wurde keine ID übermittelt obwohl diese zwingend erwartet wird. (z.B. für «DELETE»).
400	ID [*ID*] is obsolete	Es wurde eine überflüssige ID übermittelt (z.B. für «POST»).
400	json is missing or invalid	Es wird ein JSON-Objekt (z.B. für «POST») erwartet jedoch wurde keines oder ein ungültiges übermittelt.
400	json is unexpected or invalid	Es wurde ein gültiges JSON-Objekt eines unerwarteten Typs übermittelt (z.B. ein JSON-Objekt-Array anstatt eines einzelnen JSON-Objekts).
400	json is obsolete	Es wurde ein JSON-Objekt übermittelt, obwohl keines erwartet wurde (z.B. für «DELETE»).
400	*Specific Reason-Phrase for «BadRequest»*	Je nach aufgerufener «action» können spezifische Fehlermeldungen zurückgegeben werden. (z.B. «missing saltutioncode» für die action «address»)
400	query [?searchTerm=peter+meier] is obsolete	Es wurde eine Query übermittelt obwohl keine erwartet wurde (z.B. bei POST).
400	missing mandatory query parameters [ts]	Ein zwingender Query-Parameter wurde nicht übermittelt.
400	query parameters malformed [ts]	Ein Query-Parameter hatte ein unerwartetes Format (z.B. Buchstaben anstatt Zahlen)
400	obsolete query parameters [searchTerm]	Es wurde ein unerwarteter/unbekannter Query-Parameter übermittelt.

Code	Reason-Phrase	Beschreibung
401	AuthorizationResult: Unknown (0)	Es wurde keine Autorisierung übermittelt, obwohl eine erwartet wurde.
401	AuthorizationResult: Unauthorized (2)	Die übermittelte Autorisierung ist fehlgeschlagen.
401	AuthorizationResult: `UnknownUser` (3)	Der übermittelte Benutzer existiert nicht.
401	AuthorizationResult: TimestampTooOld (4)	Die übermittelte Autorisierung ist fehlgeschlagen, weil der Zeitstempel zu alt war.
401	AuthorizationResult: TimestampTooYoung (5)	Die übermittelte Autorisierung ist fehlgeschlagen, weil der Zeitstempel zu jung war (liegt in der Zukunft).
404	action [*action*] not found	Die aufgerufene «action» (z.B. «membership») existiert nicht.
404	object [*action*] not found (*id*)	Die Objekt-ID wurde für die aufgerufene action nicht gefunden.
405	method [*method*] not allowed	Die aufgerufene http-Methode ist nicht implementiert (z.B. «PATCH»).
405	method [*method*] not allowed for [*action*]	Die aufgerufene http-Methode wird grundsätzlich unterstützt, jedoch nicht von der angefragten «action» (z.B. «DELETE» für «address»)
409	given object-id(*object-ID*) already exists for [*action*]	Die «ObjectID», welch dem json-objekt per HTTP-POST übermittelt wurde existiert bereits. Location-Header des bestehenden Objekts wird zurückgegeben.
410	object [*action*] no longer available (*id*)	Die Objekt-ID wurde für die aufgerufene action gefunden, jedoch wurde das Objekt bereits gelöscht oder war nur temporär verfügbar (z.B. bei Legacy-Services aus SOAP-Ablösung).

500 – 599 Server-Fehler

Code	Reason-Phrase	Beschreibung
500	InternalServerError	Es ist ein allgemeiner Fehler in der API-Schnittstelle aufgetreten. Der Request konnte möglicherweise nicht an OM übermittelt werden.
500	OM-Error occurred (*errorcode*)	Es ist ein Fehler aufgetreten, während OM die Anfrage verarbeitete. Der Fehlercode bezieht sich auf die OM-Applikation und sollte zur Unterstützung der Fehleranalyse an den OM-Support kommuniziert werden.
~~**502**~~	~~network to OM failed~~	~~Es konnte keine Antwort von OM empfangen werden.~~
~~**502**~~	~~unkown response from OM~~	~~Die Antwort von OM konnte nicht interpretiert werden.~~
500	fatal error occurred (details in txt-logs)	Die Anfrage endete in einem Absturz, oder die Antwort von OM konnte nicht interpretiert werden.
500	OM-Error occurred (unkown user [*username*])	Der Benutzer wurde serverseitig erfolgreich authentisiert; jedoch wurde er nicht vollständig in OM erfasst.

Code	Reason-Phrase	Beschreibung
500	Error: «…..» (*-code*)	Es ist ein Fehler in der Verbindung mit OM aufgetreten. Möglicherweise ist OM temporär nicht verfügbar. (mehrere Fehlermeldungen sind möglich z.B. «*There are no Magic xpa Servers currently available to serve this Application» (-104)*)
500	OM-Error occurred (no ObjectID returned for one or more objects)	Eine Liste von Objekten wurde angefordert. Die behandelnde «action» hat für eine oder mehrere Objekte keine «ObjectID» zurückgegeben. Dies wird unterbunden um Fehler auf der Empfängerseite vorzubeugen.
503	OM is in maintenance mode	OM steht temporär nicht zur Verfügung, da Wartungsarbeiten durchgeführt werden. Bitte versuchen Sie es später erneut.

550 – 599 Proprietäre Fehler

Code	Reason-Phrase	Beschreibung
550	Ressource not yet available	Die Ressource, auf die im OM zugegriffen werden will ist im System vorhanden aber noch nicht verarbeitet worden. Die Anfrage soll an einem späteren Zeitpunkt nochmals geschickt werden.
550	Wait for previous process to complete.	Eine vor kurzem gesendete, ähnliche Anfrage steht im Konflikt mit der jetzigen Anfrage. Sobald der Prozess der ersten Anfrage abgeschlossen ist (BusinessProcess PATCH isApproved = true), kann die neue Anfrage gemacht werden.