Order Webservice

- Create / easy.affiliate REST-API (Orders)
- Update / easy.affiliate REST-API (Orders)
- Create / easy.affiliate REST-API (Baskets)
- Update / easy.affiliate REST-API (Baskets)

Create / easy.affiliate REST-API (Orders)

□□ Einleitung

Um neue Orders automatisiert im easy.affiliate-System anzulegen, steht ein REST-konformer Webservice zur Verfügung. Über die HTTP-Methode **POST** können neue Transaktionen übermittelt werden. Jeder User erhält hierfür einen Authentifizierungs-Token und eine Login-ID, die im Frontend einsehbar sind.

□□ Authentifizierung

Für den Zugriff sind folgende Header erforderlich:

Header	Beschreibung	Typ
Content-Type	Muss auf `application/json` gesetzt sein	String
X-Network-ID	Netzwerk-ID: meist `-1` für mandantenübergreifende API	Integer
X-Auth-Token	Dein API-Token (im Frontend sichtbar)	String
X-Auth-ID	Deine Login-ID (im Frontend sichtbar)	Integer

□□ Endpunkte

- **Admin:** `https://SUBDOMAIN.de/ws/V6/admin/JSON/Orders`
- **Advertiser:** `https://SUBDOMAIN.de/ws/V6/advertiser/JSON/Orders`

□□ Beispiel (cURL)

```
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 '[{
    "ordertoken": "testorder001",
    "campaign_id": 1,
    "trigger_id": 2,
    "status": 1,
    "turnover": 199.99,
    "trigger_tstamp": "1785315641"
  }]' \
  https://SUBDOMAIN.de/ws/V6/admin/JSON/Orders
```

□□ Body-Parameter

Parameter	Beschreibung	Datentyp	Pflichtfeld
ordertoken	Eindeutiger Identifier für die Order. Muss systemweit eindeutig sein. Wird kein Wert übergeben, generiert das System automatisch einen (`AUTOID_…`).	String	□ Ja
campaign_id	ID der Kampagne, der die Order zugeordnet wird.	Integer	□ Ja
trigger_id	ID des Triggers, der bei Nutzung von `ordertoken` zwingend angegeben werden muss. Alternativ per `trigger_title`.	Integer	□ Ja
status	Status der Order: `0` = offen, `1` = bestätigt, `2` = storniert. *(Bei der Anlage immer `0`.)*	Integer	□ Ja
turnover	Bestellwert in Kampagnenwährung. Komma und Punkt als Dezimaltrenner erlaubt.	Float	□ Ja

Parameter	Beschreibung	Datentyp	Pflichtfeld
commission	Provisionswert in Kampagnenwährung. Kann übergeben werden, falls selbst berechnet.	Float	Nein
cancel_reason	Angabe eines Stornogrundes bei Status = 2 (storniert).	String	Nein
currency	Bestellwährung als 3-stelliger ISO-Code (z. B. `EUR`, `CHF`). Default = Kampagnen-Einstellung, sonst `EUR`. Umrechnungskurs wird automatisch gesetzt.	String	Nein
descr	Freitext-Beschreibung der Order (max. 255 Zeichen).	String	Nein
trigger_tstamp	Bestellzeitpunkt als Unix-Timestamp. Fehlt der Wert, wird die aktuelle Zeit genutzt.	Integer	Nein
emid	Klick-/Action-ID aus dem Tracking (24-stellig). Dient der Klick-Zuordnung und kann die `campaign_id` auflösen.	String	Nein*
vc	Gutscheincode-Tracking: löst Kampagne/Werbemittel über die Admedia-Einstellungen auf.	String	Nein*
project_id	Direkte Zuordnung per Projekt (Publisher). Erzeugt eine Track-Action, sofern eine Connection zur Kampagne besteht.	Integer	Nein*
subid	Attributionswert (SubID), wird am Auftrag gespeichert (max. 500 Zeichen).	String	Nein
sub_status	Substatus der Order (frei belegbar).	String	Nein
get_parameters	Objekt `{ "key": "value", … }` mit beliebigen Zusatzfeldern (z. B. `voucher`, `coupon`).	Object	Nein

Parameter	Beschreibung	Datentyp	Pflichtfeld
(beliebiger Key)	Jeder weitere, nicht reservierte Parameter wird als Key/Value am Auftrag gespeichert (Key max. 25 Zeichen).	String	Nein

> * **Kampagnen-/Klick-Zuordnung:** `campaign_id` ist Pflicht, kann aber alternativ über `emid` oder `vc` aufgelöst werden. Für die Zuordnung zur ursprünglichen Klick-/View-Aktion sollte **eine** der Attributions-Methoden (`emid` oder `project_id`) mitgegeben werden.

> **Dublettenschutz:** Dieselbe Kombination aus `campaign_id` + `trigger_id` + `ordertoken` wird innerhalb abgelehnt (`ORDER_EXISTS` / `ORDER_REQUEST_FREEZED`).

□□ Hinweise

Wichtige Implementierungsdetails:

- **ordertoken:** Muss **eindeutig** sein – idealerweise eine Kombination aus Zeitstempel, Shop-ID oder externer Ordernummer.
- **trigger_id:** Wenn du diese nicht kennst, frage im Frontend deine Kampagnenkonfiguration ab.
- **Bulk-Aktionen:** Du kannst auch mehrere Orders in einem Array auf einmal anlegen.
- **Updates:** Wird eine Order mit gleichem `ordertoken` bereits im System gefunden, kann ein `PUT-Request` genutzt werden, um sie zu aktualisieren.

Update / easy.affiliate REST-API (Orders)

Mit diesem Aufruf ändern Sie bestehende Transaktionen (Orders): Status setzen, Umsatz korrigieren, Stornogrund hinterlegen. Einzeln oder viele auf einmal. Neue Transaktionen legt dieser Aufruf nie an, dafür gibt es Create Orders.

Den kompletten Ablauf eines Sales-Abgleichs, auch per Datei-Upload und SFTP, beschreibt Automatisierter Sales-Abgleich.

Endpunkte

Login	URL
Admin	`https://<Ihre-Domain>/ws/V6/admin/REST/Orders`
Advertiser	`https://<Ihre-Domain>/ws/V6/advertiser/REST/Orders`

Methode: `PUT`. Statt `REST` funktioniert auch `JSON` im Pfad, beide verhalten sich gleich.

Welche Transaktionen ein Login ändern darf:

- **Advertiser-Login:** nur Transaktionen der Kampagnen seines Advertisers.
- **Admin-Login mit Mandant:** nur Transaktionen der Kampagnen dieses Mandanten.
- **Admin-Login ohne Mandant:** alle Transaktionen der Installation.

Transaktionen außerhalb dieses Bereichs behandelt die API wie nicht vorhanden.

Header

Header	Wert
`Content-Type`	`application/json`
`X-Auth-ID`	Login-ID des Benutzers

Header	Wert
`X-Auth-Token`	Access-Token des Benutzers, zu finden unter **API → Daten-API**
`X-Network-ID`	ID des Mandanten. Bei nur einem Mandanten oder mandantenübergreifend `-1`.

Wo die Login-ID steht: Wo ist die User ID hinterlegt?

Beispiel

```
curl -X PUT "https://<Ihre-Domain>/ws/V6/admin/REST/Orders" \
  -H "Content-Type: application/json" \
  -H "X-Auth-ID: <LOGIN-ID>" \
  -H "X-Auth-Token: <ACCESS-TOKEN>" \
  -H "X-Network-ID: -1" \
  -d '[
    { "id": 12345, "status": 1 },
    { "id": 12346, "status": 2, "cancel_reason": "Retoure" },
    { "id": 12347, "status": 1, "turnover": 89.90 },
    { "campaign_id": 1, "ordertoken": "123a456b", "status": 2, "cancel_reason": "Storno" }
  ]'
```

Der Body ist eine Liste von Objekten, je Objekt eine Transaktion. Ein einzelnes Objekt ohne Liste geht auch.

Transaktion finden

Weg	Felder	Hinweis
Transaktions-ID (empfohlen)	`id`	eindeutig, mehr ist nicht nötig. Geben Sie zusätzlich `campaign_id` mit, muss sie zur Transaktion passen.
Bestellnummer	`campaign_id` + `ordertoken`	`campaign_id` ist hier Pflicht. Gibt es zur Bestellnummer mehrere Transaktionen in der Kampagne (z. B. Lead und Sale), ist die Zuordnung nicht eindeutig. Dann immer `id` verwenden.

Weg	Felder	Hinweis
Klick-ID	`emid` + `ordertoken`	Eine 24-stellige `emid` ersetzt die `campaign_id`, die Kampagne wird aus dem Klick ermittelt.

Felder

Feld	Bedeutung	Typ
`status`	`0` offen, `1` bestätigt, `2` storniert. Auch als Wort: `offen`, `bestätigt`, `storniert`, `open`, `confirmed`, `cancelled`. Andere Werte werden abgewiesen.	Integer
`turnover`	Netto-Umsatz in Kampagnenwährung, Punkt als Dezimaltrenner	Float
`original_turnover`	Umsatz in Fremdwährung. **Überschreibt `turnover`**, wenn gesetzt.	Float
`commission`	fester Provisionsbetrag. Nur mitgeben, wenn er bewusst von der Provisionsgruppe abweichen soll.	Float
`cancel_reason`	Stornogrund, Freitext	String
`status_change_date`	Datum der Statusänderung (ISO 8601). Ohne Angabe setzt das System es bei einem Statuswechsel selbst. Bestimmt auch das Abrechnungsdatum.	Datetime
`sub_status`	Substatus, frei belegbar	String
`description`	Beschreibung, höchstens 255 Zeichen	String
`order_currency`	Bestellwährung als ISO-Code, z. B. `EUR`	String
`subid_value`	SubID der Transaktion. Heißt hier `subid_value`, nicht `subid` wie beim Anlegen.	String
`get_parameters`	Objekt `{ "key": "value" }`, wird je Transaktion ergänzt oder überschrieben	Object

Felder, die Sie nicht mitschicken, bleiben unverändert.

Provision

Ohne `commission` gilt: Bei prozentualer Provision rechnet das System die Provision aus dem neuen `turnover` neu. Feste Provisionen bleiben, wie sie sind. Der Wert `"commission": "autocalculate"` bewirkt dasselbe und ist nicht nötig. Ein Zahlenwert in `commission` wird genau so übernommen.

Nicht änderbar

Ausgezahlte Transaktionen (Status 3) sind abgerechnet und lassen sich nicht mehr ändern. Status 3 kann auch nicht per API gesetzt werden, er entsteht bei der Auszahlung.

Antwort

```
{
  "data": [
    { "id": 12345, "status": 1, "turnover": 49.9, "commission": 4.99, "cancel_reason": null,
"status_change_date": "2026-10-28T14:50:44+00:00", ... },
    { "id": 12346, "status": 2, "turnover": 19.99, "commission": 2, "cancel_reason":
"Retoure", ... },
    { "id": 12347, "status": 1, "turnover": 89.9, "commission": 8.99, ... }
  ]
}
```

- `data` enthält jede geänderte Transaktion mit ihrem neuen Stand, so wie sie jetzt im System steht.
- **Gleichen Sie die IDs in `data` mit den gesendeten ab.** Fehlt eine, wurde sie nicht geändert: nicht gefunden, bereits ausgezahlt, außerhalb Ihres Bereichs oder ungültiger Status. In einer Liste gibt es dafür keine eigene Fehlermeldung.
- Schicken Sie ein einzelnes Objekt ohne Liste, steht die Transaktion direkt in `data`. Kann sie nicht geändert werden, kommt `"error": true`, der Grund steht in `msg`, zum Beispiel `status 7 is not allowed, use 0 (open), 1 (confirmed) or 2 (cancelled)`.

Im Beispiel oben fehlt die vierte Transaktion in der Antwort, wenn es die Bestellnummer `123a456b` in Kampagne 1 nicht gibt.

Fehler

Antwort	Bedeutung
`"message": "Authentification failed."`	Login-ID oder Access-Token falsch, oder der Login ist gesperrt
`"error": true` bei einem einzelnen Objekt	Transaktion nicht gefunden, nicht änderbar oder ungültiger Status. Der Grund steht in `msg`.
Transaktion fehlt in `data`	siehe Antwort

Jede Änderung steht in der Historie der Transaktion mit Zeitpunkt und Benutzer. Ein eigener Benutzer für die Anbindung, etwa „Salesabgleich API“, macht dort sofort sichtbar, welche Änderung aus der Schnittstelle kam.

Create / easy.affiliate REST-API (Baskets)

□□ Einleitung

Um neue Orders **inklusive Warenkorb** automatisiert im easy.affiliate-System anzulegen, steht ein REST-konformer Webservice zur Verfügung. Über die HTTP-Methode `POST` wird pro Aufruf eine Bestellung mit ihren Warenkorb-Positionen übermittelt.

Die Bestellung durchläuft dabei dieselbe Logik wie ein getrackter Verkauf mit Warenkorb (`/trck/ebasket/`): Publisher-Zuordnung, Provision je Position (Regeln nach Artikel, Marke, Kategorie oder Default des Triggers), Bestellwert sowie serverseitige Tags und Postbacks der Kampagne. **Bestellwert, Provision und Status werden immer vom System berechnet** und können nicht übergeben werden.

Jeder User erhält hierfür einen Authentifizierungs-Token und eine Login-ID, die im Frontend einsehbar sind.

□□ Authentifizierung

Für den Zugriff sind folgende Header erforderlich:

Header	Beschreibung	Typ
`Content-Type`	Muss auf `application/json` gesetzt sein	String
`X-Network-ID`	Netzwerk-ID des Mandanten (muss > 0 sein, **nicht** `-1`). Die Kampagne muss zu diesem Mandanten gehören.	Integer
`X-Auth-Token`	Dein API-Token (im Frontend sichtbar)	String
`X-Auth-ID`	Deine Login-ID (im Frontend sichtbar)	Integer

Berechtigung: Nur Admin-Logins mit Lese- und Schreibrecht auf die Bestellungen (`administration_orders`). Der Login muss zum Mandanten aus `X-Network-ID` gehören.

□□ Endpunkte

- **Admin:** `https://SUBDOMAIN.de/ws/V6/admin/REST/Baskets`

□□ Beispiel (cURL)

```
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 '{
    "campaign_id": 123,
    "trigger_id": 5,
    "ordertoken": "E-10045",
    "emid": "5f0123456789abcdef012345",
    "basket": [
      {"article_number": "A-1", "productname": "Hemd", "category": "Herren", "amount": 2,
"price": 19.90, "additional": {"brand": "X"}},
      {"article_number": "B-7", "amount": 1, "price": 5}
    ]
  }' \
  https://SUBDOMAIN.de/ws/V6/admin/REST/Baskets
```

□□ Body-Parameter (Order)

Parameter	Beschreibung	Datentyp	Pflichtfeld
`campaign_id`	ID der Kampagne, der die Order zugeordnet wird.	Integer	□ Ja
`trigger_id`	ID eines Triggers dieser Kampagne (z. B. Sale-Trigger).	Integer	□ Ja

Parameter	Beschreibung	Datentyp	Pflichtfeld
`ordertoken`	Eindeutiger Identifier für die Order (eigene Bestellnummer). 1–255 druckbare ASCII-Zeichen, ohne Leerzeichen und ohne `(` `)` `;`.	String	☐ Ja
`basket`	Liste der Warenkorb-Positionen (min. 1, max. 200), siehe unten.	Array	☐ Ja
`emid`	Klick-/Action-ID aus dem Tracking (24-stellig). Muss zu einem Klick dieser Kampagne gehören.	String	Nein*
`project_id`	Direkte Zuordnung per Projekt (Publisher). Das Projekt muss mit der Kampagne verbunden sein.	Integer	Nein*
`currency`	Bestellwährung als 3-stelliger ISO-Code (z. B. EUR, CHF). Default = Kampagnen-Einstellung, sonst EUR. Unbekannte Währungen werden abgelehnt.	String	Nein
`trigger_tstamp`	Bestellzeitpunkt als Unix-Timestamp (Sekunden). Fehlt der Wert, wird die aktuelle Zeit genutzt.	Integer	Nein
`descr`	Freitext-Beschreibung der Order (max. 255 Zeichen).	String	Nein

* **Publisher-Zuordnung:** Es darf **entweder** `emid` **oder** `project_id` übergeben werden, nicht beides. Beide werden vorab geprüft: Eine unbekannte/abgelaufene `emid`, eine `emid` einer anderen Kampagne oder ein nicht verbundenes Projekt wird abgelehnt, statt die Order still dem Systempartner zuzuordnen. Ohne beide Angaben entscheidet die Kampagne, ob die Order an den Systempartner geht (Antwort enthält dann `notice: NO_SESSION_FOUND`).

Nicht erlaubt: `turnover`, `commission`, `status` und alle weiteren, nicht aufgeführten Felder werden mit einem Fehler abgelehnt.

☐☐ Body-Parameter (Warenkorb-Position in `basket`)

Parameter	Beschreibung	Datentyp	Pflichtfeld
`article_number`	Artikelnummer / SKU (1–255 Zeichen). Provisionsregeln nach Artikel greifen darauf.	String	□ Ja
`amount`	Menge, ganze Zahl ≥ 1.	Integer	□ Ja
`price`	Preis **einer** Einheit, ≥ 0, **nur Punkt** als Dezimaltrenner (z. B. `19.90`), max. 4 Nachkommastellen. Bei Brutto-Tracking der Kampagne wird die MwSt. wie beim Tracking abgezogen.	Float	□ Ja
`productname`	Produktname (max. 255 Zeichen).	String	Nein
`category`	Kategorie für Provisionsregeln nach Kategorie. Leer = Artikelnummer.	String	Nein
`additional`	Objekt `{ "key": "value", … }` mit max. 50 flachen Zusatzfeldern, wird an der Position gespeichert. `additional.brand` wird für Provisionsregeln nach Marke genutzt. Keys: 1–50 Zeichen aus `A-Z a-z 0-9 _ -`; reservierte Keys (z. B. `currency`, `emid`, `campaign_id`, `commission`, `turnover`) sind nicht erlaubt.	Object	Nein

Der Bestellwert der Order ergibt sich aus der Summe `price × amount` aller Positionen, die Provision aus der Summe der Positions-Provisionen.

Dublettenschutz: Dieselbe Kombination aus `campaign_id` + `trigger_id` + `ordertoken` wird abgelehnt (`ORDER_EXISTS`), die Antwort enthält die `order_id` der vorhandenen Order. Ein erneuter Aufruf, z. B. nach einem Timeout, ist dadurch gefahrlos möglich.

□□ Antwort

Erfolg:

```
{
  "order_inserted": true,
  "order_id": 4711815,
  "emid": "5f0123456789abcdef012345",
  "sessionid": "5f0123456789abcdef000001",
  "basket": [
    {"id": 981234, "productid": "A-1", "productname": "Hemd"},
    {"id": 981235, "productid": "B-7", "productname": ""}
  ]
}
```

Fehler: `order_inserted: false`, `order_id: -1`, dazu `error` und `error_description`. Alle Eingabefehler werden gesammelt in einer Antwort geliefert.

Error	Bedeutung
`INVALID_REQUEST`	Eingabe ungültig (Pflichtfeld fehlt, Format falsch, unbekanntes Feld, Trigger gehört nicht zur Kampagne, `emid` / `project_id` passt nicht, Währung unbekannt). Details in `error_description`.
`ORDER_EXISTS`	Order mit diesem `ordertoken` existiert bereits für Kampagne und Trigger; `order_id` enthält die vorhandene Order.
`TRACKING_UNAVAILABLE`	Das Tracking hat nicht geantwortet, die Order wurde vermutlich nicht angelegt. Erneut senden (Dublettenschutz greift).
sonstige	Ablehnung durch die Tracking-Logik bzw. Kampagnen-Einstellungen.

□□ Hinweise

Wichtige Implementierungsdetails:

- **Ein Aufruf = eine Order:** Der Body ist ein einzelnes JSON-Objekt, kein Array. Mehrere Orders werden mit mehreren Aufrufen angelegt.
- **ordertoken:** Muss je Kampagne und Trigger eindeutig sein – idealerweise die externe Bestellnummer des Shops.
- **trigger_id:** Wenn du diese nicht kennst, frage im Frontend deine Kampagnenkonfiguration ab.
- **Preise:** Immer mit Punkt als Dezimaltrenner übergeben (`12.50`, nicht `12,50`).

Update / easy.affiliate REST-API (Baskets)

□□ Einleitung

Um Warenkorb-Positionen von Orders automatisiert im easy.affiliate-System bearbeiten zu können, stellt die easy Marketing GmbH eine Webservice-API zur Verfügung. Diese ermöglicht es, einzelne Positionen bestehender Orders im Nachgang zu bearbeiten – z. B. bei Teilretouren, Teilstornos oder Preiskorrekturen.

Nach jeder Änderung werden **Bestellwert, Provision und Status der Order automatisch aus ihren Positionen neu berechnet**.

Jedem User stehen hierfür ein Authentifizierungs-Token und eine Login_id zur Verfügung, die über das Frontend abgerufen werden können.

- **User-ID:** Wo ist die User-ID hinterlegt? (Siehe Frontend-Profil)
- **Access-Token:** Wo ist der Access-Token hinterlegt? (Siehe API-Einstellungen)

Endpunkte

Zielgruppe	URL
Admin	`https://SUBDOMAIN.de/ws/V6/admin/REST/Baskets`

Beispiele

Beispiel mit cURL

```
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": 981234, "amount": 1 },{ "id": 981235, "status": 2 }]' \
  https://SUBDOMAIN.de/ws/V6/admin/REST/Baskets
```

Codebeispiel

Mit der folgenden Methode können eine oder mehrere Warenkorb-Positionen geupdatet werden. Ein Beispielaufruf sieht folgendermaßen aus:

Headers

Variable	Wert / Bedeutung
Content-Type	application/json
X-Network-ID	-1
X-Auth-Token	ADMIN_APIUSER_TOKEN
X-Auth-ID	ADMIN_APIUSER_LOGIN_ID

Body

```
[
    {
        "id": 981234,
        "amount": 1
    },
    {
        "basket_item_id": 981235,
        "status": 2
    },
    {
        "id": 981236,
        "price": 17.50,
        "additional": { "retoure_grund": "zu klein" }
    }
]
```

Für eine einzelne Position kann statt der Liste auch direkt ein Objekt gesendet werden, z. B. `{ "id": 981234, "amount": 1 }`.

Variablenerläuterung

Headers

Variable	Bedeutung	Datentyp
Content-Type	Der Content-Type des Requests	String
X-NETWORKID	Hier wird die ID des Mandanten eingetragen. Wenn nur ein Mandant vorhanden ist oder mandantenübergreifend gearbeitet wird, muss der Wert “-1” eingetragen werden.	Integer
X-AUTH-TOKEN	Hier wird der API-Authentifizierungs-Token des Admin Nutzers hinterlegt.	String
X-AUTH-ID	Hier wird die ID des Admin Nutzers hinterlegt.	Integer

Body

Parameter	Beschreibung	Datentyp	Pflichtfeld
`id`	ID der Warenkorb-Position. Steht in der Antwort der Anlage (`basket[].id`) bzw. im Order-Export.	Integer	☑ Ja (oder `basket_item_id`)
`basket_item_id`	Alternative Schreibweise für `id`.	Integer	☑ Ja (oder `id`)
`amount`	Neue Menge der Position.	Integer	Nein
`price`	Neuer Preis **einer** Einheit, Punkt als Dezimaltrenner.	Float	Nein
`status`	Status der Position: 0 = offen, 1 = bestätigt, 2 = storniert, 3 = ausgezahlt.	Integer	Nein
`commission`	Provisionswert der Position in Kampagnenwährung. Wird er nicht übergeben, wird die Provision neu berechnet (Preis × Menge × Provisionssatz der Position).	Float	Nein

Parameter	Beschreibung	Datentyp	Pflichtfeld
`additional`	Objekt `{ "key": "value", … }`. Die übergebenen Keys werden in die bestehenden Zusatzfelder der Position eingefügt bzw. überschrieben; nicht übergebene Keys bleiben erhalten.	Object	Nein

*) Hinweis: Alle Werte außer `additional` müssen numerisch sein. Weitere Felder (z. B. `productname`, `article_number`, `campaign_id`) werden ignoriert und können über diesen Endpunkt nicht geändert werden.

□□ Auswirkung auf die Order

Nach jeder Änderung wird die zugehörige Order aus **allen** ihren Positionen neu berechnet:

Wert der Order	Berechnung
`turnover`	Summe aus `price × amount` aller Positionen mit Status 0, 1 oder 3 (stornierte Positionen zählen nicht)
`commission`	Summe der Provisionen aller Positionen mit Status 0, 1 oder 3
`status`	Niedrigster Status aller Positionen (eine offene Position hält die Order offen; sind alle Positionen storniert, ist die Order storniert)

Die Neuberechnung erfolgt nur, solange die Order den Status offen, bestätigt oder storniert hat. **Bereits ausgezahlte Orders (Status 3) werden nicht mehr verändert.**

□□ Antwort

Die Antwort enthält die gespeicherten Positionen mit ihren aktuellen Werten (bei einer Liste als Array, bei einem einzelnen Objekt als Objekt).

Bei einer Liste werden Positionen, die nicht gefunden wurden oder ungültige Werte enthalten, übersprungen und fehlen in der Antwort – die übrigen Positionen werden trotzdem gespeichert. Bei einem einzelnen Objekt antwortet der Webservice in diesem Fall mit einem Fehler.

□□ Hinweise

- **Teilretoure:** Menge reduzieren (`amount`) oder die betroffene Position stornieren (`status: 2`) – Bestellwert und Provision der Order passen sich automatisch an.
- **Komplettstorno:** Alle Positionen auf `status: 2` setzen, die Order wird dann ebenfalls storniert.
- **Positions-IDs:** Kommen aus der Antwort von *Create / easy.affiliate REST-API (Baskets)* oder per `GET` auf denselben Endpunkt.