Publisher Webservice

- Verwaltung von Projekten über die V6 REST-API
- Publisher-Verwaltung über die V6 REST-API

Verwaltung von Projekten über die V6 REST-API

Einleitung

Die V6 REST-API bietet Admins die Möglichkeit, Projekte effizient zu verwalten. Mit dem `/Projects`-Endpoint können Projekte erstellt, ausgelesen, bearbeitet und gelöscht werden. Zusätzlich können Projekte anhand verschiedener Parameter gefiltert werden.

Endpoint

Für Admins: `https://SUBDOMAIN.de/ws/V6/admin/JSON/Projects`

Parameter & Filter

Erforderliche Parameter (bei POST und PUT)

Parameter	Typ	Beschreibung
`title`	String	Titel des Projekts (Pflichtfeld)
`status`	Integer	Status des Projekts (Pflichtfeld)
`url`	String	URL des Projekts (Pflichtfeld)

Filteroptionen (bei GET)

Es kann nach allen Spalten der Tabelle `publisher.projects` gefiltert werden, z. B.:

- `id`
- `status`
- `publisher_id`
- `projecttype`
- `title`
- `url`
- `channel_id`
- `hidden`

Beispiel für GET mit Filtern:

```
curl -X GET -H "Content-Type: application/json" -H "X-Network-ID: -1" -H "X-Auth-Token: ADMIN_APIUSER_TOKEN" -H "X-Auth-ID: ADMIN_APIUSER_LOGIN_ID" "https://SUBDOMAIN.de/ws/V6/admin/JSON/Projects?status=1&publisher_id=10"
```

CRUD-Operationen

GET: Abrufen von Projekten

Abrufen einer Liste von Projekten oder eines spezifischen Projekts anhand der `id`.

```
curl -X GET -H "Content-Type: application/json" -H "X-Network-ID: -1" -H "X-Auth-Token: ADMIN_APIUSER_TOKEN" -H "X-Auth-ID: ADMIN_APIUSER_LOGIN_ID" "https://SUBDOMAIN.de/ws/V6/admin/JSON/Projects?id=5"
```

POST: Erstellen eines neuen Projekts

```
curl -X POST -H "Content-Type: application/json" -H "X-Network-ID: -1" -H "X-Auth-Token: ADMIN_APIUSER_TOKEN" -H "X-Auth-ID: ADMIN_APIUSER_LOGIN_ID" -d '{
  "title": "Neues Projekt",
  "status": 1,
  "url": "https://example.com"
}' "https://SUBDOMAIN.de/ws/V6/admin/JSON/Projects"
```

PUT: Aktualisieren eines bestehenden Projekts

```
curl -X PUT -H "Content-Type: application/json" -H "X-Network-ID: -1" -H "X-Auth-Token: ADMIN_APIUSER_TOKEN" -H "X-Auth-ID: ADMIN_APIUSER_LOGIN_ID" -d '{
  "id": 5,
  "title": "Geändertes Projekt",
  "status": 2,
  "url": "https://updated-example.com"
}' "https://SUBDOMAIN.de/ws/V6/admin/JSON/Projects"
```

DELETE: Löschen eines Projekts

Löschen eines Projekts anhand der `id`.

```
curl -X DELETE -H "Content-Type: application/json" -H "X-Network-ID: -1" -H "X-Auth-Token: ADMIN_APIUSER_TOKEN" -H "X-Auth-ID: ADMIN_APIUSER_LOGIN_ID" "https://SUBDOMAIN.de/ws/V6/admin/JSON/Projects?id=5"
```

Antworten (Responses)

Erfolgreiche Antwort

```
{
  "id": 5,
  "status": 1,
  "publisher_id": 10,
  "projecttype": "page",
  "title": "Projektbeispiel",
  "description": "Beschreibung des Projekts",
  "url": "https://example.com",
  "channel_id": 2,
  "project_identifier_string": "abc123",
```

```
  "reach": "1000",
  "statistic_type": "internal",
  "external_sources_entity_id": "-1",
  "trackingtemplate": "template",
  "kpi_whitelist": "kpi1,kpi2",
  "hidden": false,
  "login_type": "pub",
  "exclude_from_salary": 0,
  "default_project": false,
  "insert_timestamp": "2025-01-27T12:00:00"
}
```


Fehlerhafte Anfrage


```
{
  "error": "Missing required field: title"
}
```


Hinweise zur Fehlerbehandlung

- Bei fehlenden Pflichtfeldern (`title`, `status`, `url`) wird ein entsprechender Fehler zurückgegeben.
- Ungültige oder nicht vorhandene `id` bei GET, PUT oder DELETE resultieren in einer leeren Antwort oder einer Fehlermeldung.

Publisher-Verwaltung über die V6 REST-API

Authentifizierung

Erforderliche Header für alle Anfragen:

Header	Beschreibung
`X-Network-ID`	Netzwerk-ID, z.B. `-1` für global
`X-Auth-Token`	API-User-Token
`X-Auth-ID`	API-User-Login-ID
`Content-Type`	`application/json`

Endpunkt

Base URL: `https://SUBDOMAIN.de/ws/V6/admin/REST/Publisher`

Es werden die üblichen HTTP-Methoden verwendet: `POST` zum Anlegen, `PUT` zum Aktualisieren, `GET` zum Abrufen.

CREATE Operation (POST)

Methode & Pfad: `POST /ws/V6/admin/REST/Publisher`

Erforderliche Parameter

Parameter	Typ	Beschreibung
`email`	String	Max. 255 Zeichen

Parameter	Typ	Beschreibung
`salutation`	String	z.B. `mr` / `mrs`
`prename`	String	Max. 255 Zeichen
`surname`	String	Max. 255 Zeichen

Optionale Parameter

`company`, `street`, `zip`, `city`, `telephone`, `country` (ISO 3166 ALPHA-3), `billing_sepa_owner`, `billing_sepa_iban`, `billing_sepa_bic`, `companytype` (z.B. `un` / `priv`), `language_interface` (Default: `DEU`), `billing_mode` (Default: `1`), `billing_media` (Default: `1`), `country_billing` (Default: `DEU`), `country_publisher` (Default: `DEU`), `billing_limit` (Default: `25`), `dialog_email` (Default: Wert von `email`)

> Für `salutation`, `companytype` und `country` gibt es keine feste Werteliste, die serverseitig erzwungen wird — bitte trotzdem an die üblichen Werte halten, damit nachgelagerte Auswertungen (Rechnungsstellung, Sprachwahl etc.) korrekt funktionieren.

```
curl -X POST -H "Content-Type: application/json" -H "X-Network-ID: -1" -H "X-Auth-Token:
ADMIN_APIUSER_TOKEN" -H "X-Auth-ID: ADMIN_APIUSER_LOGIN_ID" -d '{
  "email": "max.mustermann@example.com",
  "salutation": "mr",
  "prename": "Max",
  "surname": "Mustermann",
  "company": "Musterfirma",
  "street": "Musterstraße 1",
  "zip": "12345",
  "city": "Musterstadt",
  "country": "DEU",
  "billing_sepa_owner": "Max Mustermann",
  "billing_sepa_iban": "DE12345678901234567890",
  "billing_sepa_bic": "GENODEF1M01",
  "language_interface": "DEU",
  "billing_mode": 1,
  "billing_media": 1,
  "country_billing": "DEU",
  "country_publisher": "DEU",
```

```
  "billing_limit": 50
}' "https://SUBDOMAIN.de/ws/V6/admin/REST/Publisher"
```

Response

```
{
  "data": {
    "id": 123,
    "name": "Max Mustermann",
    "email": "max.mustermann@example.com"
  },
  "log": "no logmessage provided",
  "sessionId": "",
  "session": null
}
```

Die eigentlichen Daten stehen im `data`-Feld.

UPDATE Operation (PUT)

Methode & Pfad: `PUT /ws/V6/admin/REST/Publisher/{id}`

Ein bestehender Publisher wird über seine ID im URL-Pfad aktualisiert.

Aktualisierbare Felder Es können dieselben Felder übergeben werden, die auch beim Anlegen verwendet werden, z. B. `status`, `company`, `street`, `zip`, `city`, `telephone`, `country`, Billing-Felder oder Sprach-/Abrechnungswerte.

Welche Felder konkret geändert werden, hängt davon ab, welche Werte im Request-Body mitgegeben werden.

```
curl -X PUT -H "Content-Type: application/json" -H "X-Network-ID: -1" -H "X-Auth-Token:
ADMIN_APIUSER_TOKEN" -H "X-Auth-ID: ADMIN_APIUSER_LOGIN_ID" -d '{
  "status": 1
}' "https://SUBDOMAIN.de/ws/V6/admin/REST/Publisher/123"
```

Response

```
{
  "data": {
    "id": 123
  },
  "log": "no logmessage provided",
  "sessionId": "",
  "session": null
}
```

Es werden nur die Felder geändert, die im Request übergeben werden.

READ Operation (GET)

Methode & Pfad: `GET /ws/V6/admin/REST/Publisher` bzw. `GET /ws/V6/admin/REST/Publisher/{id}`

Einzelnen Publisher abrufen

`GET /ws/V6/admin/REST/Publisher/{id}` — `id` ist Teil des URL-Pfads, keine Query-Parameter.

Liste abrufen (mit Filtern)

Filter werden als Query-Parameter im Format `params[feld]=wert` übergeben, z.B.:

```
GET /ws/V6/admin/REST/Publisher?params[status]=1
```

Filterbar sind sowohl die Basis-Felder des Publisher-Accounts (`id`, `status`, `type`) als auch alle Felder aus dem Registrierungsprozess (`email`, `prename`, `surname`, `company`, `street`, `zip`, `city`, `telephone`, `country`, Billing-Felder etc.). Mehrere Filter zusammen werden UND-verknüpft, z.B.:

```
GET /ws/V6/admin/REST/Publisher?params[status]=5&params[prename]=Max
```

Filter auf Registrierungsfelder sind exakte Treffer (kein Teilstring-/Wildcard-Match).

Status-Werte

Bekannte Werte für `status`:

Status	Bedeutung
`1`	Bestätigter, aktiver Publisher
`5`	Neu registriert, noch nicht freigegeben

(Für eine vollständige Übersicht aller möglichen Status-Werte bitte im Admin-Interface unter der Publisher-Verwaltung nachsehen.)

Response-Format der Liste

```
{
  "data": {
    "123": {
      "type": "pub",
      "id": 123,
      "status": 5,
      "settings": {
        "email": "max.mustermann@example.com",
        "prename": "Max",
        "surname": "Mustermann",
        "company": "",
        "street": "Musterstraße 1",
        "zip": "12345",
        "city": "Musterstadt",
        "country": "DEU",
        "...": "weitere im Registrierungsprozess gespeicherte Felder"
      }
    }
  },
  "log": "no logmessage provided",
  "sessionId": "",
  "session": null
}
```

Das `settings`-Objekt enthält alle für den Publisher gespeicherten Registrierungsdaten — welche Felder konkret enthalten sind, hängt davon ab, was beim jeweiligen Publisher tatsächlich hinterlegt wurde (kein festes Schema).

> **Bekannte Einschränkung:** Informationen zur Werbefläche (Werbefläche/Werbeflächen-URL), die im Registrierungsprozess ebenfalls abgefragt werden, sind über diese API aktuell **nicht** abrufbar. Das wird als Erweiterung nachgezogen.

Beispiele

- Einzelner Publisher: `GET /ws/V6/admin/REST/Publisher/123`
- Neu registrierte, noch nicht freigegebene Publisher: `GET /ws/V6/admin/REST/Publisher?params[status]=5`
- Publisher per E-Mail suchen: `GET /ws/V6/admin/REST/Publisher?params[email]=max.mustermann@example.com`

Fehlerbehandlung

Fehler werden in folgendem Format zurückgegeben:

```
{
  "error": true,
  "msg": "<Fehlerbeschreibung>",
  "type": "Ws\\Error",
  "data": []
}
```

Der HTTP-Statuscode ist bei Fehlern durchgängig `500`.

Fehler	Beispiel-`msg`
Fehlendes Pflichtfeld bei CREATE	`internal server error. Details: Missing required field: email`
Publisher nicht gefunden	`internal server error. Details: Model not found.`
Filter ohne Treffer	kein Fehler — liefert `"data": []`