Flex 2.0 API Payment Schnittstelle

Inhaltsverzeichnis

1. Allgemeines

Dieses Handbuch beschreibt die flex.API v2 zur Online-Abwicklung von Zahlungen.

1.1. Das einfach.sicher.zahlen-System

Das einfach.sicher.zahlen-System von secupay ermöglicht die Onlineabwicklung von Zahlungsvorgängen in Internetshops und auf Internetportalen.

Bei fehlerhaften Daten oder unzureichender Bonität des Endkunden erfolgt eine Ablehnung mit entsprechender Fehlermeldung. Der Kunde sollte dann auf die Seite mit der Auswahl der Bezahlarten zurück geleitet werden.

Zur erfolgreichen Durchführung einer Demo-Transaktion benötigt Ihr Vertrag die entsprechenden Konditionen. Sollte die Anfrage mit dem Fehlercode 0007 abgelehnt werden, melden Sie sich bitte beim Kundendienst für die Freischaltung.

1.2. Demoanfragen

Um Anfragen als Demo zu kennzeichnen und damit eine reale Buchung der Transaktion zu verhindern, ist bei den payment-Requests der Parameter `demo` vorgesehen. Wird der Parameter mit dem Wert “1” oder “true” als String oder boolean übermittelt, wird die Transaktion simuliert und es entstehen weder Transaktionskosten noch reale Buchungen.

2. Systembeschreibung

2.1. API

Die secupay Flex API, nachfolgend flex.API genannt, lässt besonders leicht mit einfachen HTTP(S)-POST-Anfragen im JSON-Format ansprechen.

Der Hostname der flex.API ist `api.secupay.ag`, das Protokoll `https://`, die Basis URL für alle Anfragen lautet deshalb:

```
https://api.secupay.ag/
```

Aus Sicherheitsgründen sind Anfragen auf Port 80 nicht verfügbar und werden serverseitig auf Port 443/HTTPS umgeschrieben.

2.1.1. Testumgebung

Damit Sie die Anbindung an die flex.API testen können, haben wir ein Testsystem eingerichtet. Sie können hier alle flex.API Aufrufe wie im Live System Testen ohne das Ihre Entwicklungen Auswirkungen auf Ihren Live Vertrag haben.

Nach Ihren erfolgreichen Tests, brauchen Sie nur die URL und den API-Key austauschen um auf dem Live System (siehe Punkt "API") zu arbeiten, alle anderen flex.API aufrufe bleiben gleich.

Verwenden Sie für Ihre Testumgebung folgende URL:

```
https://api-dist.secupay-ag.de/
```

Transaktionen werden hierbei nicht gebucht und es fallen keinerlei Anfragekosten an. Den API-Key für das Testsystem erhalten Sie auf Anfrage von unserem Kundendienst.

2.2. Pfade

Die Funktionen der flex.API sind über den anfragenden Pfad zu erreichen. Diese sind in verschiedene Namensräume unterteilt. Die Zahlfunktionen sind dabei unter /payment/<FUNKTION> zu erreichen.

Beispiele:

Namensraum: payment, Funktion: init

```
https://api.secupay.ag/payment/init
```

Namensraum: payment, Funktion: gettypes

```
https://api.secupay.ag/payment/gettypes
```

2.3. HTTP-Header

Anfragen an die flex.API benötigen HTTP Header um verschiedene Ein- und Ausgabeformate zu unterstützen.

Des Weiteren ist es möglich die Sprache der gemeldeten Fehler zu steuern.

2.3.1. Content-Type und Accept

Die Schnittstelle unterstützt zwei Ausgabeformate – JSON und XML. Innerhalb des secupay-Systems wird mit JSON gearbeitet und falls gewünscht in XML umgewandelt.

HTTP-Header Content-Type

Mit dem HTTP-Header Content-Type teilt der Client der flex.API mit, in welchem Format die Daten im HTTP-Body (PUT- oder POST-Methode) gesendet werden Daten werden im XML-Format gesendet

```
Content-Type: text/xml; charset=utf-8;
```

Daten werden in JSON gesendet

```
Content-Type: application/json; charset=utf-8;
```

Die Angabe charset=utf-8; hat keine Bedeutung, da die flex.API immer UTF-8 verwendet. Die Angabe „charset“ wird von der flex.API automatisch in UTF-8 geändert.

HTTP-Header Accept / Antwortformat

Mit dem HTTP-Header Accept bestimmt der Client in welchen Format die flex.API antworten soll.

Beispiel für das Empfangen von XML Ergebnissen:

```
Accept: text/xml;
```

Daten werden in JSON empfangen:

```
Accept: application/json;
```

Beim HTTP-Header Accept ist es üblich mehrere Datenformate anzugeben. Die flex.API wertet aber nur den ersten Wert aus. Es sollte also davon abgesehen werden mehr als ein Format im Accept-Header zu senden.

Die HTTP-Header Accept und Content-Type können beliebig gemischt werden. Es ist also möglich eine POST Anfrage in JSON zu senden und die Antwort im XML-Format zu erhalten.

2.4. Sprache

Die flex.API unterstützt derzeit Deutsch und Englisch für Fehlermeldungen & Fehlercodes.

So gibt es beispielsweise, bei falsch übermittelten Daten an die flex.API, Fehlercodes die vom Client-Programm abgefangen werden können.

Zum Anderen gibt es aber auch qualifizierte Fehlermeldungen, die dem Benutzer direkt in der entsprechenden Oberfläche angezeigt werden.

Ein typischer Validierungsfehler ist eine leere Eingabe bei einem Pflichtfeld.

Derzeit gültige Werte:

```
de_DE
```

sowie

```
en_US
```

2.5. Zeichensatz

Die flex.API verwendet als Zeichensatz ausschließlich UTF-8, sowohl für eingehende als auch für ausgehende Daten. Eine Überprüfung der eingehenden Daten hinsichtlich des Zeichensatzes findet nicht statt, das heißt der Client ist selbst für eine eventuell notwendige Konvertierung verantwortlich.

2.6. Datenstruktur

Unabhängig von dem angefragten Namensraum, sowie vom Format sind alle eingehenden und ausgehenden Daten einheitlich strukturiert. Alle Anfragen mit HTTP-Body kapseln die übertragenen Nutzdaten in data. Alle Antworten der flex.API sind in drei Bereiche untergliedert: status, data und errors.

Nachfolgend ein Beispiel zur Verdeutlichung:

```
{
  "status":"ok",
  "data":{
    "hash":"...",
    "iframe_url":"..."
  },
```

```
  "errors":null
}
```

2.6.1. Antwort: Status

Der Status zeigt an, ob die Anfrage erfolgreich war. Hierbei gibt es drei unterschiedliche Status:

ok - Die Anfrage an die flex.API war erfolgreich und die gewünschte Aktion wurde durchgeführt.

error - Die gewünschte Aktion konnte nicht durchgeführt werden, die Ursache ist technischer Natur. Solche Fehler deuten auf Bugs der flex.API oder des Clients hin. In diesem Fall ist der Eingriff eines Entwicklers erforderlich.

failed - Die gewünschte Aktion konnte nicht durchgeführt werden. Meist handelt es sich um Validierungsfehler, deshalb sollte dem Benutzer das Formular mit den Fehlermeldungen angezeigt werden.

2.6.2. Antwort: Data

Der Bereich data enthält die eigentlichen Nutzdaten. Die Struktur der Nutzdaten ist abhängig von dem Namensraum sowie der Funktion. Meist handelt es sich um eine einfache Name-Wert-Zuordnung oder Listen von Name-Wert-Zuordnungen.

Beachten: Die hier zurückgegebenen Daten können zukünftig um zusätzliche Wertpaare erweitert werden. Dies darf bei der Verarbeitung nicht zu Fehlern führen.

2.6.3. Antwort: Errors

Der Bereich wird nur beim Status „Failed“ oder „Error“ gesetzt. Dann enthält er eine Liste mit Fehlercodes und Fehlermeldungen.

3. Payment

3.1. Liste der verfügbaren Zahlungsarten

```
https://api.secupay.ag/payment/gettypes
```

Dieser Aufruf gibt eine Liste aller möglichen Zahlungsarten zurück. Diese Liste ist abhängig vom API-Key und kann sich dynamisch ändern. Die Zahlarten sind bei jedem Kunden der secupay in den entsprechenden Verträgen hinterlegt.

Es empfiehlt sich, diese Abfrage zur Auflistung der verfügbaren Zahlungsarten auszuführen, damit es nicht zu ungewollten Fehlermeldungen kommt. Wenn die entsprechende Zahlungsart nicht verfügbar sein sollte.

Die Abfrage gibt die jeweils verfügbaren Zahlungsarten in Echtzeit wieder.

3.1.1. Abfrage

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace"
```

```
    }
}
```

3.1.2. Antwort

```
{
  "status":"ok",
  "data":[
    "prepay",
    "debit",
    "invoice",
    "creditcard",
    "transfer",
    "payout",
    "sofort"
  ],
  "errors":null
}
```

3.2. Init / Einreichung einer Transaktion

```
https://api.secupay.ag/payment/init
```

Zu Beginn muss der Zahlvorgang dem secupay-System bekannt gemacht werden. Dabei werden die Informationen über die Zahlung vom Shop-System übertragen. secupay initialisiert damit den Zahlvorgang, legt die internen Transaktionsdaten an und erzeugt den Hash, der bei der weiteren Kommunikation verwendet wird.

Die übertragenen Informationen können je nach Zahlungsart und Shop-System unterschiedlich sein. Deshalb sind viele Angaben optional oder werden in manchen Fällen nicht berücksichtigt. Damit der Zahlvorgang initialisiert werden kann, sind daher vor allem die technischen Parameter wichtig.

Folgende Parameter sind für das Anlegen einer Zahlung zwingend zu übertragen:

- `apikey` - Der API.key dient zur Anmeldung am secupay-System
- `payment_type` - Dieser Wert gibt die Zahlart des Vorgangs an. Typische Werte sind debit oder creditcard . Beispiele finden Sie ab Punkt "Beispiel für Prepay (Vorkasse)"
- `amount` - Der Betrag in der kleinsten Einheit der Währung (z. B. Eurocent)
- `currency` - Währung (ISO-Code)
- `url_success` - Nach erfolgreicher Transaktion wird der Kunde auf diese Seite weitergeleitet
- `url_failure` - Nach Abbruch bzw. aufgetretenen Fehlern wird der Kunde auf diese Seite weitergeleitet
- `url_push` - Auf diese wird bei wesentliche Statusänderungen an der Transaktionen der neue Status per HTTP POST übermittelt (siehe Punkt "Push API")

Folgende Parameter sind für die Angabe des Zahlers zwingend zu übertragen:

- `firstname` - Vorname
- `lastname` - Name
- `company` - Firma (wenn vorhanden, ansonsten nicht übertragen)
- `email` - E-Mail-Adresse
- `street` - Straße
- `housenumber` - Hausnummer
- `zip` - PLZ
- `city` - Ort
- `country` - Land

Die Zahlung wird mit den in Abschnitt "Parameter" beschriebenen Parametern eingereicht. Die flex.API prüft die Daten auf Vollständigkeit. Wurden alle für die Transaktion benötigten Daten übergeben, liefert die flex.API den hash und den iFrame-Link der Transaktion zurück.

Zahlungsmitteldaten werden in der Regel von secupay in einem separat zur Verfügung gestellten iFrame abgefragt. Dazu antwortet die flex.API mit dem Hash und dem Link zum Öffnen des iFrames zur Eingabe. Im Folgenden wird zunächst von secupay der Genehmigungsprozess durchgeführt. Danach liefert die flex.API den Hash und den Status der Transaktion zurück.


```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "payment_type":"debit",
    "demo":1,
    "payment_action":"sale",
    "url_success":"http://shop.example.com/success.php",
    "url_failure":"http://shop.example.com/failure.php",
    "url_push":"http://shop.example.com/push.php",
    "apiversion":"2.11",
    "shop":"APITest Shop",
    "shopversion":"1.0",
    "moduleversion":"1.0",
    "language":"de_DE",
    "title":"Herr",
    "firstname":"Test FN",
    "lastname":"Test LN",
    "company":"Test Company",
    "street":"Test Street",
    "housenumber":"5t",
    "zip":"12345",
    "city":"TestCity",
    "country":"DE",
    "telephone":"+4912342134123",
    "dob_value":"01.02.1903",
    "email":"test@ema.il",
    "ip":"172.31.6.49",
    "amount":"199",
    "currency":"EUR",
    "purpose":"Test Order #1",
    "order_id":"100203",
```

```
    "delivery_address":{
      "firstname":"Test FN",
      "lastname":"Test LN",
      "company":"Test Company",
      "street":"Test Street",
      "housenumber":"5t",
      "zip":"12345",
      "city":"TestCity",
      "country":"DE"
    },
    "experience":{
      "positive":"1",
      "negative":"0"
    },
    "basket":[
      {
        "item_type":"shipping",
        "name":"standard delivery",
        "tax":"19",
        "total":"1324"
      },
      {
        "item_type":"article",
        "article_number":"3211",
        "quantity":"2",
        "name":"Testname 1",
        "ean":"4123412341243",
        "tax":"19",
        "total":"1324",
        "price":"1000"
      },
      {
        "item_type":"article",
        "article_number":"48875",
        "quantity":"2",
        "name":"Testname 1",
        "ean":"4123412341236",
        "tax":"19",
        "total":"1324",
        "price":"1000"
      }
    ]
  }
}
```

3.2.1. Antwort

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3303"
  },
  "errors":null
}
```

Über die zurückgegebene URL lässt sich nun die Zahlung vom Kunden beenden, es steht Ihnen offen die URL entweder als iFrame einzubetten oder per Weiterleitung aufzurufen. Die Eingabemasken sind grundsätzlich so aufgebaut, dass sie sich entsprechend des Platzangebots von selbst in der Darstellung anpassen.

Der Parameter “payment_action” steuert die Verarbeitung der Transaktion durch uns, vorerst gibt es hier die Werte “sale” und “authorization”. Sale ist dabei eine direkte Zahlung. Um die Transaktion erst später durchzuführen muss hier “authorization” übermittelt werden.

• experience - Optional siehe Punkt "Experience / Bewertung".

3.2.2. Beispiel für Prepay (Vorkasse)

Bei der Zahlungsart Prepay (Vorkasse) unterscheidet sich der Ablauf darin, das kein iFrame verwendet werden muss. Alle benötigten Informationen sind direkt in der Antwort auf das init-Request enthalten.

Diese Informationen müssen Ihrem Kunden in geeigneter Form zur Verfügung gestellt werden, damit die Vorkassentrasaktion später beglichen werden kann.

Ändern Sie für Prepay (Vorkasse) den payment_type in prepay in dem Beispiel für die Einreichung einer Transaktion.

```
{
  "data":{
    "payment_type":"prepay",
    ...
  }
}
```

Als Antwort erhalten Sie folgende Daten

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3303",
    "purpose":"TA 123456",
    "payment_data":{
      "accountowner":"secupay AG",
      "accountnumber":"0123456789",
      "bankcode":"01234567",
      "iban":"DE00012345678912345678",
      "bic":"COBADEFF103",
      "bankname":"Test Bank"
    }
  },
  "errors":null
}
```

- purpose - Der Verwendungszweck welcher für die Überweisung zu verwenden ist, um eine automatische Zuordnung der späteren Zahlung zur Transaktion zu ermöglichen.
- payment_data - Bankverbindung die für die Überweisung zu verwenden ist
- iframe_url - hat für prepay Transaktionen keine Verwendung

3.2.3. Beispiel für Debit (Lastschrift)

Ändern Sie für Debit (Lastschrift) den payment_type in debit in dem Beispiel für die Einreichung einer Transaktion.

```
{
  "data":{
    "payment_type":"debit",
    ...
  }
}
```

Als Antwort erhalten Sie folgende Daten:

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3303"
  },
  "errors":null
}
```

Leiten Sie den Zahler anschließend auf die *iframe_url* weiter, damit dieser die Zahlungsmittel-Daten eingeben kann und die Zahlung abschließen kann.

3.2.4. Beispiel für Sofort (Sofortüberweisung)

Ändern Sie für Sofort (Sofortüberweisung) den payment_type in sofort in dem Beispiel für die Einreichung einer Transaktion.

```
{
  "data":{
    "payment_type":"sofort",
    ...
  }
}
```

Als Antwort erhalten Sie folgende Daten:

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3303"
  },
  "errors":null
}
```

Leiten Sie den Zahler anschließend auf die *iframe_url* weiter, damit dieser zu Sofort weitergleitet werden kann um die Zahlung abschließen zu können.

3.2.5. Beispiel für Invoice (Rechnung)

Ändern Sie für Invoice (Rechnung) den payment_type in invoice in dem Beispiel für die Einreichung einer Transaktion.

- dob_value - Geburtsdatum - Pflichtfeld bei Rechnungskauf

```
{
  "data":{
    "payment_type":"invoice",
    "dob_value":"01.02.1903",
    ...
  }
}
```

- Als Antwort erhalten Sie folgende Daten:

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3303"
  },
  "errors":null
}
```

3.2.6. Beispiel für Creditcard (Kreditkarte)

Ändern Sie für Creditcard (Kreditkarte) den payment_type in creditcard in dem Beispiel für die Einreichung einer Transaktion.

```
{
  "data":{
    "payment_type":"creditcard",
    ...
  }
}
```

Als Antwort erhalten Sie folgende Daten:

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3303"
  },
  "errors":null
}
```

3.2.7. Beispiel für Transfer (Überweisung)

Bei der Zahlungsart Transfer (Überweisung zwischen verschiedenen secupay-Händlerkonten) müssen alle Zahlungsinformationen im init-Request übermittelt werden.

```
{
  "data":{
    "payment_type":"transfer",
    "purpose":"Test Order #1",
    "payment_data":{
      "accountowner":"Test LN",
      "iban":"DE00012345678912345678",
      "bic":"COBADEFF103"
    },
    ...
  }
}
```

Antwort:

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3303",
    "purpose":"TA 123456",
    "payment_data":{
      "accountowner":"secupay AG",
      "iban":"DE00012345678912345678",
      "bic":"COBADEFF103"
    }
  },
  "errors":null
}
```

• iframe_url - hat für transfer Transaktionen keine Verwendung

3.2.8. Beispiel für Payout (Zins- oder Rückzahlungen für Crowdfunding)

Bei der Zahlungsart Payout (Zins- oder Rückzahlungen für Crowdfunding) sind alle benötigten Informationen direkt in der Antwort auf das init Request enthalten und es wird kein iFrame verwendet.

Bei dieser Zahlart werden im Gegensatz zu Refund die Ursprungstransaktionen nicht beeinflusst.

Für Payout muss ein Zahlungseingang erfolgen. Der Transaktionswert kann nicht vom bestehenden Projektguthaben übernommen werden.

```
{
  "data":{
    "payment_type":"payout",
    "amount":"3200",
    "purpose":"Payout #1",
    "basket":[
      {
        "item_type":"transaction_payout",
        "name":"Payout Purpose",
        "transaction_hash":"tujevzgobryk3303",
        "total":"1000"
      },
      {
        "item_type":"transaction_payout",
        "name":"Payout Purpose",
```

```
        "transaction_hash":"tujevzgobryk3304",
        "total":"1200"
      },
      {
        "item_type":"transaction_payout",
        "name":"Payout Purpose",
        "transaction_hash":"tujevzgobryk3305",
        "total":"1000"
      }
    ]
  }
}
```

Basket Items: es können maximal 200 Items gleichzeitig übergeben werden)

- name - ist der Verwendungszweck (ist auf 160 Zeichen begrenzt)
- transaction_hash - Hash der Ursprungstransaktion
- total - der zu überweisende Betrag für diese Transaktion

Antwort:

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3306",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3306",
    "purpose":"TA 123456",
    "payment_data":{
      "accountowner":"secupay AG",
      "iban":"DE00012345678912345678",
      "bic":"COBADEFF103"
    }
  },
  "errors":null
}
```

- purpose - der Verwendungszweck der für die Überweisung zu verwenden ist, um eine Zuordnung der späteren Zahlung zur Transaktion zu ermöglichen.
- payment_data - Bankverbindung die für die Überweisung zu verwenden ist
- iframe_url - hat für payout Transaktionen keine Verwendung

3.3. API-Key Erzeugung per flex.API

Diese Funktion bietet eine Möglichkeit dynamisch einen API-Key zu erzeugen. Für den normalen Shop-Betrieb sollte diese Funktion nicht notwendig sein und ist auch nur für explizit freigeschaltete API-Key's gültig.

```
https://api.secupay.ag/payment/requestapikey
```

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "user":{
      "title":"Herr",
      "company":"Firma",
      "firstname":"Test FN",
      "lastname":"Test LN",
      "street":"Test Street",
      "housenumber":"5t",
      "zip":"12345",
      "city":"TestCity",
      "country":"TestCountry",
      "telephone":"+4912342134123",
      "industry":"4812",
      "dob_value":"01.02.1903",
      "email":"test@ema.il",
      "homepage":"www.test.de"
    },
    "payout_account":{
      "accountowner":"Test Inhaber",
      "iban":"DE00012345678912345678",
      "bic":"COBADEFF103"
    },
    "project":{
      "name":"APITest Shop"
    },
    "iframe_opts":{
      "show_basket":true,
      "basket_title":"Ihr Warenkorb",
      "submit_button_title":"Kostenpflichtig Bestellen!"
    },
    "payin_account":false
  }
}
```

3.3.1. API-Key Erzeugung per flex.API

3.3.1.1. Spezifische Parameter

Bis auf iframe_opts sind alle Parameter Pflichtangaben.

- user - Vertragsdaten des Projektinhaber
- payout_account - Konto auf welches die Transaktionen ausgezahlt werden sollen
- project - Projekt spezifische Eigenschaften, vorerst nur der Name des Projekts
- iframe_opts - definiert eine Liste von Standards welche die Darstellung des iFrames beeinflussen
    - show_basket - Soll im Iframe ein Basket angezeigt werden, true/false
    - basket_title - Der Titel des Warenkorbes
    - submit_button_title - Der Text des Absenden Buttons
    - logo_base64 - Das zu hinterlegende Logo für das iFrame als base64 Darstellung
    - cession - Ansprache im iFrame, ’personal’ oder ’formal’
- payin_account - Abfrage eines virtuellen Kontos (setzt eine entsprechende Freischaltung dieser Option für den jeweiligen Vertrag voraus) als Ziel für Überweisungen. Dieser Parameter gibt

zusätzlich eine mit dem Vertrag verknüpfte Bankverbindung, die IBAN und die BIC eines virtuellen Kontos zurück.

logo_base64 sowie cession werden falls nicht explizit übermittelt aus dem vorhandenen Vertrag übernommen.

Antwort


```
{
  "status":"ok",
  "data":{
    "apikey":"48411xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx744e",
    "contract_id":"1234",
    "payin_account":{
      "accountowner":"Vorname Name des Vertragspartners",
      "iban":"DE00012345678912345678",
      "bic":"COBADEFF103"
    }
  },
  "errors":null
}
```


3.4. Statusabfrage

```
https://api.secupay.ag/payment/status
```

Über diese Funktion kann der Status einer Transaktion und weitere Informationen zu einem Hash abgefragt werden. In Zukunft kann die Antwort um weitere Informationen erweitert werden.

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "hash":"tujevzgobryk3303"
  }
}
```

Antwort:


```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "status":"accepted",
    "created":"2013-06-10 10:27:42",
    "demo":0,
    "trans_id":"1831201",
    "amount":"100",
    "opt":{

    }
  },
```

```
  "errors":null
}
```

Der im `data` Teil der Antwort enthaltene Status muss nach der Weiterleitung auf `url_success` auf `"accepted"` stehen. Eine Ausnahme kann bei Vorkasse und vorautorisierten Transaktionen bestehen → `"proceed"`.

Es wird empfohlen, nachdem die Weiterleitung auf `url_success` erfolgt ist, durch eine Statusabfrage den Status `"accepted"` bzw. `"proceed"` zu verifizieren.

Im Property `"opt"` werden Payment-type-spezifische Rückantworten übermittelt. Zum Beispiel: Bei `invoice` werden hier die Kontodaten für die Überweisung der Rechnung übermittelt, um dem Kunden z. B. in einer E-Mail darüber zuinformieren. Derzeit wird das Property nur bei Vorkasse und Rechnungskauf-Transaktionen zurückgeliefert, in der Zukunft macht dies aber auch bei anderen Zahlarten Sinn. Neue Implementierungen sollten diese Property daher grundsätzlich beachten.

3.4.1. Zusätzliche Rückgabeparameter bei einer Statusanfrage Invoice

Bei einer Statusanfrage auf `invoice`-Transaktionen bekommen Sie zusätzliche Felder im opt-Property zurückgeliefert:

Antwort:

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "status":"accepted",
    "created":"2013-06-10 10:27:42",
    "demo":0,
    "trans_id":"1831201",
    "amount":"100",
    "currency":"EUR",
    "opt":{
      "recipient_legal":"secupay AG, Goethestraße 6, 01896 Pulsnitz",
      "payment_link":"https://api.secupay.ag/payment/tujevzgobryk3303",
      "payment_qr_image_url":"https://api.secupay.ag/qr?d=http%3A%2F%2F....",
      "transfer_payment_data":{
        "purpose":"TA 123456 DT 20130610",
        "accountowner":"secupay AG",
        "iban":"DE00 0123 4567 8912 3456 78",
        "bic":"COBADEFF103",
        "bankname":"Test Bank"
      },
      "invoice_number":"182018014170",
      "shipping_date":"2018-11-12 08:50:40",
      "shipped":true
    }
  },
  "errors":null
}
```

3.4.2. Zusätzliche Rückgabeparameter bei einer Statusanfrage Prepay

Bei einer Statusanfrage auf `prepay`-Transaktionen bekommen Sie zusätzliche Felder im opt-Property zurückgeliefert:

Antwort:

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "status":"proceed",
    "created":"2013-06-10 10:27:42",
    "demo":0,
    "trans_id":"1831201",
    "amount":"100",
    "opt":{
      "recipient_legal":"secupay AG, Goethestraße 6, 01896 Pulsnitz",
      "payment_link":"https://api.secupay.ag/payment/tujevzgobryk3303",
      "payment_qr_image_url":"https://api.secupay.ag/qr?d=http%3A%2F%2F....",
      "transfer_payment_data":{
        "purpose":"TA 123456",
        "accountowner":"secupay AG",
        "iban":"DE00 0123 4567 8912 3456 78",
        "bic":"COBADEFF103",
        "bankname":"Test Bank"
      }
    }
  },
  "errors":null
}
```


3.5. Capture vorautorisierter Zahlungen

Vorautorisierte Zahlungen müssen, um die Zahlung durchzuführen, gecleared werden, dies kann auf 2 Wegen erfolgen, entweder per POST auf

```
https://api.secupay.ag/payment/<hash>/capture
```

oder per GET auf

```
https://api.secupay.ag/payment/<hash>/capture/<apikey>
```

Mittels der GET Variante erhält der Anwender im Browser eine Übersicht über die Transaktion und kann per Button die Zahlung durchführen.

In der POST Version muss wie gewohnt der API-Key im data array übermittelt werden.

3.6. Storno vorautorisierter/nicht final eingereichter Zahlungen

Auch beim Storno sind 2 Wege vorgesehen, einmal mit Interaktion des Anwenders per GET, sowie eine reine Server zu Server Übermittlung. Storniert werden können vorerst nur Transaktionen welche vorautorisiert sind oder noch nicht final eingereicht wurden (Bei Lastschriften ist dies in der Regel der nachfolgende Tag, etwa 7 Uhr). POST:

```
https://api.secupay.ag/payment/<hash>/cancel
```

GET:

```
https://api.secupay.ag/payment/<hash>/cancel/<apikey>
```

In der POST Variante wird bei Erfolg/Misserfolg entsprechend status failed/ok zurückgeliefert, bei einem Fehler inklusive entsprechender Fehlermeldung im error array.

Antwort

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "demo":0,
    "trans_id":"1831201"
  },
  "errors":null
}
```

3.7. addData (Versandinformationen übertragen)

Beim Versand haben Sie die Möglichkeit, an secupay Sendungsinformationen zu übermitteln. Bei Lieferungen, die aus mehreren Paketen besteht führen Sie bitte den addData-Aufruf für jede Sendung einzeln aus.

Beim **Kauf auf Rechnung** gilt:

- addData ersetzt nicht den **Capture-Aufruf**.
- Bei Einzelsendungen nur per Capture-Aufruf
- bei mehreren Sendungen die letzte Sendung per Capture-Aufruf übermitteln.

Hierzu muss auf folgende URL eine POST-Anfrage gesendet werden:

```
https://api.secupay.ag/payment/adddata/
```

Im Data-Objekt der Anfrage muss zusätzlich noch der Parameter `apikey` und `hash` der Transaktion übermittelt werden:

```
{
 "data":{
  "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
  "hash":"oxaijkeaucbl2041",
  "tracking":{
   "provider":"DHL",
   "number":"TC123456789"
    },
  "invoice_number":"0001"
 }
}
```

3.8. Invoice (Kauf auf Rechnung)

Bei der Zahlart invoice/Kauf auf Rechnung gibt es zusätzlich zu den normalen API-Aufrufen noch die Funktion, das Versanddatum zu setzen und Trackinginformationen zu übersenden. Dies ist erforderlich, um die Fälligkeit der Forderung zu markieren und die bei secupay nachgelagerten Prozess-Schritte anzustoßen.

3.8.1. Capture (Versanddatum der Rechnung setzen)

Um das Versanddatum einer Rechnung zu setzen, muss folgende URL aufgerufen werden:

```
https://api.secupay.ag/payment/{hash}/capture/{apikey}
```

Die Werte für den hash sowie des API.key des Händlers sind entsprechend zu setzen. Beispiel einer URL:

```
https://api.secupay.ag/payment/oxaijkeaucbl2041/capture/6801fxxxxxxxxxxxxxxx
xxxxxxxxxxxxxx7ace
```

Alternativ ist auch ein Setzen des Versanddatums per POST ohne eine direkte Nutzerinteraktion möglich, in dieser Variante wird der Zeitpunkt der Anfrage als Versanddatum verwendet. Hierzu muss auf folgende URL eine POST-Anfrage gesendet werden:

```
https://api.secupay.ag/payment/{hash}/capture/
```

Im Data-Objekt der Anfrage muss zusätzlich noch der Parameter `apikey` übermittelt werden:

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "tracking":{
      "provider":"DHL",
      "number":"TC123456789"
    },
    "invoice_number":"RN 0001"
  }
}
```

Die Parameter `tracking` (Versanddienstleister und Tracking-Nummer) und `invoice_number` (Rechnungsnummer) sind optional. Die hier angegebene `invoice_number` ist bei einer späteren Statusanfrage in der Antwort enthalten.

3.8.2. Zusätzliche Rückgabeparameter bei einer Statusanfrage

Bei einer Statusanfrage auf invoice-Transaktionen bekommen Sie zusätzliche Felder im `opt` Property zurückgeliefert:

Antwort:

```
{
  "status":"ok",
```

```
  "data":{
    "hash":"tujevzgobryk3303",
    "status":"accepted",
    "created":"2013-06-10 10:27:42",
    "demo":0,
    "trans_id":"1831201",
    "amount":100,
    "opt":{
      "recipient_legal":"secupay AG, Goethestraße 6, 01896 Pulsnitz",
      "payment_link":"https://api.secupay.ag/ payment/tujevzgobryk3303",
      "payment_qr_image_url":"https://api.secupay.ag/ qr?d=http
%3A%2F%2F....",
      "transfer_payment_data":{
        "purpose":"TA 123456 DT 20130610",
        "accountowner":"secupay AG",
        "iban":"DE00012345678912345678",
        "bic":"COBADEFF103",
        "accountnumer":"0123456789",
        "bankcode":"01234567",
        "bankname":"Test Bank"
      },
      "invoice_number":"RN 0001"
    }
  },
  "errors":null
}
```


3.8.3. Zahlung der Rechnung durch den Endkunden

Um die Rechnung zu begleichen hat der Kunde mehrere Möglichkeiten. Diese werden ihm in einem iFrame angeboten. Die URL hierfür ist dieselbe wie die zurückgegebene iFrame URL in init. Diese URL kann direkt per QR-Code oder per Link in eMails bzw Rechnungs-PDFs integriert werden, um dem Kunden eine einfache Möglichkeit zum Begleichen der Rechnung zu bieten.

3.9. Wiederkehrende Zahlungen (Abo)

Um wiederkehrende Zahlungen durchführen zu können müssen Sie zuerst Ihren Vertrag dafür freischalten lassen. Anschließend gibt es 2 Methoden um eine initiale ABO Zahlung durchzuführen. Die bevorzugte Möglichkeit ist das Anfragen auf /payment/getSubscription für eine bereits erfolgreiche Zahlung. Alternativ können Sie auch im init der 1. ABO Transaktion ein Subobjekt“subscription”übermitteln, woraufhin Sie in der Antwort den Parameter “subscription_id” zurück bekommen. Mit diesem werden die Folgetransaktionen ausgelöst. Wichtig ist hierbei nur, dass es das Objekt gibt, falls kein ABO Verwendungszweck gewünscht ist also ein leeres Subobjekt “subscription”.

Die Subscription-Id kann verwendet werden, um eine neue Zahlung unter Verwendung der vom Zahler eingegebenen Zahlungsmitteldaten zu erzeugen.

Die Interaktion zwischen Ihnen und secupay läuft hierbei ohne Interaktion durch den Endkunden ab.

Init Request:

```
{
  ...

  "data":{
```

```
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "subscription":{
      "purpose":"ABO Monatlich bei ..."
    },
    "amount":2350,
    "purpose":"Ihre Bestellung bei ..."
  }
}
```

3.9.1. Erstellen einer Subscription durch einen gegebenen Hash

Falls Sie nachträglich zu einer bereits getätigten Zahlung ein Abo erzeugen möchten, ist dies mitderactiongetSubscriptionmöglich.HierbeiwirdperPOSTauf/payment/getSubscription der zu verwendende Apikey sowie der Hash der Ursprungstransaktion übermittelt.

Als Antwort erhalten Sie daraufhin eine Abo-Id mit welcher nun Abo Zahlungen durchgeführt werden können. POST

```
https://api.secupay.ag/payment/getSubscription
```

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "hash":"tglswdmhziwb84171",
    "subscription":{
      "purpose":"ABO Monatlich bei ..."
    }
  }
}
```

Antwort

```
{
  "status":"ok",
  "data":{
    "subscription_id":1234
  },
  "errors":null
}
```

3.9.2. Verwendung der Subscription-Id

Verwendungszweck

Den Verwendungszweck von Abo Transaktionen können Sie an mehreren Stellen definieren, hierbei gilt folgende Rangabfolge:

1. purpose im POST auf payment/subscription
2. purpose im Unterobjekt “subscription” bei payment[init|getSubscription]

3. purpose der Ursprungstransaktion

Um das Rücklastschriftrisiko zu minimieren, achten Sie bitte auf einen sinnvollen Verwendungszweck den der Kunde eindeutig zuordnen kann.

3.9.3. Abo-Zahlung ausführen

Erforderliche EIngabeparameter:

```
Eingabe-Parameter:
in['amount']
in['apikey']
in['purpose']
in['url_push']
in['basket']
in['email']
```

Um aus der aus /payment/init zurückgelieferten Subscription-Id eine Abo Zahlung zu erzeugen, ist die nachfolgende POST Anfrage auf /payment/subscription auszuführen. POST

```
https://api.secupay.ag/payment/subscription
```

Im data Objekt der Anfrage müssen zusätzlich noch die Parameter“amount”,“subscription_id” sowie “apikey” übermittelt werden:

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "subscription_id":1234,
    "amount":2350,
    "purpose":"Ihre Bestellung bei ..."
  }
}
```

Antwort

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303"
  },
  "errors":null
}
```

3.9.4. Besonderheiten bei Wiederkehrenden Rechnungskauf Transaktionen

Bei einer wiederkehrenden Rechnungskauf Transaktion sollte dem Kunden bei Erstellung der neuen Zahlung eine Mitteilung gesendet werden mit der Angabe der Überweisungsdaten, welche als Antwort auf die Transaktionserstellung zurückgeliefert werden.

3.10. Rückstellung von Transaktionen

Transaktionen können zurückgestellt und damit die Auszahlung an die stakeholder einer Transaktion zu hemmen, z.B. bis die Leistungserbringung abgeschlossen.

```
https://api.secupay.ag/payment/init
```

Init siehe Abschnitt "Init"

```
{
  ...

  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "set_accrual":"1"
  }
}
```

Antwort gibt den Erfolg True bzw. nicht Erfolg false von accrual zurück

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303",
    "iframe_url":"https://api.secupay.ag/payment/tujevzgobryk3303",
    "accrual":true
  },
  "errors":null
}
```

Das Aufheben der Rückstellung erfolgt mittels POST über folgende URL POST

```
https://api.secupay.ag/payment/reverseaccrual
```

```
{
  data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "hash":"tujevzgobryk3303"
  }
}
```

3.11. Hinzufügen einer stakeholder Position zu einer bestehenden Transaktion

Für die Ausführung der Transaktion wird jeweils der letzte Betrag zu einer apikey/name-Kombination verwendet. Diese Funktion kann nur ausgeführt solange ”set_accrual”:”1” nicht durch ”reverseaccrual” aufgehoben wurde.

```
https://api.secupay.ag/payment/additem
```

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "hash":"tujevzgobryk3303",
    "basket":[
      {
        "item_type":"stakeholder_payment",
        "apikey":"37373xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx2fe2",
        "name":"plattform fee",
        "total":"1324"
      }
    ]
  }
}
```

Antwort

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303"
  },
  "errors":null
}
```

3.12. Nachbucher zu genehmigter Transaktion

Es wird ein Transaktion zu einer genehmigten Vorgänger-Transaktion unter Verwendung der vorhandenen Zahlungsmittel angelegt.

Der Zahlbetrag ist auf 20% des Ursprungsbetrags beschränkt. Je Transaktion ist nur eine Nachbucher-TA zulässig, für eine Nachbucher-TA kann keine untergeordnete Nachbucher-TA erstellt werden.

Die Funktion kann nur aufgerufen werden, wenn eine entsprechende Freischaltung für den jeweiligen Vertrag vorliegt.

```
https://api.secupay.ag/payment/initSubsequent
```

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "hash":"tglswdmhziwb84171",
    "amount":100,
    "basket":[
      {
        "item_type":"article",
        "article_number":"3211",
        "quantity":"2",
        "name":"Testname 1",
        "ean":"4123412341243",
```

```
        "tax":"19",
        "total":"1324",
        "price":"1000"
      }
    ]
  }
}
```

Antwort

```
{
  "status":"ok",
  "data":{
    "hash":"tujevzgobryk3303"
  },
  "errors":null
}
```

3.13. Refund eingereichter Transaktionen

Das Einreichen des Refund erfolgt per Server zu Server Übermittlung. Wenn der Zahlungseingang zur Transaktion aussteht, wird diese storniert bzw. der Betrag angepasst. Andernfalls wird eine neue Refund-Transaktion angelegt und die TransaktionsID wird zurückgegeben.

Die Funktion kann nur aufgerufen werden, wenn eine entsprechende Freischaltung für den jeweiliegen Vertrag vorliegt.

```
https://api.secupay.ag/payment/refund
```

```
{
  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "hash":"tglswdmhziwb84171",
    "amount":100
  }
}
```

Antwort bei Storno

```
{
  "status":"ok",
  "data":{
    "hash":"tglswdmhziwb84171",
    "action":"voided",
    "amount":100,
    "remaining_transaction_amount":0
  },
```

```
  "errors":null
}
```

Antwort bei Teilstorno

```
{
  "status":"ok",
  "data":{
    "hash":"tglswdmhziwb84171",
    "action":"voided",
    "amount":100,
    "remaining_transaction_amount":5000
  },
  "errors":null
}
```

Antwort bei Refund

```
{
  "status":"ok",
  "data":{
    "hash":"tglswdmhziwb84171",
    "action":"refund",
    "amount":100,
    "trans_id":"1831201",
    "purpose":"TA-Code 1831201(08.02.15)|Teilerstattung von TA 1821001"
  },
  "errors":null
}
```

• *remaining_transaction_amount* ist der neue Transaktionswert z.b. 0 bei vollständiger Stornierung

4. Push API

Aus Sicherheitsgründen bietet secupay die Möglichkeit jede über die API abgesetzte Transaktion zusätzlich über eine reine Server zu Server Kommunikation zu bestätigen.

Dabei wird beim Einreichen der Transaktion der Parameter url_push übermittelt. Auf diesen wird bei Statusänderungen an der Transaktionen der neue Status per HTTP POST übermittelt. Dies ermöglicht z.B. den Einsatz einer eigenen WaWi welche automatisch den aktuellen Status des Zahlvorgangs erhält und damit die Abarbeitung und den Versand der Pakete vereinfacht und sicherer gestaltet.

Hierbei wird bei Umstellung des Transaktionsstatus im secupay System eine HTTP(S) POST Anfrage auf die übermittelte Push url erstellt: Url:

```
https://push.example.net/push_client.php
```

```
POST /push_client.php HTTP/1.1

hash=jtnjpfgrbrqk3300&status_id=6&status_description=abgeschlossen&changed=13
65444092&payment_status=accepted&apikey=6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7
ace
```

- hash – den betroffenen Hash
- status_id – die detaillierte secupay Status-Id
- status_description – der detaillierte secupay Status-Text
- changed – UTC Unix Timestamp wann die besagte Statusänderung eingetreten ist
- apikey – der bei der Transaktion verwendete API-Key, dieser sollte vor Annahme des Push's abgeglichen werden
- payment_status – Vereinfachte Form des Secupay Status, dies vereinfacht die Einordnung in Kategorien, möglich sind hierbei
    - accepted - Transaktion wurde erfolgreich durchgeführt
    - authorized - Transaktion ist autorisiert
    - denied - Transaktion wurde abgelehnt
    - issue - Es gibt ein Problem mit der Transaktion, z.B. Rücklastschrift Fall
    - void - Die Transaktion wurde storniert oder gutgeschrieben
    - issue_resolved-DasProblemmitderTransaktionwurdegelöst,z.B.Rücklastschrift beglichen
    - refund - Dieser Betrag für die Transaktion wurde für die Rückabwicklung vorgemerkt und wird im nächsten Zyklus bearbeitet
- Die Push Notification muss von der empfangenden Seite nun noch quittiert werden. Dazu muss sie als einfachen Text mit dem kompletten Anfrage-Text inklusive ack=Approved antworten.

Für das oben genannte Beispiel erwartet die API dabei folgende Antwort:

```
ack=Approved&hash=jtnjpfgrbrqk3300&status_id=6&status_description=abgeschloss
en&changed=1365444092&payment_status=accepted&apikey=6801fxxxxxxxxxxxxxxxxxx
xxxxxxxxxxx7ace
```

Können Sie den Hash auf Ihrer Seite nicht zuordnen, sollten Sie für die Fehleranalyse und spätere Auswertung mit ack=Disapproved senden. Bei Bedarf kann hier auch noch der Rückgabeparameter error mit einem nützlichen Hinweistext ergänzt werden. Beispiel einer Abweisung eines Push Vorganges:

```
ack=Disapproved&error=no+matching+order+found+for+hash&hash=
jtnjpfgrbrqk3300&status_id=6&status_description=abgeschlossen&changed=1365444
092&payment_status=accepted&apikey=6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace
```

Aus Sicherheitsgründen empfehlen wir, alle Eingaben der PushAPI auf den Referrer api.secupay.ag bzw in der Testumgebung api-dist.secupay-ag.de zu prüfen. Beispiel PHP:

```
$ref = $_SERVER[
  'HTTP_REFERER'
];

if (strpos($ref, 'api.secupay.ag') !== FALSE){
  echo" Der Push ist von $ref";
}
```

Sie sollten die Anfragen der Push API nur per https (Port 443) zulassen.

5. Parameter

Name	Beispiel	Kommentar
apikey	6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace	
payment_type	debit	
demo	0	
url_success	http://shop.example.com/success.php	
url_failure	http://shop.example.com/failure.php	
url_push	http://shop.example.com/shop/push	
language	de_DE	
shop	APITest Shop	
shopversion	1.0	
moduleversion	1.0	
title	Herr	
firstname	Test FN	
lastname	Test LN	
company	Test Company	
industry	4812	
street	Test Street	
housenumber	5t	
zip	12345	
city	TestCity	
country	DE	
telephone	+4912342134123	

dob	01.02.1903	
email	test@ema.il	
ip	172.31.6.49	
amount	1199	
currency	EUR	
purpose	Test Order #1	
basket		siehe extra Definition Basket
userfields		siehe extra Definition Userfields
delivery_address		siehe extra Definition Lieferadresse
order_id	100203	
note	default note text	
apiversion	2.11	
labels		siehe extra Definition Labels
set_accrual	false	siehe extra Definition Rückstellung von Transaktionen
merchant_customer_id		siehe extra Definition Kundenummer

5.1. Besonderheiten bei einzelnen Parametern

• amount - hierbei handelt es sich jeweils um die kleinste Einheit der entsprechend gewählten Währung. Wird keine Währung explizit übermittelt, gehen wir von Eurocent aus.

5.2. Basket

5.2.1. Warenkorbstruktur

Beim Basket handelt es sich im Gegensatz zu den anderen Parametern um keine flache Struktur, sondern um ein Array aus Objekten. Dies ermöglicht die einfache Übermittlung von beliebigen Paramtern für die verschiedenen Positionen im Warenkorb.

Die korrekte Übermittlung des Warenkorbes hilft uns beim Support der Transaktionen sowie der Risikoabschätzung der Bestellung.

Nachfolgende Struktur wird von uns nativ aufgenommen und kann bei Bedarf erweitert werden.

```
[
  {
    "article_number":"1234",
    "item_type":"article",
    "name":"Testname1",
    "model":"test model",
```

```
      "ean":"2309842",
      "quantity":"2",
      "price":"50",
      "total":"100",
      "tax":"19"
    },
    {
      "article_number":"1235",
      "item_type":"article",
      "name":"Testname2",
      "model":"test model",
      "ean":"2309843",
      "quantity":"4",
      "price":"199",
      "total":"796",
      "tax":"19"
    },
    {
      "item_type":"shipping",
      "name":"standard shipping fee",
      "total":"500",
      "tax":"19"
    },
  ]
```

5.2.1.1. Item Types (Warenkorbpositionsarten)

EineweitereDifferenzierungderWarenkorbbestandteilekannüberdenParameteritem_type erreicht werden.

Folgende Arten von Warenkorbpositionen sind möglich:

- article - Waren-/Dienstleistungsposition
- shipping - Versandkosten
- donation - Spendenateil
- stakeholder_payment - Betrag für Dritte (setzt eine entsprechende Freischaltung für den jeweiligen Vertrag voraus)

Für die Art stakeholder_payment wird zusätzlich der Parameter apikey benötigt um den vorgesehenen Empfänger zu identifizieren und der Betrag im Parameter total übergeben.

```
[
  {
    "item_type":"stakeholder_payment",
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "name":"plattform fee",
    "total":"499"
  },
]
```

Für die Art transaction_payout wird zusätzlich der Parameter transaction_hash benötigt um den vorgesehenen Empfänger zu identifizieren, der Verwendungszweck wird im “name” Feld und der Betrag im Parameter total übergeben.

```
[
  {
    "item_type":"transaction_payout",
    "name":"Payout Purpose",
    "transaction_hash":"tujevzgobryk3303",
    "total":"1000"
  },
]
```

5.3. Userfields

Mit Hilfe der `userfields` ist es möglich zusätzliche Parameter zu übermitteln, welche im Nachgang die Filterung in unserem Backend erlauben (todo). Sinnvoll sind hier ggfs. Notizen, besondere Eigenschaften des Käufers oder der Bestellung.

```
{
  "userfield_1":"test 1",
  "userfield_2":"test 2",
  "userfield_3":"test 3"
}
```

5.4. Lieferadresse

Bei der `delivery_address` handelt es sich um ein Objekt das die Elemente der Lieferadresse aufnimmt.

Die korrekte Übermittlung der Lieferadresse hilft uns beim Support der Transaktionen sowie der Risikoabschätzung der Bestellung.

Nachfolgende Struktur wird von uns nativ aufgenommen und kann bei Bedarf erweitert werden.

```
{
  "firstname":"Test FN",
  "lastname":"Test LN",
  "company":"Test Company",
  "street":"Test Street",
  "housenumber":"5t",
  "zip":"12345",
  "city":"TestCity",
  "country":"DE"
}
```

5.5. Übergabe der Kundenummer des shop-Kunden

Zur Verringerung der Ablehnungsquote ist es günstig, die shop-Kunden für die bereits Erfahrungen vorliegen, eindeutig zuordnen zu können. Durch Übermittlung der Nutzernummer wird dies ermöglicht.

```
https://api.secupay.ag/payment/init
```

Init siehe Abschnitt "Init"

```
{
  ...

  "data":{
    "apikey":"6801fxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx7ace",
    "merchant_customer_id":"1"
  }
}
```

5.6. Experience / Bewertung

experience-Teilen Sie uns Ihre Zahlungserfahrungen mit dem Kunden mit (setzt eine entsprechende Freischaltung für den jeweiligen Vertrag voraus).

- positive - hier können Sie uns mitteilen wie oft der Kunde pünktlich bezahlt hat.

- 0 - keine positiven Erfahrungen mit dem Kunden
- 1 - eine positiv abgeschlossene Transaktion / Verkauf
- 2 oder aktueller Wert - min 2 positiv abgeschlossene Transaktionen / Verkäufe oder Anzahl dieser Transaktionen

negative - hier können Sie uns mitteilen, wenn ein Kunde Negativmerkmale hat

- 0 - keine negativen Erfahrungen mit dem Kunden
- 1 - Kunde hat aktuelle überfällige offene Posten
- 2 oder aktueller Wert - Kunde hat in der Vergangenheit RLS verursacht oder erst nach mehreren Mahnungen bezahlt

```
{
  "data":{
    ...

    "experience":{
      "positive":"1",
      "negative":"0"
    }
}
```

5.7. Eigene Labels / Beschriftung Warenkorb Titel, Senden und Abbrechen Button

Mit der Übergabe der Labels im init (siehe Punkt "Init") können Sie selbst die Beschriftung des Warenkorbtitels, der Senden und Abbrechen Buttons ändern.

```
{
  "data":{
    ...

  "labels":{
    "en_US":{
      "basket_title":"Your Order",
      "submit_button_title":"Submit",
      "cancel_button_title":"Return to Basket"
    },
```

```
      "de_DE":{
        "basket_title":"Ihre Bestellung",
        "submit_button_title":"Daten Übermitteln",
        "cancel_button_title":"Zum Warenkorb"
      }
    }
  }
```

5.8. industry / Branchencode

• industry - hier geben Sie den für Ihren Handel passenden MCC Code für Kreditkartenzahlungen an. Diese MCC Codes finden Sie als Liste unter https://github.com/greggles/mcc-codes

6. Fehlercodes

Die Fehlerausgaben bestehen immer aus einem code und einer dazugehörigen Message. Die Sprache der ausgegebenen Fehlermeldungen richtet sich nach dem in der Anfrage gesetzten language Parameter. Derzeit werden Englisch “en_us” (Standard) sowie Deutsch“de_de” unterstützt.

Nachfolgend eine Auflistung der möglichen Fehlermeldungen:

ID	Englische Bezeichnung	Deutsche Bezeichnung
0001	Invalid apikey	Ungültiger apikey
0002	Invalid hash	Ungültiger hash
0003	Cannot capture unauthorized payment	nicht autorisierte Zahlung kann nicht eingezogen werden
0004	Cannot cancel/void unaccepted payment	nicht abgeschlossene Zahlung kann nicht storniert werden
0005	Invalid amount	Ungüliger Betrag
0006	Cannot refund unaccepted payment	nicht abgeschlossene Zahlung kann nicht rückabgewickelt werden
0007	No payment type available	Keine Zahlart verfügbar
0008	Hash has already been processed	Hash wurde schon verarbeitet
0009	Scoring invalid	Scoring ungülig
0010	Payment currently not available	Diese Zahlungsart ist derzeit nicht verfügbar
0011	Payment couldnt be finalized	Zahlung konnte nicht abgeschlossen werden
0012	Selected payment type is not available	ausgewählte Zahlart ist nicht verfügbar
0013	Apikey mismatch	Apikey stimmt nicht überein
0014	Cannot capture specified payment	Zahlung konnte nicht abgeschlossen werden

0015	Cannot cancel/void specified payment	Zahlung konnte nicht storniert bzw erstattet werden
0016	Cannot refund specified payment	Zahlung konnte nicht erstattet werden
0017	Invalid Paymentdata	Zahlungsmitteldaten ungültig
0018	Missing Parameter	Fehlernder Parameter
0019	Couldnt create the user	Konnte den Nutzer nicht erstellen
0020	Username/email unavailable	Nutzername/E-Mail bereits vergeben
0021	Username or password invalid	Nutzername oder Passwort ungültig
0022	Userauth Token invalid	Userauth Token ungültig
0023	Could not connect the card to the user	Konnte die Karte nicht verknüpfen
0024	Invalid value for parameter	Ungültiger Wert für einen Parameter
0025	Cannot process specified payment	Angegebene Zahlung konnte nicht verarbeitet werden
0026	Cannot create payment information	Zahlungsinformationen konnten nicht erstellt werden
0027	Invalid data	Daten fehlerhaft
0028	Payment Provider declined the payment	Fehler beim einreichen der Transaktion beim Zahlungsanbieter.
0029	No transaction available for this terminal	Für dieses Terminal ist keine Transaktion hinterlegt.
0030	Cannot create transaction	Konnte die Transaktion nicht erstellen
0031	Unknown parking zone	Unbekannte Parkzone
0032	The password you entered is too short	Das eingegebene Passwort ist zu kurz
0033	The passwords are not equal	Die eingegebenen Passwörter sind nicht gültig
0034	The data provided is not sufficient	Übermittelte Daten nicht vollständig
0035	There was an error creating the Apikey	Fehler beim anlegen des APIKeys
0036	The Plate is not valid	Das Kennzeichen ist ungültig
0037	Payment data missing	Zahlungsmitteldaten fehlen
0038	Cannot find any active ticket for this zone and car	Es konnte kein aktives Ticket für diese Zone und dieses Fahrzeug gefunden werden
0039	There is already valid ticket for this car in this zone	Es existiert bereits ein Ticket für diese Zone und dieses Fahrzeug

0040	There is not any tariff valid for your parking time	Für die angegebene Parkzeit existiert kein gültiger Tarif
0041	Password recovery failed	Wiederherstellung des Passworts ist gescheitert.
0042	Your password has been recovered	Ihr Passwort wurde wieder hergestellt.
0043	Error recovering the password	Fehler beim wiederherstellen des Passworts
0044	Failed to save your payment data	Konnte die Zahlungsmitteldaten nicht speichern.
0045	Your payment data has been saved	Ihre Zahlungsmitteldaten wurden gespeichert.
0046	Referenced payment is not accepted	Referenzierte Zahlung ist nicht akzeptiert
0047	Cannot set accrual	Kann accrual nicht setzen
0048	Cannot reverse accrual	Kann accrual nicht entfernen
0049	Failed to add Data	Hinzufügen der Daten ist fehlgeschlagen
0050	Your bank doesn't support SEPA direct debit	Die angegebene Bank nimmt nicht am SEPA-Lastschriftverfahren teil.
0051	You are not approved for Subsequent.	Sie sind nicht für Nachbuchungen freigegeben.
0052	Subsequent posting already Available.	Nachbuchung bereits vorhanden.
0053	The is already a subsequent posting.	Dies ist bereits eine Nachbuchung.
0054	Amount too high.	Betrag zu hoch.
0055	You are not approved for Refund.	Sie sind nicht für Refund freigegeben.
0056	Missing or Invalid Parameter	Fehler: Pflichtparameter fehlt.
0057	Transaction not found	Fehler: Transaktion nicht gefunden.
0058	Refund amount higher than transaction amount	Fehler: Gutschriftsbetrag übersteigt Transaktionsbetrag.
0059	Invalid transaction status for refund	Fehler: unzulässiger Transaktionsstatus.
0060	Payment type doesn't support refund	Fehler: Produkt nicht unterstützt.
0061	Refund amount higher than remaining transaction amount	Fehler: Gutschriftsbetrag übersteigt verfügbaren Transaktionsbetrag.
0062	Refund creation failed	Fehler: Gutschrifterstellung fehlgeschlagen.
0063	Refund not implemented for payment type	Fehler: für Paymenttype nicht implementiert.
0064	You are not approved for getSubscription.	Sie sind nicht für die Funktion getSubscription freigegeben.
0065	Accrual not set for transaction.	Accrual ist für die Transaktion nicht gesetzt.
0066	Selected currency is not available.	ausgewählte Währung ist nicht verfügbar.

0067	This currency is not available for you.	Sie sind nicht für diese Währung freigegeben.
0068	You have not allowed characters transferred.	Sie haben nicht zulässige Zeichen übertragen.
0069	Refund amount does not match the available transaction amount.	Fehler: Gutschriftsbetrag entspricht nicht dem verfügbaren Transaktionsbetrag.
0070	Total amount does not match the basket amount.	Der Gesamtbetrag entspricht nicht der Summe des Basket.
0071	Basket field name is not valid.	Feld Name ist nicht gestezt.
0072	Payment type of transaction_payout transaction does not support Payout.	Zahlungsart der Transaktion_Payout-Transaktion unterstützt keine Auszahlung.
0073	Invalid transaction status of transaction_payout transaction.	Ungültiger Transaktionsstatus der Transaktion_Payout-Transaktion.
0074	Access denied to transaction_payout transaction.	Zugriff verweigert auf Transaktion_Payout-Transaktion.
0075	Basket has to many items.	Warenkorb enthält zu viele Positionen.
0076	PIN Wrong.	Code falsch.
0077	The pin was entered incorrectly too often.	Der Code wurde zu oft falsch eingegeben.
0078	You have entered an incorrect telephone number (allowed characters "+" and 0-9)	Sie haben eine falsche Telefonnummer eingegeben (erlaubte Zeichen "+" und 0-9)
0079	Operation not approved for this project Apikey.	Aktion nicht zulässig für diesen project Apikey.
0080	You are not approved for this Operation.	Sie sind nicht für diese Aktion freigegeben.
0081	Due to significant risk of fraud this user is rejected.	Aufgrund erheblicher Betrugsgefahr wird dieser Nutzer abgelehnt.
0082	The payment is already completed	Die Zahlung ist bereits abgeschlossen