Technik-Handbuch

Impressum

1&1 Internet AG
Elgendorfer Straße 57
56410 Montabaur
Deutschland

www.1und1.de
info@1und1.de

1&1 Internet AG
Parkring 10
1010 Wien
Österreich

Stand: März 2010

Copyright

Das Handbuch nutzen

Das Handbuch verwendet kleine Bilder (sog. „Icons“) als Verständnis- und Orientierungshilfen. Sie werden folgende Icons finden:

Achtung! Bei diesem Hinweis gibt es stets etwas zu beachten. Lesen Sie solche Hinweise besonders aufmerksam durch.

Erklärung. Vertiefendes Wissen zu einem bestimmten Bereich oder Begriff finden Sie überall, wo dieses Bild auftaucht.

Tipp, Hinweis. Schauen Sie sich unsere Tipps und Hinweise an. Hinter diesem Bildchen verbergen sich Informationen, die Ihnen die Arbeit mit ipayment erleichtern werden.

Profi-Information.

Inhaltsverzeichnis

Das Zahlungssystem ipayment

Das ist ipayment

ipayment unterstützt Ihren Geschäftserfolg im Internet, indem es Ihren Kunden die sichere Zahlung per Kreditkarte, internetbasiertem elektronischen Lastschriftverfahren und weiteren Zahlungsmedien ermöglicht. Dabei sorgt ipayment für die sichere Übertragung der von Ihren Kunden eingegebenen Zahlungsdaten zu Ihrem Zahlungsanbieter, der die Verbuchung der Zahlungen sicherstellt. Zusätzlich sind weitere Aktionen möglich, wie zum Beispiel verzögerte Abbuchungen, Stornierungen und Gutschriften.

Bei einigen Shop-Systemen bereits integriert

Die von der 1&1 Internet AG angebotenen Shop-Systeme sind bereits an ipayment angebunden. Wenn Sie ein solches Shop-System nutzen, ist diese Dokumentation nicht für Sie relevant. Auch bei anderen Shop-Systemen ist ipayment eventuell bereits integriert. Unter http://www.1und1.info/xml/order/WpIpaymentShopSysteme finden Sie eine Übersicht.

Wenn Sie sich nicht sicher sind, ob ipayment in Ihrem Shop-System integriert ist, wenden Sie sich am besten an den Support des entsprechenden Shop-Systems.

Diese Dokumentation enthält Informationen zur technischen Ansteuerung der ipayment-Schnittstellen für eine individuelle Integration von ipayment in Ihre Webanwendung. Viele der beschriebenen ipayment-Funktionen stehen Ihnen unabhängig von der Art der Nutzung zusätzlich im ipayment-Konfigurationsmenü zur Verfügung. Sie erreichen das ipayment-Konfigurationsmenü unter https://konfig.ipayment.de.

Für die Nutzung von ipayment benötigen Sie zusätzlich einen Akzeptanzvertrag für die von Ihnen gewünschten Zahlungsmedien mit einem Zahlungsanbieter Ihrer Wahl. Eine Übersicht zu den von ipayment unterstützten Zahlungsanbietern finden Sie unter https://ipayment.de > **Zahlungsanbieter**.

Weiterführende Informationen zu ipayment

Vollständige Informationen zum Umfang der Funktionalitäten des ipayment-Systems finden Sie auf unserer Webseite unter https://ipayment.de.

Hinweise zu den Zugangsdaten und zur Einrichtung Ihres ipayment-Accounts lesen Sie in unserer FAQ unter http://faq.ipayment.de.

Die letzten Änderungen am ipayment-System haben wir in unserem Changelog zusammengefasst: https://ipayment.de/technik/changelog.php4

Das finden Sie in diesem Dokument

Funktionen zur Zahlungsabwicklung

In dieser Dokumentation werden alle Parameter aufgelistet, die zur Abwicklung von Zahlungen, sicheren Speicherung der Zahlungsdaten und für weitere Funktionen eingesetzt werden können. Zusätzlich finden Sie Erläuterungen und empfohlene Vorgehensweisen zur Integration von ipayment. Sie können ipayment einfach in Ihre Website, Ihr Online-Shop-System oder in Ihre Backend-Prozesse integrieren.

Lesen Sie unbedingt das Kapitel *Sichere Integration von ipayment* ab Seite 67, damit Sie ipayment sicher in Ihrer Anwendung integrieren können.

Funktionen zur Adressprüfung

Das Zahlungssystem ipayment bietet einen zusätzlichen Adressprüfungs-Service. Durch diesen Service können Sie Adressen automatisch auf postalische Korrektheit überprüfen lassen. Schreibfehler werden dabei erkannt und korrigiert.

Der Service ist derzeit nur für Adressen aus Deutschland verfügbar. Er basiert auf den Postdaten, die von der Deutschen Post AG geliefert werden. Der Service prüft nur, ob Straße, Ort und Postleitzahl zusammenpassen. Es kann nicht geprüft werden, ob an der angegebenen Adresse tatsächlich jemand wohnt.

Für die Nutzung des Adressprüfungs-Services fallen bei erfolgreicher Adressprüfung Kosten an. Die Höhe dieser Kosten können Sie der Website https:/ipayment.de entnehmen. Den Adressprüfungs-Service können Sie auch unabhängig von der Abwicklung von Zahlungen mit ipayment einsetzen.

Die ipayment-Adressprüfung können Sie über zwei Schnittstellen in Ihr System integrieren: CGI und SOAP-Webservice. Weiterführende Informationen zu den möglichen Parametern erhalten Sie im Kapitel *Parameter für die Adressprüfung* ab Seite 59.

ipayment-Funktionen testen (Simulationsmodus)

Sie können einen ipayment-Account in den Simulationsmodus schalten. Das heißt, der Account nimmt Transaktionen entgegen, simuliert die Abwicklung und liefert die Ergebnisparameter zurück. Die Zahlungsdaten werden dabei nicht an die Zahlungsanbieter übermittelt, es werden keine echten Zahlungen abgewickelt. Die Zahlungsdaten werden jedoch vom ipayment-System auf Plausibilität geprüft. Bei Kreditkartendaten bedeutet das zum Beispiel, dass das System überprüft, ob das Verfalldatum in der Zukunft liegt oder ob die Kartenprüfnummer die korrekte Länge hat. Ob die Daten wirklich korrekt sind, kann nur die Bank überprüfen. Diese Überprüfung findet im Simulationsmodus nicht statt.

Um einen bestehenden ipayment-Account in den Simulationsmodus zu schalten, wechseln Sie in Ihr ipayment-Konfigurationsmenü und klicken auf **Anwendungen**. Klicken Sie hinter der gewünschten Anwendung auf **Konfigurieren** und wählen Sie **Details ändern**. Aktivieren Sie dort den Simulationsmodus. Beachten Sie, dass Sie diese Anwendung nicht für „echte“ Transaktionen verwenden können, solange der Simulationsmodus aktiv ist.

Wenn Sie noch keinen ipayment-Account besitzen, können Sie folgende Parameter verwenden, um die Funktionalität von ipayment zu testen:

Standard-Testaccount

Account-ID:	`99999`
Anwendungs-ID:	`99999`
Anwendungspasswort:	`0`
Adminaktionspasswort:	`5cfgRT34xsdedtFLdfHxj7tfwx24fe`

Testaccount mit Security-Key

Account-ID:	`99999`
Anwendungs-ID:	`99998`
Anwendungspasswort:	`0`
Adminaktionspasswort:	`5cfgRT34xsdedtFLdfHxj7tfwx24fe`
Security-Key:	`testtest`

Mit diesen Account-Daten können Sie die in diesem Dokument beschriebenen Schnittstellen testen. Die Nutzung des ipayment-Konfigurationsmenüs ist mit diesen Daten nicht möglich.

Ihre Test-Transaktionen können nicht von anderen Benutzern eingesehen werden.

Sicherheit für Zahlungen

Was ist PCI DSS?

PCI DSS ist die Abkürzung für **Payment Card Industry Data Security Standard.** Dieses Regelwerk wurde von Visa und MasterCard ins Leben gerufen. Es dient der sicheren Abwicklung von Kreditkartentransaktionen. Die Beachtung des PCI DSS wird von allen wichtigen Kreditkartenorganisationen gefordert. Die Anforderungen werden vom PCI Security Standards Council (www.pcisecuritystandards.org) gepflegt und regelmäßig den aktuellen Sicherheitsanforderungen angepasst.

Die Kreditkartenorganisationen verlangen von jedem, der Kreditkartendaten verarbeitet oder speichert, eine Zertifizierung über die Einhaltung des PCI Regelwerks. Das kann zum Beispiel ein Händler, Dienstleister oder Payment Service Provider sein. Jeder Kontakt mit Kreditkartendaten, zum Beispiel in einem Shopsystem, zieht die Pflicht zur PCI-Zertifizierung nach sich. Eine Verarbeitung liegt zum Beispiel dann vor, wenn ein Formular mit Kreditkartendaten an den Shop-Server gesendet und dort weiterverarbeitet wird. Bei Nichtbefolgung können Strafgebühren verhängt, Einschränkungen ausgesprochen oder die Akzeptanz von Kreditkarten untersagt werden. Ein Händler ist verpflichtet, ausschließlich mit Partnern zusammenzuarbeiten, die ebenfalls PCI-zertifiziert sind. Durch die regelmäßige Aktualisierung des PCI-Regelwerkes ist sichergestellt, dass auch neue Gefahren berücksichtigt werden. Dadurch ist auch in Zukunft die Sicherheit der Zahlungsdaten bei allen Unternehmen gewährleistet.

Bei einer PCI-Zertifizierung werden unter anderem geprüft:

- Die Sicherheitsanforderungen der Rechner, die mit der Verarbeitung und Speicherung der Kreditkarten zu tun haben.
- Die verwendete Infrastruktur, zum Beispiel die Installation einer Firewall zum Schutz von Angriffen, Anforderungen an Webserver und Scripte und der Einsatz von Virenschutzprogrammen.
- Sicherheit, Abläufe und verwendete Standards bei der Entwicklung der Web-Anwendung.
- Die internen Abläufe, Prozesse und Sicherheitspolitik des Unternehmens.

Die Zertifizierung muss, abhängig vom Umsatzvolumen, jährlich wiederholt werden. Durchgeführt wird die Zertifizierung durch einen berechtigten Zertifizierer. Je nach Zertifizierungslevel beinhaltet eine Zertifizierung zum Beispiel das Ausfüllen von umfangreichen Fragebögen, regelmäßige Server-Scans und vor-Ort-Audits des Unternehmens. Die dabei anfallenden Kosten müssen vom zertifizierten Unternehmen getragen werden.

ipayment und PCI DSS

ipayment ist seit 2004 auf dem höchsten Anforderungslevel nach PCI DSS zertifiziert. Diese Zertifizierung wird jährlich wiederholt, so dass die Einhaltung des PCI-Sicherheitsstandards gewährleistet ist.

Sie arbeiten bereits PCI-konform, wenn Sie ein Shopsystem einsetzen, das auf eine ipayment-Schnittstelle zugreift, bei der die vollständige Verarbeitung der Kreditkartendaten durch ipayment stattfindet. Ein solches Shop-System ist zum Beispiel der 1&1 E-Shop, Oxid eSales oder

osCommerce und xt:Commerce (aktuelle Versionen von 2008). Wenn dies zutrifft, müssen Sie sich keine Gedanken zur regelmäßigen und teueren Zertifizierung machen. Das Gleiche gilt für eine individuelle Integration von ipayment im **normalen Modus** oder im **Silent-Modus**, bei denen der Shop keinen direkten Kontakt mit den Zahlungsdaten hat. Die Zahlungsdaten werden bei ipayment unter Einhaltung der PCI-Spezifikationen sicher verarbeitet. Gleiches gilt auch, wenn der **Gateway-Modus** oder der **Webservice** nur mit Transaktionstypen verwendet wird, bei denen keine Zahlungsdaten an ipayment gesendet werden, wie zum Bespiel Abbuchungen, Stornos, oder ähnliches.

Sie sind verpflichtet, die kostenpflichtige Zertifizierung durchführen zu lassen, wenn Sie eine andere Integrationsmethode verwenden. Ebenfalls verpflichtet sind Sie, wenn Sie mit einem Shop-System arbeiten, das die Kreditkartendaten selbst verarbeitet, bevor diese an ipayment versendet werden.

Weiterführende Informationen

Informationen zu einer eigenen PCI-Zertifizierung erhalten Sie von Ihrem Zahlungsanbieter bzw. Acquirer sowie unter www.pcisecuritystandards.org.

Transaktionstypen

Es gibt verschiedene Transaktionstypen, die für die Zahlungsabwicklung nötig sind. Diese Transaktionstypen entsprechen zum Beispiel Zahlungsvorgängen, die über Online-Shops ausgeführt werden. Welche Zahlungsdaten und weitere Informationen benötigt werden, hängt von der Art der Transaktion ab. Zahlungsdaten sind zum Beispiel Kreditkartendaten oder Bankdaten, weitere Informationen können zum Beispiel Referenzen auf zuvor abgewickelte Transaktionen sein.

Abwicklung von Zahlungen

Die beiden nachfolgenden Transaktionstypen (**preauth** und **auth**) werden bei der Abwicklung von Zahlungen am häufigsten eingesetzt.

Verzögerte Zahlungsabwicklung/Vorautorisierung (preauth)

Beispiel:

Sie möchten die Beträge erst dann abbuchen, wenn die Ware verschickt wird (Fernabsatzgesetz).

Bei der Abwicklung von Kreditkarten oder Prepaid-Karten wird bei diesem Transaktionstyp der Betrag auf der Karte reserviert und somit für die Zahlung festgehalten. Dabei wird auch geprüft, ob die Karte existiert und aktuell genügend Guthaben aufweist. Nicht geprüft wird der Name des Karteninhabers. Die Reservierung ist bei Kreditkarten meistens maximal 28 Tage gültig, die genaue Gültigkeitsdauer ist vom Zahlungsanbieter abhängig. Idealerweise sollten die Beträge innerhalb von 5 bis 7 Tagen nach der Reservierung abgebucht werden.

Trotz der Tatsache, dass die Reservierung meistens nur 28 Tage gültig ist, können alle Abbuchungen auch später ausgeführt werden. Zu näheren Informationen zu den Abbuchungen lesen Sie bitte die Hinweise beim Transaktionstyp **capture**: *Abbuchen von vorautorisierten Transaktionen (capture)* (Seite 14).

Bei der Abwicklung von Zahlungen per ELV (Elektronisches Lastschriftverfahren) wird bei einer Vorautorisierung nur die Plausibilität der Kontodaten geprüft. Es wird also geprüft, ob das Konto bei der angegebenen Bank existieren könnte. Ob es tatsächlich existiert und genügend Guthaben aufweist, kann erst durch das Ausführen einer echten Buchung festgestellt werden. Zusätzlich werden alle Blacklist- und Scoring-Prüfungen der Zahlungsanbieter ausgeführt.

Sofortige Buchung einer Zahlung (auth)

Beispiel:

Sie möchten die Zahlung sofort ausführen.

Bei diesem Transaktionstyp finden Vorautorisierung und Verbuchung des Geldbetrags in einem Schritt statt. Dieser Transaktionstyp wird immer dann eingesetzt, wenn kein spezieller Transaktionstyp ausgewählt wurde. Es ist die einfachste Lösung zur Abwicklung von Zahlungen, weil dabei keine weiteren Aktionen erforderlich sind.

Zahlungsdaten überprüfen

Die folgenden Transaktionstypen dienen zur Überprüfung von Zahlungsdaten.

Plausibilitätsprüfung der Zahlungsdaten (base_check)

Beispiel:

Sie möchten die Zahlungsdaten allgemein auf Gültigkeit überprüfen.

Dieser Transaktionstyp führt eine Plausibilitätsprüfung der angegebenen Zahlungsdaten durch, ohne die Daten an einen Zahlungsanbieter zu senden. Bei Kreditkarten und Kontodaten wird geprüft, ob die Kreditkarte oder das Bankkonto existieren könnte. Eine erfolgreiche Prüfung garantiert jedoch nicht, dass die Kreditkarte oder das Konto wirklich existiert oder ausreichend Guthaben aufweist.

Die Daten werden bei diesem Transaktionstyp nicht in der Transaktionsdatenbank von ipayment gespeichert. Wenn Sie **base_check** gemeinsam mit dem ipayment-Storage-Service einsetzen, werden die Zahlungsdaten im Storage gespeichert. Mehr Informationen zum Storage-Service lesen Sie unter *Storage-Service zum Speichern von Zahlungsdaten* auf Seite 50.

Für Anfragen mittels **base_check** werden keine Transaktionsgebühren berechnet.

Detaillierte Prüfung der Zahlungsdaten (check_save)

Beispiel:

Sie möchten die Zahlungsdaten detailliert auf Gültigkeit überprüfen.

Bei diesem Transaktionstyp werden die Zahlungsdaten an den Zahlungsanbieter gesendet. Gleichzeitig versucht das System, eine Vorautorisierung (**preauth**) durchzuführen. Nach erfolgreicher Vorautorisierung wird die Reservierung sofort wieder storniert.

Bei Kreditkarten kann es vorkommen, dass Banken die Stornierung des reservierten Betrags nicht unterstützen. Das hat zur Folge, dass der Betrag auf der Karte reserviert bleibt, bis die maximale Reservierungsdauer (meist ca. 7 bis 28 Tage) erreicht wird. Wir empfehlen deshalb, **check_save** nur mit kleinen Beiträgen (zwischen 2 und 10 Euro) einzusetzen.

Bei Zahlungen per ELV ist eine direkte Prüfung der Korrektheit der Daten nicht möglich. Es können nur eine Plausibilitätsprüfung sowie die zusätzlichen Blacklist- und Scoring-Prüfungen der Zahlungsanbieter stattfinden. Eine direkte Prüfung ist aufgrund der Abläufe bei elektronischen Lastschriftverfahren nur bei einer echten Buchung auf ein echtes Bankkonto möglich.

Aktionen für durchgeführte Zahlungen

Die nachfolgenden Transaktionstypen können Sie für bereits ausgeführte Zahlungen (Transaktionen) einsetzen. Für diese Transaktionstypen werden keine Zahlungsdaten benötigt. Notwendig ist nur ein zusätzlicher Parameter, der die Transaktionsnummer der ursprünglich ausgeführten Zahlung enthält.

Abbuchen von vorautorisierten Transaktionen (capture)

Beispiel:

Die Ware steht zur Auslieferung bereit. Sie möchten den reservierten Betrag jetzt abbuchen.

Den Transaktionstyp **capture** setzen Sie ein, um einen vorautorisierten Betrag abzubuchen. Dabei können Sie unterscheiden, ob Sie den Betrag komplett abbuchen oder aufteilen möchten (zum Beispiel für Teillieferungen).

- **Komplettabbuchung:** Der reservierte Betrag wird vollständig abgebucht. Wenn diese Komplettabbuchung innerhalb des maximalen Reservierungszeitraums (meistens 28 Tage) erfolgt, ist die Abbuchung sofort erfolgreich. Bei späteren Abbuchungen muss das System zunächst eine Bankanfrage für eine erneute Reservierung des Betrags stellen. Hierbei kann es passieren, dass die Transaktion abgelehnt wird, zum Beispiel weil die Karte inzwischen gesperrt ist oder weil der maximale Verfügbarkeitsrahmen erreicht wurde.
- **Teilabbuchungen:** Für jeden Teilbetrag wird eine erneute Bankanfrage zur Reservierung des Teilbetrags gestellt – unabhängig davon, ob der maximale Reservierungszeitraum bereits erreicht wurde oder nicht. Dadurch kann es passieren, dass die Transaktion abgelehnt wird, zum Beispiel weil die Karte inzwischen gesperrt ist. Die Summe der einzelnen Teilabbuchungen darf nicht höher sein als 115% des Betrags, der ursprünglich vorautorisiert wurde. Die ursprüngliche Reservierung wird bei der ersten Teilabbuchung storniert.

Abbuchung so schnell wie möglich durchführen

Je länger die Zeitspanne zwischen Vorautorisierung und Abbuchung ist, umso höher ist die Gefahr, dass Teilabbuchungen nicht ausgeführt werden können. Führen Sie die Abbuchungen deshalb immer vor dem Versand der Ware durch.

Die Regularien einiger Kreditkartenorganisationen besagen, dass eine Komplettabbuchung innerhalb von 6 Tagen nach der Reservierung erfolgen soll. Danach haben die Banken grundsätzlich das Recht, eine Buchung wegen zu später Übertragung („Late Presentment“) als Chargeback abzulehnen. Falls eine Komplettabbuchung für Ihr Geschäft nicht so schnell möglich ist, sollten Sie sich mit Ihrem Zahlungsanbieter in Verbindung setzen.

Stornieren einer Vorautorisierung (reverse)

Beispiel:

Sie möchten den reservierten Betrag wieder freigeben oder eine Abbuchung am selben Tag wieder stornieren.

Sie können eine vorautorisierte Transaktion stornieren, solange noch keine Abbuchung stattgefunden hat. Bei manchen Zahlungsanbietern können Sie die Stornierung per **reverse** auch noch am Tag einer Abbuchung vornehmen.

Eine Stornierung führt dazu, dass der reservierte Betrag nicht abgebucht wird. Diese Stornierung funktioniert nur bei Komplettbeträgen. Bei Teilbeträgen oder wenn die Abbuchung bereits durchgeführt wurde, können Sie den Transaktionstyp **refund_cap** verwenden.

Rückbuchung eines Betrages (refund_cap)

Beispiel:

Sie möchten den Betrag stornieren, nachdem die Zahlung schon ausgeführt wurde.

Mit **refund_cap** können Sie bewirken, dass ein bereits abgebuchter Betrag zurückgebucht wird. Sie können zu einer Abbuchung mehrere Rückbuchungen durchführen. Die Summe der einzelnen Teil-Gutschriften darf jedoch bei Kreditkarten 115% des ursprünglichen Abbuchungsbetrags nicht überschreiten.

Auf der Kreditkartenabrechnung des Käufers wird sowohl die Abbuchung, als auch die Gutschrift vermerkt.

Für Zahlungen per ELV können Sie **refund_cap** nur eingeschränkt oder gar nicht nutzen. Die mögliche Vorgehensweise hängt vom Zahlungsanbieter ab.

Erneut Zahlungen abwickeln

Sie können auch weitere Zahlungen über bereits mit ipayment benutzten Kreditkarten ausführen, ohne dass Sie die Zahlungsdaten noch einmal eingeben müssen. ipayment verwendet dafür die Daten, die durch die vorherige Transaktion bereits im System gespeichert sind. Die Transaktionsnummer dieser Transaktion wird anstelle der Zahlungsdaten in einem zusätzlichen Parameter übergeben. Diese Daten bleiben mindestens für drei Monate gespeichert. Die Speicherzeit kann auf bis zu 12 Monate verlängert werden.

Erneute Vorautorisierung (re_preauth)

Beispiel:

Es ist eine weitere Bestellung des Kunden eingegangen, die nun reserviert werden soll.

Sie können eine erneute Vorautorisierung mit **re_preauth** vornehmen. Dabei verwendet ipayment die noch gespeicherten Zahlungsdaten einer früher abgewickelten Transaktion. Lesen Sie auch den Abschnitt *Verzögerte Zahlungsabwicklung/Vorautorisierung (preauth)* auf Seite 13.

Erneute sofortige Buchung (re_auth)

Beispiel:

Es ist eine weitere Bestellung des Kunden eingegangen, die nun belastet werden soll.

Über **re_auth** können Sie erneut eine Vorautorisierung vornehmen und gleichzeitig den Betrag abbuchen. ipayment verwendet dabei die noch gespeicherten Zahlungsdaten einer früher abgewickelten Transaktion. Lesen Sie auch den Abschnitt *Sofortige Buchung einer Zahlung (auth)* auf Seite 13.

Freie Gutschriften

Sie können über ipayment sogenannte „freie Gutschriften“ durchführen. Das bedeutet, dass Sie einer Kreditkarte einen Betrag gutschreiben können, ohne dass zuvor eine Abbuchung von dieser Kreditkarte stattgefunden hat. Verwenden Sie hierfür den Transaktionstypen **grefund_cap**.

Freie Gutschriften ausführen (grefund_cap)

Beispiel:

Sie möchten eine Gutschrift für eine Kreditkarte durchführen, mit der bisher noch keine Transaktion mit ipayment durchgeführt wurde.

Den Regelwerken der meisten Kreditkarten-Marken nach ist es verboten, eine Gutschrift durchzuführen, wenn zuvor kein Geld von dieser Kreditkarte abgebucht wurde. Aus diesem Grund müssen Sie die Nutzung dieses Transaktionstyps zuerst vom ipayment-Support freischal-

ten lassen. Bitte wenden Sie sich in diesem Fall mit Begründung für den Freischaltungswunsch per E-Mail an support@ipayment.de.

Führen Sie nach vorheriger Freischaltung den Transaktionstyp **grefund_cap** aus. Die Gutschrift wird sofort verbucht. Bei Nutzung dieses Transaktionstyps müssen wie bei normalen Zahlungen die vollständigen Zahlungsdaten angegeben werden. Dieser Transaktionstyp ist nur für Kreditkartenzahlungen verfügbar.

Der Transaktionstyp **grefund_cap** sollte nicht in einem Online-Shop-Umfeld sondern höchstens von Backend-Systemen verwendet werden.

Telefonisch autorisierte Zahlungen

Es kann passieren, dass eine Abbuchung oder Gutschrift bei Zahlungen mit Kreditkarten abgelehnt wird. Sie erkennen anhand des Fehlercodes, ob alternativ eine telefonische Autorisierung möglich ist. Das kommt zum Beispiel vor, wenn es sich bei der Transaktion um sehr hohe Beiträge handelt. Rufen Sie in diesem Fall den Genehmigungsdienst Ihres Zahlungsanbieters an und führen Sie eine telefonisch autorisierte Zahlung durch. Halten Sie am Telefon die Kreditkartendaten und den Betrag parat. Sie erhalten eine Autorisierungsnummer, mit der Sie dann die Zahlungen über ipayment abbuchen können.

Im E-Commerce-Umfeld ist eine telefonische Autorisierung meist unpraktikabel, weil hier der Kunde normalerweise nicht direkt mit dem Händler Kontakt aufnimmt. Dieser direkte Kontakt ist jedoch für die telefonische Autorisierung notwendig.

Abbuchen einer telefonisch autorisierten Zahlung (voice_auth)

Beispiel:

Sie haben die telefonische Genehmigung eingeholt und möchten den Betrag nun abbuchen.

Nach der Genehmigung durch den Genehmigungsdienst können Sie den Transaktionstyp **voice_auth** einsetzen. Neben den Kreditkartendaten benötigen Sie als zusätzlichen Parameter die Autorisierungsnummer, die Ihnen am Telefon genannt wurde. Der Betrag wird sofort abgebucht.

Abbuchen einer telefonisch autorisierten Gutschrift (voice_grefund_cap)

Beispiel:

Sie möchten eine Gutschrift für eine Kreditkarte durchführen, nachdem Sie die telefonische Genehmigung eingeholt haben.

Damit Sie diesen Transaktionstyp nutzen können, müssen Sie **voice_grefund_cap** zunächst vom Support freischalten lassen. Kontaktieren Sie dazu support@ipayment.de und nennen Sie in der E-Mail die Begründung für die gewünschte Freischaltung.

Nach der Freischaltung können Sie **voice_grefund_cap** einsetzen. Neben den Kreditkartendaten benötigen Sie als zusätzlichen Parameter die Autorisierungsnummer, die Ihnen der Genehmigungsdienst am Telefon genannt hat. Die Gutschrift wird sofort verbucht.

Schnittstellen des ipayment-Systems

Nachfolgend sehen Sie eine Übersicht aller Schnittstellen von ipayment. Diese Schnittstellen werden anschließend genauer beschrieben. Die empfohlenen Integrationsmethoden für die jeweiligen Funktionen sind grau hervorgehoben.

Funktion/Aktion	Normaler CGI-Modus	Silent CGI-Modus	Gateway CGI-Modus	SOAP-Web-service
Ausführung von Zahlungen/Integration in Online-Shops oder auf Webseiten				
Keine PCI-DSS-Zertifizierung nötig	X	**X**		
3-D-Secure-Sicherheitsverfahren sofort einsetzbar	X	**X**		
Ausführen von sofortigen Zahlungen	X	**X**	X	X
Ausführen von Zahlungen mit späterer Abbuchung	X	**X**	X	X
Herkunftsland der Zahlungsdaten und der IP	X	**X**	X	X
Zahlungsdaten in Storage-Service speichern	X	**X**	X	X
Gesicherte Rückmeldung für erfolgreiche Transaktionen	X	**X**	X*	X*
Durchführung umfangreicher Betrugsprüfungen	X	**X**	X	X
Versteckte Integration ohne Web-Ausgabe von ipayment		**X**	X	X
Ausführen von Aktionen für Zahlungen in Backend-Systemen				
Abbuchen von Zahlungen		X	**X**	**X**
Stornieren von Zahlungen		X	**X**	**X**
Ausführen von nochmaligen Zahlungen		X	**X**	**X**
Nahtlose Integration in Backend-Systeme			**X**	**X**
Weitere ipayment-Funktionen				
Durchführung einzelner Adressprüfungen			X	**X**
Prüfung von E-Mail-Adressen				**X**

* Durch direkte Kommunikationsantwort bei korrekten Timeout-Einstellungen sichergestellt.

Integration per CGI

Die Integration per CGI basiert auf einem CGI-Aufruf eines Web-Scriptes auf dem ipayment-Server per HTTP-GET oder HTTP-POST. Alle Werte werden als „Key-Value-Paare" übergeben und mit einem &-Zeichen getrennt. Einige Zeichen haben in einer URL eine besondere Bedeutung, wie zum Beispiel ?, %, & und das Leerzeichen. Diese und andere Sonderzeichen müssen URL-kodiert (escaped) werden, damit sie korrekt übertragen werden. Ein ? wird zu %3F, ein % zu %25, ein & zu %26, das Leerzeichen zu + usw.

So kodieren Sie eine URL in PHP

```
<?
  $original_url= "https://mein_shop/index.php";
  $params= array(
    'addr_name'      => "Hans Mustermann",
    'addr_street'    => "Musterstraße 1",
    'addr_city'      => "Musterstadt",
    'addr_zip'       => "12345",
    'addr_country'   => "DE"
  );
  $url_encoded= $original_url."?";
  foreach ($params as $key => $value) {
    $url_encoded.= $key."=".urlencode($value)."&";
  }
?>
```

So kodieren Sie eine URL in Perl

```
#!/usr/bin/perl -w

use strict;
use URI::Escape;

my $original_url= "https://mein_shop/index.pl";
my %params = (
  addr_name      => "Hans Mustermann",
  addr_street    => "Musterstraße 1",
  addr_city      => "Musterstadt",
  addr_zip       => "12345",
  addr_country   => "DE",
);
my $url_encoded= $original_url."?";

while ( my ($key, $value) = each(%params) ) {
  $url_encoded.= $key."=".uri_escape($value)."&";
}
print $url_encoded;
print "\n";
```

Sie können die Zeichen in Perl auch auf folgende Weise ersetzen:

```
$value =~ s/([^A-Za-z0-9])/sprintf("%%%02X", ord($1))/seg;
```

Überblick der Schnittstellen für Zahlungen per CGI

Für den Einsatz von ipayment über CGI steht Ihnen ein CGI-Script zur Verfügung, das über eine SSL-gesicherte Verbindung aufgerufen wird. Das Script arbeitet mit festen und optionalen Parametern. Welche Parameter benötigt werden, hängt in der Regel vom benutzten Modus und der Zahlungsart ab. Derzeit sind drei verschiedene Modi möglich: Der **Silent-Modus**, der **normale Modus** und der **Gateway-Modus**. Die drei CGI-Schnittstellen bieten Ihnen viel Flexibilität bei der Anbindung des ipayment-Zahlungssystems.

Empfohlene CGI-Integrationsmethoden

Über den **Silent-Modus** können Sie ipayment nahezu unsichtbar Integrieren. Der **Silent-Modus** ist die flexibelste Integrationsmöglichkeit für die Einbindung in einen eigenen Online-

Shop oder in eine eigene Website. In diesem Modus können Sie direkt auf Ihren Webseiten ein Formular zur Erfassung der Kartendaten erstellen, das keine direkten Hinweise auf ipayment enthält. Die eingegebenen Daten werden dennoch direkt an ipayment gesendet. Je nach Ergebnis (Erfolg oder Fehler) wird Ihr Kunde auf eine Seite innerhalb Ihres Shops weitergeleitet. Von ipayment selbst erfolgt keine Ausgabe.

Am einfachsten ist die Integration über den **normalen Modus**. Sowohl die HTML-Seiten für die Eingabe der Zahlungsdaten als auch die Bestätigungsseite werden direkt vom ipayment-Server ausgegeben. Das Design der HTML-Seiten können Sie im ipayment-Konfigurationsmenü selbst bestimmen. Für Sie fällt fast keine zusätzliche Arbeit an. Durch einen Link oder Button können Ihre Kunden von der Bestätigungsseite aus in Ihren Shop zurückkehren.

Wenn Sie 3-D Secure verwenden, wird Ihr Kunde bei Bedarf direkt von ipayment zu der Website seiner Bank weitergeleitet, damit dort sein 3-D-Secure-Passwort geprüft werden kann. Nach erfolgter Prüfung übernimmt ipayment automatisch alle nötigen Abläufe, so dass Ihr Kunde je nach Schnittstelle wieder direkt in Ihren Shop zurückkehrt.

Keine eigene Zertifizierung notwendig

Wenn Sie eine dieser beiden Integrationsmethoden verwenden, müssen Sie keine Zertifizierung nach dem PCI-DSS-Regelwerk vornehmen, weil die Zahlungsdaten nur vom ipayment-Server verarbeitet werden. Zusätzlich können Sie die 3-D-Secure-Sicherheitsverfahren einsetzen, ohne dass weitere Anpassungen notwendig sind.

Weitere CGI-Integrationsmethoden

Der dritte CGI-Modus ist der **Gateway-Modus**. Mit diesem Modus können Sie die Integration von ipayment komplett verstecken. Der Datenaustausch findet direkt zwischen Ihrer Anwendung und dem ipayment-Server statt. Wenn Sie diesen Integrationsmodus einsetzen, müssen Sie eine Zertifizierung nach dem PCI-DSS-Regelwerk vornehmen lassen, weil Ihr Shopsystem in diesem Fall direkt Kreditkartendaten verarbeitet. Anpassungen für die Verwendung des 3-D-Secure-Sicherheitsverfahrens müssen Sie ebenfalls an Ihrem Shop vornehmen.

Einbindung der Integrationsmethoden

Script testen und einbinden

Um die Einbindung zu testen, können Sie das folgende Formular verwenden:

https://ipayment.de/merchant/<Account-ID>/example/2.0/

Ersetzen Sie dabei den Teil `<Account-ID>` durch Ihre tatsächliche Account-ID. Das Beispielformular enthält alle wichtigen Parameter, so dass Sie die Funktionen ausgiebig testen können.

Das Script für die Abwicklung der Zahlungen finden Sie unter folgender URL:

https://ipayment.de/merchant/<Account-ID>/processor/2.0/

Ersetzen Sie auch hier den Teil `<Account-ID>` durch Ihre tatsächliche Account-ID.

Um die Scripts zu testen, können Sie die Test-Zugangsdaten verwenden. Mehr Informationen dazu finden Sie im Abschnitt *ipayment-Funktionen testen (Simulationsmodus)* auf Seite 9.

Bei erfolgreichen Transaktionen werden die beim Aufruf angegebenen Parameter `trxuser_id`, `trx_amount`, `trx_currency`, `trx_user_comment` und die Adressdaten Ihres Kunden in den Ergebnis-Parametern wiederholt.

Alle wichtigen Parameter werden nach der Transaktion an die im Parameter `redirect_url` angegebene URL übergeben. Zusätzliche Parameter, die das Shopsystem übergeben hat und die dem ipayment-System unbekannt sind, werden ebenfalls an die `redirect_url` übermittelt. Dies ist zum Beispiel nützlich, um shopeigene Daten wie Shop-Session-IDs oder ähnliches weiterzugeben.

Wenn die Transaktion aufgrund eines Fehlers nicht ausgeführt werden konnte, werden zusätzlich zu den üblichen Rückgabeparameter einige der beim Aufruf übergebenen Parameter wiederholt, wie zum Beispiel `trx_amount`, `silent`, `silent_error_url`, `redirect_url`, `redirect_action`, `trx_currency`, `hidden_trigger_url`. Weiterhin werden wie bei erfolgreichen Zahlungen zusätzliche vom Shop übergebene Parameter mit zurückgegeben. Die Liste der üblichen Rückgabeparameter finden Sie ab Seite 60 (Kapitel *Rückgabeparameter*).

Zahlungen per CGI im Silent-Modus

Der **Silent-Modus** erlaubt es Ihnen, ipayment unsichtbar in Ihren Shop einzubinden. Während des Bestellvorgangs werden die Zahlungsdaten direkt zu ipayment übermittelt und dort verarbeitet, ohne dass von ipayment eine für Ihren Kunden sichtbare Ausgabe erfolgt. Sie können ein Formular zur Erfassung der Kartendaten erstellen, das perfekt zum Layout Ihres Online-Shops passt. Die Rückmeldungen (Ergebnis oder Fehler) werden ebenfalls auf einer eigenen Seite in Ihrem Shop angezeigt. Dadurch verlässt Ihr Kunde während des Bestellvorgangs nie merklich Ihren Shop.

ipayment ist über den **Silent-Modus** relativ einfach in Ihre Webanwendung integrierbar. Verwenden Sie dazu die Parameter `redirect_url` (Erfolgsmeldung) oder `silent_error_url` (Fehlermeldung). Beide URLs müssen CGI-Scripte in Ihrem Shop sein, die aufgrund der übergebenen Parameter entsprechende Aktionen ausführen können. Solche Aktionen können zum Beispiel das Speichern der Bestellung oder die Ausgabe der Fehlermeldung sein.

Manipulationen der Zahlungsdaten verhindern

Neben den Zahlungsdaten werden im Silent-Modus beim Aufruf von ipayment die Basisdaten (zum Beispiel Ihre ipayment Account-Daten) und weitere feststehende Parameter (zum Beispiel Betrag, Währung und Einstellungen) übertragen. Diese Werte können vom Käufer eingesehen und auch manipuliert werden. Kein Problem ist es, wenn nach einer erfolgreichen Zahlung die Korrektheit dieser Werte geprüft wird.

Wenn Sie verhindern möchten, dass diese Parameter vom Käufer eingesehen oder manipuliert werden können, können Sie diese Daten vorab an ipayment übertragen und eine Session bei ipayment vorgenerieren. Danach wird beim Aufruf nur noch die ID dieser Session übertragen. Wie das genau funktioniert, können Sie im Kapitel *Session-IDs vorgenerieren* auf Seite 42 nachlesen.

Der ipayment-Server verwendet ein 256-Bit-SSL-Zertifikat. Dadurch ist die Datensicherheit der Zahlungsinformationen bei der Verarbeitung auf dem ipayment-Server gewährleistet. Damit die Sicherheit auch bei der Eingabe und Übermittlung der Daten gewährleistet werden kann, muss auf Ihren Seiten ebenfalls SSL vorhanden sein. Wenn Sie kein SSL verwenden, kann es passieren, dass einige Browser beim Rücksprung vom SSL-gesicherten ipayment-Server in Ihre Webanwendungen Warnungen ausgeben.

Zusätzlich empfehlen wir dringend, ein gesondertes „Hidden-Trigger-Script“ zu verwenden, um bei erfolgreichen Transaktionen gesicherte Rückmeldungen zu erhalten. Weitere Informationen zu diesem Verfahren finden Sie unter *Gesicherte Rückmeldung erfolgreicher Transaktionen* ab Seite 41.

Beim **Silent-Modus** können Sie 3-D Secure für „Verified by Visa“, „MasterCard SecureCode“ und „Maestro SecureCode“ einsetzen, ohne dass weitere Änderungen notwendig sind. 3-D Secure ist im **Silent-Modus** automatisch verfügbar. Eine gesonderte PCI-Zertifizierung ist ebenfalls nicht notwendig, da die Zahlungsdaten ausschließlich auf dem ipayment-Server verarbeitet werden.

Beispiel zur Anwendung

In unseren FAQ finden Sie ein Beispiel-Script für die Einbindung von ipayment im „Silent-Modus“:

http://faq.ipayment.de > **Fragen zur individuellen Anbindung von ipayment** > **Wie kann ich ipayment im „Silent-Modus“ benutzen?**

Zahlungen per CGI im Normalen Modus

Der **Normale Modus** ist die einfachste Möglichkeit, ipayment in Ihren Shop zu integrieren. In diesem Modus werden die HTML-Seiten für die Eingabe der Zahlungsdaten und die Rückgabe des Ergebnisses (Erfolg oder Fehler) vom ipayment-Server ausgegeben. Sie können das Design dieser HTML-Seiten im Konfigurationsmenü von ipayment anpassen. Parameter wie die Adressdaten können Sie schon beim Aufruf übergeben, diese werden dann im Formular vorausgefüllt. Wir empfehlen, die Zahlungsdaten nicht bereits beim Aufruf zu übergeben, da dabei eine Verarbeitung der Zahlungsdaten auf Ihrem System stattfindet und somit eine PCI-DSS-Zertifizierung notwendig wäre.

Manipulationen der Zahlungsdaten verhindern

Neben den Zahlungsdaten werden im „normalen Modus“ beim Aufruf von ipayment die Basisdaten (zum Beispiel Ihre ipayment Account-Daten) und weitere feststehende Parameter (zum Beispiel Betrag, Währung und Einstellungen) übertragen. Diese Werte können vom Käufer eingesehen und auch manipuliert werden. Kein Problem ist es, wenn nach einer erfolgreichen Zahlung die Korrektheit dieser Werte geprüft wird.

Wenn Sie verhindern möchten, dass diese Parameter vom Käufer eingesehen oder manipuliert werden können, können Sie diese Daten vorab an ipayment übertragen und eine Session bei ipayment vorgenerieren. Danach wird beim Aufruf nur noch die ID dieser Session übertragen. Wie das genau funktioniert, können Sie im Kapitel *Session-IDs vorgenerieren* auf Seite 42 nachlesen.

Nach erfolgter Transaktion wird Ihrem Kunden eine Bestätigungsseite angezeigt. Auf dieser Bestätigungsseite ist ein Link enthalten, über den Ihr Kunde in Ihren Shop zurückkehren kann. Alternativ können Sie einstellen, dass diese Bestätigungsseite nicht angezeigt wird. Stattdessen wird wie beim **Silent-Modus** eine Weiterleitung auf die Erfolgs-URL in Ihrer Web-Anwendung ausgeführt. Weitere Informationen finden Sie im Abschnitt *Redirect-Aktion* auf Seite 34.

Der ipayment-Server verwendet ein 256-Bit-SSL-Zertifikat. Dadurch ist die Datensicherheit der Zahlungsinformationen bei der Verarbeitung auf dem ipayment-Server gewährleistet. Damit die Sicherheit auch bei der Eingabe und Übermittlung der Daten gewährleistet werden kann, sollte auf Ihren Seiten ebenfalls SSL vorhanden sein. Wenn Sie kein SSL verwenden, kann es passieren, dass einige Browser beim Rücksprung vom SSL-gesicherten ipayment-Server an Ihre Webanwendungen Warnungen ausgeben.

Zusätzlich empfehlen wir dringend, ein gesondertes „Hidden-Trigger-Script“ zu verwenden, um bei erfolgreichen Transaktionen gesicherte Rückmeldungen zu erhalten. Weitere Informa-

tionen zu diesem Verfahren finden Sie unter *Gesicherte Rückmeldung erfolgreicher Transaktionen* ab Seite 41.

Beim **Normalen Modus** können Sie 3-D Secure für „Verified by Visa“, „Mastercard SecureCode“ und „Maestro SecureCode“ einsetzen, ohne dass weitere Änderungen notwendig sind. 3-D Secure ist im **Normalen Modus** automatisch verfügbar. Eine gesonderte PCI-Zertifizierung ist ebenfalls nicht notwendig, da die Zahlungsdaten ausschließlich auf dem ipayment-Server verarbeitet werden.

Zahlungen per CGI im Gateway-Modus

Die Integration über den **Gateway-Modus** ist kompliziert, bietet aber viele Vorteile, weil Sie in diesem Modus die meisten Einstellungen vornehmen können. Bei diesem Integrationsmodus können Sie ipayment so in Ihre Anwendung einbinden, dass Ihre Kunden keinerlei Hinweise auf ipayment erhalten. Sie senden die Zahlungsanfrage im Hintergrund an ipayment und erhalten das Ergebnis direkt als URL-kodierten Parameter-String zurück. Dadurch können Sie ipayment flexibel in Ihre bereits bestehenden Systeme integrieren. Die einzige Voraussetzung hierfür besteht darin, dass Ihr Programm eine SSL-Verbindung aufbauen kann.

PCI-Zertifizierung notwendig

Beim **Gateway-Modus** werden die Zahlungsdaten nicht nur vom ipayment-System verarbeitet, sondern auch durch Ihre Anwendung. Deshalb müssen Sie sich um die kostenpflichtige PCI-Zertifizierung kümmern, die von vielen Zahlungsanbietern gefordert wird. Lesen Sie mehr zum Thema PCI-Zertifizierung unter *Was ist PCI DSS?* auf Seite 11.

Die Rückgabe des Ergebnisses erfolgt in zwei Zeilen. In der ersten Zeile steht der Status. „Status=0“ bedeutet, dass die Abwicklung erfolgreich war. Im Fehlerfall wird „Status=1“oder „Status=-1“ zurückgeliefert. In der zweiten Zeile stehen die Ergebnisparameter mit allen Rückgabewerten (beginnend mit „Params=“).

Da Transaktionen aufgrund der vielen weiteren beteiligten Bankensystemen in Einzelfällen auch länger dauern können, empfehlen wir für die Kommunikation ein Timeout von ca. 5 bis 10 Minuten. Wenn das Kommunikations-Timeout geringer ist, kann es vorkommen, dass Ihr Shop-System die Antwort vom ipayment-Server nicht erreicht, obwohl die Transaktion eigentlich erfolgreich war.

Adressprüfungen per CGI

Die Adressprüfung per CGI findet über ein Script statt.

Adressprüfung einbinden und testen

Um die Einbindung des Adresschecks zu testen, können Sie das folgende Formular verwenden:

https://ipayment.de/merchant/<Account-ID>/example_addrcheck.php

Ersetzen Sie dabei den Teil `<Account-ID>` durch Ihre tatsächliche Account-ID. Das Beispielformular enthält alle wichtigen Parameter, so dass Sie die Funktionen ausgiebig testen können.

Das Script für die Abwicklung der Adressprüfung finden Sie unter folgender URL:

https://ipayment.de/merchant/<Account-ID>/addresscheck.php

Ersetzen Sie auch hier den Teil `<Account-ID>` durch Ihre tatsächliche Account-ID.

Die Rückgabewerte sind aufgrund Ihres komplexen Aufbaus bei der CGI-Schnittstelle mehrzeilig. Die Zeilen sind alle wie ein HTTP-GET-String aufgebaut und codiert.

In der ersten Zeile, die immer vorhanden ist, wird ein Fehlercode im Parameter `ret_errorcode` zurückgegeben. Wenn dieser Fehlercode ungleich 0 ist, existiert nur diese erste Zeile. Weitere Informationen zum aufgetretenen Fehler werden über die Parameter `ret_errormsg` und `ret_additionalmsg` zurückgegeben. Die Fehlercodes entsprechen den im gesamten ipayment-System verwendeten Fehlercodes. Wenn der Fehlercode gleich 0 ist, existiert noch ein Wert `status`, der den Gesamtstatus der Adressprüfung angibt. Darauf folgen die Ergebniswerte für die geprüften Adressfelder, wobei pro Zeile das Ergebnis eines Feldes ausgegeben wird. Weitere Informationen dazu lesen Sie unter *Rückgabeparameter Adressprüfung* ab Seite 65.

Eine Beispiel-Antwort für eine erfolgreiche Prüfung:

```
ret_errorcode=0&status=ERROR
ret_field=addr_street&status=CORRECTED&origvalue=B%E4ren&suggestionlist=
B%E4derstr.&statusdetail=street%2Fvalue+uniquely+determined
ret_field=addr_city&status=OK&origvalue=Karlsruhe
ret_field=addr_zip&status=SUGGESTIONS&origvalue=01000&suggestionlist=761
89|76149|76227|19322&statusdetail=postcode%2Fsuggestions+found
ret_field=addr_street_number&status=ERROR&origvalue=&statusdetail=street
_number%2Fvalue+indeterminable
```

Eine Beispiel-Antwort für einen Fehler:

```
ret_errorcode=2004&ret_fatalerror=1&ret_errormsg=The+given+payment+infor
ma-
tion+is+not+valid.&ret_additionalmsg=Country+is+needed+for+addresscheck.
```

Integration per SOAP-Webservice

Sie können die komplette ipayment-Funktionalität über einen SOAP-Webservice nutzen. Hierfür steht Ihnen eine WSDL-Beschreibung zur Verfügung. Den Service finden Sie unter

https://ipayment.de/service/3.0/ Dort finden Sie sowohl eine kurze Beschreibung des Webservices mit allen Funktionen, als auch die entsprechende WSDL-Beschreibungsdatei.

Die Webservice-Schnittstelle bietet neben den reinen Zahlungsfunktionen des ipayment-Systems auch Funktionen zur Adress- und E-Mail-Prüfung.

Anders als bei der CGI-Integration können Sie hier keine „Key-Value-Paare" übergeben, sondern verschiedene Funktionen für die einzelnen Aktionen ausführen. Die Parameter dieser Funktionen sind meist Strukturen mit den möglichen Datenfeldern. Optionale Felder können Sie dabei weglassen.

Zahlungen per SOAP-Webservice

Über den SOAP-Webservice können Sie die am meisten verwendeten Zahlungsparameter des ipayment-Systems einsetzen. Zusätzliche Parameter können Sie zudem über den `otherOptions`-Hash in der `optionData`-Struktur übergeben. Die Key-Namen der Hash-Einträge entsprechen den Parameternamen, die auch bei der CGI-Integration verwendet werden.

Die Funktionen des Webservice zur Zahlungsabwicklung sind so aufgeteilt, dass pro Transaktionstyp eine eigene Funktion mit den jeweils nötigen und erlaubten Parametern zur Verfügung steht.

Da Transaktionen aufgrund der vielen weiteren beteiligten Bankensystemen in Einzelfällen auch länger dauern können, empfehlen wir für die Kommunikation ein Timeout von ca. 5 bis 10 Minuten. Wenn das Kommunikations-Timeout geringer ist, empfängt Ihr System eventuell die Antwort des ipayment-Servers nicht, obwohl die Transaktion erfolgreich war.

PCI-Zertifizierung notwendig

Beim SOAP-Webservice werden die Zahlungsdaten nicht nur vom ipayment-System verarbeitet, sondern auch durch Ihre Anwendung. Deshalb müssen Sie eventuell eine kostenpflichtige PCI-Zertifizierung absolvieren, die von vielen Zahlungsanbietern gefordert wird. Lesen Sie mehr zum Thema PCI-Zertifizierung unter *Was ist PCI DSS?* auf Seite 11.

Der SOAP-Webservice bietet die nachfolgend beschriebenen Methoden für die Transaktionsabwicklung. Anhand des Elements `<paymentData>` wird automatisch ermittelt, ob die Transaktion per Kreditkarte oder ELV ausgeführt werden soll..

Methode	Beschreibung
`authorize`	Führt eine Autorisierung mit sofortiger Abbuchung durch. Mehr Informationen: *Sofortige Buchung einer Zahlung (auth)* auf Seite 13.
`basecheck`	Führt die Basisprüfung der Zahlungsdaten durch. Es wird keine Online-Abfrage beim Zahlungsanbieter gemacht. Mehr Informationen: *Plausibilitätsprüfung der Zahlungsdaten (base_check)* auf Seite 14.
`capture`	Führt die Verbuchung einer zuvor autorisierten Zahlung durch. Mehr Informationen: *Abbuchen von vorautorisierten Transaktionen (capture)* auf Seite 14.
`checkAddress`	Führt eine Adressprüfung durch. Als Rückgabewert erhalten Sie einen globalen Status-Wert zur Prüfung und weitere Rückgabewerte pro Adressfeld.
`checkEmail`	Führt eine Überprüfung auf syntaktische Korrektheit von E-

Methode	Beschreibung
	Mail-Adressen durch. Außerdem wird überprüft, ob E-Mails an die angegebene Domain zustellbar wären. Es wird nicht geprüft, ob die E-Mail-Adresse tatsächlich existiert, sondern ob die MX-Einträge für diese Domain verfügbar sind oder ob die Domain selbst E-Mails annehmen kann.
`checksave`	Führt eine Prüfung der Zahlungsdaten beim Zahlungsanbieter durch. Mehr Informationen: *Detaillierte Prüfung der Zahlungsdaten (check_save)* auf Seite 14.
`createSession`	Generiert eine Session für den **Normalen Modus** oder den **Silent-Modus**. Mehr Informationen: Kapitel *Session-IDs vorgenerieren* auf Seite 42.
`generalRefund`	Führt eine freie Gutschrift auf eine Kreditkarte aus, die bislang noch nicht im ipayment-System verwendet wurde. Mehr Informationen: *Freie Gutschriften ausführen (grefund_cap)* auf Seite 16.
`paymentAuthenticationReturn`	Führt eine Übermittelung der Daten `MD` und `PaRes` durch. Diese Daten werden als Rückgabewerte übergeben, nachdem ein Karteninhaber beim Verwenden von 3-D Secure auf seine Bankenseite weitergeleitet wurde und sein Passwort eingegeben hat. Nachdem diese Daten an ipayment zurückgeliefert wurden, wird die Transaktion normal durchgeführt, der Betrag entsprechend belastet und das Ergebnis von ipayment zurückgegeben. Mehr Informationen: *Ablauf einer Zahlung mit Authentifizierung per 3-D Secure* auf Seite 44.
`preAuthorize`	Führt die Reservierung eines Betrags durch, ohne dass dieser verbucht wird. Mehr Informationen: *Verzögerte Zahlungsabwicklung/Vorautorisierung (preauth)* auf Seite 13.
`reAuthorize`	Führt eine Zahlung erneut durch, basierend auf einer bereits im ipayment-System vorhandenen Transaktion. Mehr Informationen: *Erneute sofortige Buchung (re_auth)* auf Seite 16.
`refund`	Führt eine Gutschrift auf eine erfolgte Zahlung durch. Mehr Informationen: *Rückbuchung eines Betrages (refund_cap)* auf Seite 15.
`rePreAuthorize`	Führt eine Vorautorisierung erneut durch, basierend auf einer bereits im ipayment-System vorhandenen Transaktion. Mehr Informationen: *Erneute Vorautorisierung (re_preauth)* auf Seite 16.
`reverse`	Für die Stornierung einer Vorautorisierung durch, die noch nicht verbucht wurde. Mehr Informationen: *Stornieren einer Vorautorisierung (reverse)*auf Seite 15.
`voiceAuthorizeCC`	Führt eine Abbuchung einer zuvor telefonisch autorisierten

Methode	Beschreibung
	Zahlung durch. Mehr Informationen: *Abbuchen einer telefonisch autorisierten Zahlung (voice_auth)* auf Seite 17.
`voiceGeneralRefundCC`	Führt eine Abbuchung einer telefonisch autorisierten Gutschrift durch. Mehr Informationen: *Abbuchen einer telefonisch autorisierten Gutschrift (voice_grefund_cap)* auf Seite 17.

Parameter zur Zahlungsabwicklung

In diesem Kapitel werden alle ipayment-Parameter zur Zahlungsabwicklung aufgelistet. Zu jedem Parameter finden Sie den CGI- und Webservice-Namen sowie den Datentyp. Die Namen sind wie folgt aufgebaut:

- **CGI**
  Name des Parameters in Kleinbuchstaben (zum Beispiel `trxuser_id`)
- **Webservice**
  Aufbau nach dem Schema `Feldstruktur/Parametername`, teilweise mit Großschreibung (zum Beispiel `AccountData/trxuserId`)

Verwendete Datentypen

Mögliche Datentypen bei den Parametern des ipayment-Systems sind:

- **Boolean**
  Parameter, die diesen Datentyp verwenden, können nur zwei Werte annehmen: `true` (1) oder `false` (0). Bei der Integration via CGI empfehlen wir, die numerischen Werte (0 oder 1) zu verwenden.
- **Integer**
  Dieser Datentyp enthält ganze Zahlen (32 Bit) im Wertebereich von -2147483648 bis 2147483647 ohne Nachkommastellen.
- **Long**
  Dieser Datentyp wird für alle größeren ganzen Zahlen verwendet und bietet einen höheren Wertebereich als der Datentyp Integer. Auch dieser Datentyp enthält keine Nachkommastellen.
- **String**
  Mit diesem Datentyp werden Texte abgebildet. Es sind alle Zeichen erlaubt. Wenn nicht anders angegeben, darf der Wert nicht länger als 255 Zeichen sein.

Basisparameter

Parameter zur Identifikation des ipayment-Accounts

Die nachfolgenden Parameter sind zur Identifikation des ipayment-Accounts notwendig.

Account-ID

CGI-Name:	`- (In der URL des Scripts enthalten)`
Webservice-Name:	`AccountData/accountId`
Datentyp:	`Integer`

ID des verwendeten ipayment-Accounts. Sie finden diesen Wert in Ihrem ipayment-Konfigurationsmenü unter **Allgemeine Daten**.

Anwendungs-ID

CGI-Name:	`trxuser_id`
Webservice-Name:	`AccountData/trxuserId`
Datentyp:	`Integer`

Die Anwendungs-ID ist gemeinsam mit der Account-ID die eindeutige Bezeichnung des Händlers. Innerhalb eines Accounts können Sie mehrere Anwendungen anlegen und benutzen, zum

Beispiel um ipayment an mehrere Shops anzubinden. Die Anwendungs-ID können Sie in Ihrem ipayment-Konfigurationsmenü unter **Anwendung > Details** auslesen.

Anwendungspasswort

CGI-Name:	`trxpassword`
Webservice-Name:	`AccountData/trxpassword`
Datentyp:	`Long`

Für jede Anwendung gibt es ein Anwendungspasswort, das automatisch vom ipayment-System vergeben wird. Das Passwort besteht aus Zahlen. Sie finden das Anwendungspasswort in Ihrem ipayment-Konfigurationsmenü unter **Anwendungen > Details**. Dieser Wert ist ausschließlich für die Abwicklung von Transaktionen nötig. Ein Login in das ipayment-Konfigurationsmenü ist mit diesem Passwort nicht möglich.

Admin-Aktions-Passwort

CGI-Name:	`adminactionpassword`
Webservice-Name:	`AccountData/adminactionpassword`
Datentyp:	`String`

Das Admin-Aktions-Passwort wird nur für administrative Transaktionen wie Stornierungen, Rückbuchungen oder Abbuchungen benötigt.

Durch diesen Parameter können Sie sicherstellen, dass nur befugte Scripte die entsprechenden Aktionen über die CGI- oder WSDL-Schnittstelle durchführen können. Das Admin-Aktions-Passwort finden Sie in Ihrem ipayment-Konfigurationsmenü unter **Anwendung > Details**. Ein Login in das ipayment-Konfigurationsmenü ist mit diesem Passwort nicht möglich.

Parameter für Betrag und Währung

Mit diesen Parametern können Sie den Betrag und die Währung der Transaktion übergeben.

Währung der Transaktion

CGI-Name:	`trx_currency`
Webservice-Name:	`TransactionData/trxCurrency`
Datentyp:	`String, exakt 3 Buchstaben`

Währung, in der die Zahlung abgewickelt wird. Es sind alle bekannten dreistelligen ISO-Währungscodes erlaubt. Eine Liste der bekannten Währungscodes finden Sie unter https://ipayment.de/ **> Technik**. Beachten Sie, dass die Abwicklung von Zahlungen in der angegebenen Währung mit Ihrem Zahlungsanbieter vereinbart sein muss.

Betrag der Transaktion

CGI-Name:	`trx_amount`
Webservice-Name:	`TransactionData/trxAmount`
Datentyp:	`Integer, nur positiver Wert, maximal 10.000.000`

Betrag, der abgebucht wird. Geben Sie den Wert in der kleinsten Währungseinheit ein, zum Beispiel Cent. Dezimalpunkte oder andere Zeichen außer Zahlen sind nicht erlaubt.

Beispiel: Der Betrag von 10,00 Euro wird als 1000 Cent angegeben.

Betrag der Transaktion als einzelne Bestandteile

CGI-Name:	`trx_amount_base, trx_amount_decimal`
Webservice-Name:	`- (Nicht benötigt)`
Datentyp:	`Integer, nur positive Werte`

Mit `trx_amount_base` und `trx_amount_decimal` können Sie den Betrag getrennt nach Vor- und Nachkommastellen übergeben. Dadurch wird eine Umrechung des Betrags in die kleinste

Währungseinheit überflüssig. `trx_amount_base` steht für den Betrag vor dem Komma (Euro) und `trx_amount_decimal` für den Betrag nach dem Komma (Cent). Ein Beispiel: Bei einem Betrag von 10,99 Euro ist `trx_amount_base` = 10 und `trx_amount_decimal` = 99.

Wenn `trx_amount` gesetzt ist, werden die hier genannten Parameter ignoriert.

Parameter zur Angabe der gewünschten Zahlung

Mit diesen Parametern können Sie angeben, welche Transaktionen mit welchem Zahlungsmedium ausgeführt werden.

Transaktionstyp der Transaktion

CGI-Name:	`trx_typ`
Webservice-Name:	`- (entspricht dem Methodendnamen des SOAP-Webservices)`
Datentyp:	`String, Werte siehe Beschreibung`

Je nach verwendetem ipayment-Modus, Zahlungsanbieter und Zahlungsmedium sind folgende Transaktionstypen erlaubt:

Alle Modi

- **preauth**
- **auth**
- **base_check**
- **check_save**
- **voice_auth**
- **voice_grefund_cap**

Silent-Modus, Gateway-Modus und SOAP-Webservice

- **re_preauth**
- **re_auth**
- **capture**
- **reverse**
- **refund_cap**
- **grefund_cap**

Wenn der Parameter nicht angegeben ist, wird automatisch **auth** als Transaktionstyp verwendet. Genauere Informationen zu diesem Transaktionstyp finden Sie im Kapitel *Sofortige Buchung einer Zahlung (auth)* auf Seite 13.

Originale Transaktionsnummer für Aktionen zu Transaktionen

CGI-Name:	`orig_trx_number`
Webservice-Name:	`origTrxNumber (in Methoden als Parameter wenn nötig)`
Datentyp:	`String`

Bei einigen Transaktionstypen benötigen Sie für die Durchführung einer Aktion als einzigen Parameter die Transaktionsnummer der originalen Transaktion. Diese Nummer übermitteln Sie mit `orig_trx_number`.

Zahlungsart der Transaktion

CGI-Name:	`trx_paymenttyp`
Webservice-Name:	`- (über angegebene Zahlungsdaten in PaymentData definiert)`
Datentyp:	`String, Werte siehe Beschreibung`

Mit diesem Parameter geben Sie an, wie die Zahlung erfolgen soll. Mögliche Werte: `cc` für Kreditkartenzahlung, `elv` für ELV-Zahlungen oder `pp` für Prepaid-Zahlungen. Wenn dieser Parameter nicht gesetzt ist, versucht das System, den Typ der Zahlungsart automatisch zu ermitteln. Wenn dies nicht möglich ist, wird der Wert auf `cc` (Kreditkartenzahlung) gesetzt.

Parameter für Name und Adresse des Karteninhabers

Mit diesen Parametern können Sie persönliche Daten des Karteninhabers übermitteln.

Name des Käufers

CGI-Name:	`addr_name`
Webservice-Name:	`AddressData/addrName`
Datentyp:	`String, maximal 100 Zeichen`

Name des Käufers. Dieser Parameter wird für alle Zahlungen benötigt.

E-Mail des Käufers

CGI-Name:	`addr_email`
Webservice-Name:	`AddressData/addrEmail`
Datentyp:	`String, maximal 80 Zeichen`

E-Mail-Adresse des Käufers. Wenn dieses Feld ausgefüllt wurde, wird die E-Mail-Adresse auch auf Plausibilität geprüft.

Adressdaten des Käufers

CGI-Name:	`addr_street, addr_city, addr_zip, addr_country`
Webservice-Name:	`AddressData/addrStreet, AddressData/addrCity, AddressData/addrZip, AddressData/addrCountry`
Datentyp:	`String, Länge siehe Beschreibung`

Adressdaten des Käufers. In diesen Parametern können Sie die Straße (maximal 255 Zeichen), die Stadt (maximal 50 Zeichen), die Postleitzahl (maximal 20 Zeichen) und ISO-Country-Code (3 Zeichen) des Karteninhabers übergeben. Der ISO-Country-Code basiert auf der ISO-Ländercodeliste, die Sie unter https://ipayment.de/ > **Technik** einsehen können. Wenn die Straße, die Postleitzahl oder der Ort angegeben wurden, müssen auch die anderen Felder ausgefüllt sein.

Weitere Adressdaten des Käufers

CGI-Name:	`addr_street2, addr_state`
Webservice-Name:	`AddressData/addrStreet2, AddressData/addrState`
Datentyp:	`String, Länge siehe Beschreibung`

Weitere Adressdaten des Käufers. In diesen Parametern können Sie den Straßenzusatz (maximal 255 Zeichen) und den Bundesstaat (2 Zeichen) übermitteln. Der Bundesstaat ist nur dann nötig, wenn der Karteninhaber in den USA oder in Kanada lebt. Die möglichen ISO-Codes können Sie in den ISO-Staaten-Listen unter https://ipayment.de/ **> Technik** einsehen.

Telefon und Telefax des Käufers

CGI-Name:	`addr_telefon, addr_telefax`
Webservice-Name:	`AddressData/addrTelefon, AddressData/addrTelefax`
Datentyp:	`String, jeweils maximal 30 Zeichen`

Telefonnummer und Telefaxnummer des Käufers. Diese Informationen werden ausschließlich gespeichert, aber nicht durch den Adresscheck geprüft.

Parameter zur Kennzeichnung von Transaktionen

Mit den Parametern können Sie Transaktionen mit eigenen IDs kennzeichnen, die im ipayment-System gespeichert werden.

Eindeutigkeit sicherstellen

Sie vermeiden Doppeltransaktionen, indem Sie eindeutige IDs vergeben. Weitere Informationen zur Vermeindung von Doppelstransaktionen finden Sie im Kapitel *Sichere Integration von ipayment* ab Seite 67.

Shopper-ID

CGI-Name:	`shopper_id`
Webservice-Name:	`TransactionData/shopperId`
Datentyp:	`String`

Mit diesem Parameter können Sie eine eigene ID für einen Bestellvorgang angeben. Unter dieser Shopper-ID wird die zur Bestellung gehörende Transaktion im ipayment-System gespeichert. Die Shopper-ID muss nur dann eindeutig sein, wenn die erweiterte Prüfung der IDs zur Vermeidung von Doppeltransaktionen verwendet wird.

Erweiterte Prüfung der IDs zur Vermeidung von Doppeltransaktionen durchführen?

CGI-Name:	`advanced_strict_id_check`
Webservice-Name:	`OptionData/advancedStrictIdCheck`
Datentyp:	`Boolean`

Wenn dieser ID-Check aktiv ist, wird vor der Abwicklung einer Transaktion geprüft, ob schon eine erfolgreiche Transaktion mit der angegebenen Shopper-ID abgewickelt wurde. Folgende Rückmeldungen sind möglich:

- **Die Transaktion existiert und alle Daten stimmen überein:**
  Die Zahlung wird nicht noch einmal abgewickelt. Stattdessen werden die Ergebnis-Parameter der gefundenen Transaktion zurückgegeben. Dadurch können Sie Doppelbuchungen effektiv verhindern, ohne dass Sie an Ihrer Anwendung etwas ändern müssen.
- **Die Transaktion existiert, aber die Zahlungsdaten stimmen nicht überein:**
  Die Zahlung wird mit einer Fehlermeldung abgelehnt. In diesem Fall generiert Ihr System möglicherweise keine eindeutigen Shopper-IDs.
- **Die Transaktion existiert nicht:**
  Die Zahlung wird normal abgewickelt.

Sie aktivieren den Check, indem Sie den Wert des Parameters auf 1 (`true`) setzen.

Parameter zur Referenzierung der Transaktion

Mit diesen Parametern können Sie Transaktionen weitere Informationen mitgeben. Diese Informationen werden bei ipayment gespeichert oder an Ihre Zahlungsanbieter oder Kunden weitergegeben.

Transaktionsreferenz/Rechnungsnummer

CGI-Name:	`invoice_text`
Webservice-Name:	`TransactionData/invoiceText`
Datentyp:	`String, maximale Länge siehe Beschreibung`

Bei Zahlungsabwicklungen mit einem der untenstehenden Zahlungsanbieter können Sie einen Text angeben, der an den Zahlungsanbieter übermittelt wird. Dieser Text sollte die Abbuchung genauer beschreiben. Je nach Zahlungsanbieter, kartenausgebender Stelle und Kreditkarten wird dieser Text auf der Kartenabrechnung des Kunden und/oder des Händlers ausgedruckt.

Wenn dieser Parameter nicht gesetzt ist, verwendet ipayment automatisch den Firmennamen des Händlers, den Sie im ipayment-Konfigurationsmenü unter **Allgemeine Daten** angegeben haben.

Folgende Zahlungsanbieter unterstützen dieses Textfeld:

Kreditkarten

- Acceptance (max. 30 Zeichen)
- American Express (gesondert anzumelden, max. 30 Zeichen, wird auf der Abrechnung des Karteninhabers gelistet.)
- B+S Cardservice GmbH (max. 25 Zeichen)
- ConCardis GmbH (max. 30 Zeichen)
- Deutsche Postbank AG (max. 30 Zeichen)
- Elavon (vormals Euroconex, max. 30 Zeichen)
- Telekurs Multipay AG (max. 30 Zeichen)
- Volksbanken/DZ Bank (max. 30 Zeichen)

Elektronisches Lastschriftverfahren

- Intercard (max. 2x27 Zeichen in 2 Zeilen, mit einem „\n“ als Zeilenumbruch getrennt)
- UPay (max. 10 Zeichen, standardmäßig wird kein Wert übergeben)

Transaktionskommentar

CGI-Name:	`trx_user_comment`
Webservice-Name:	`TransactionData/trxUserComment`
Datentyp:	`String`

Kommentar, der bei der Transaktion im ipayment-System gespeichert wird. Dieser Kommentar wird nicht an die Bank oder den Zahlungsanbieter übermittelt.

Parameter für Rücksprünge in den Shop

Bei den CGI-Modi **Normaler Modus** und **Silent-Modus** werden URLs für Rücksprünge in den Shop benötigt. Dadurch kann Ihr Kunde nach einer erfolgreichen Zahlung oder nach einem Fehler wieder in Ihr Shop-System gelangen. Diese URLs können Sie mit den folgenden Parametern übergeben:

Rücksprungs-URL für den Erfolgsfall

CGI-Name:	`redirect_url`
Webservice-Name:	`- (nicht benötigt)`
Datentyp:	`String`

Diese URL wird nach erfolgreicher Zahlung im **Normalen Modus** oder **Silent-Modus** aufgerufen. Im **Normalen Modus** bestimmt der Parameter `redirect_action`, ob diese Seite als normaler Link (Zahlungsparameter per `GET`), als Formular (Zahlungsparameter per `POST`) oder über eine Weiterleitung (Zahlungsparameter per `GET`) aufgerufen wird. Im **Silent-Modus** werden die Parameter immer per `GET` an das Script übergeben.

Redirect-Aktion

CGI-Name:	redirect_action
Webservice-Name:	- (nicht benötigt)
Datentyp:	String, Werte siehe Beschreibung

Dieser Parameter wird nur im **Normalen Modus** beachtet. Er bestimmt, wie die Parameter an die Erfolgs-URL übermittelt werden. Je nach Übermittlungsmodus wird ein Link ohne Parameter (Wert `GET`) oder ein Formular angezeigt (Wert `POST`). Beim Wert `REDIRECT` wird eine direkte Weiterleitung durchgeführt. Dabei wird keine Erfolgsseite von ipayment angezeigt. Wenn der Parameter weggelassen wird, erfolgt die Anzeige der Ergebnisseite mit einem Link wie bei dem Wert `GET`.

Sollen Parameter beim Erfolgs-Redirect zurückgegeben werden?

CGI-Name:	noparams_on_redirect_url
Webservice-Name:	- (nicht benötigt)
Datentyp:	Boolean

Mit diesem Parameter können Sie festlegen, ob nach einer erfolgreichen Transaktion die ipayment-Parameter übergeben werden sollen. Der Wert 1 (`true`) bedeutet, dass die Parameter nicht übergeben werden. In diesem Fall sollten Sie ein Hidden-Trigger-Script verwenden, um die Information zu übermitteln, dass die Zahlung erfolgreich war.

Funktion verhindert Browserprobleme

Einige Browser (wie der Internet Explorer 5) haben Beschränkungen in Bezug auf die Länge einer URL, wodurch es zu Problemen beim Redirect kommen kann. Verwenden Sie den Parameter `noparams_on_redirect_url`, um diese Probleme zu verhindern.

Dieser Parameter wird nur im **Silent-Modus** beachtet.

Rücksprungs-URL für Fehler im CGI-Silent-Modus

CGI-Name:	silent_error_url
Webservice-Name:	- (nicht benötigt)
Datentyp:	String

Diese URL wird im Fehlerfall vom ipayment-System mit den Fehlerinformationen und weiteren Parametern mittels `GET` aufgerufen. Diese URL muss auf ein CGI-Script verweisen, das die Parameter verarbeiten kann.

Dieser Parameter wird nur im **Silent-Modus** beachtet.

Sollen Parameter beim Fehler-Redirect zurück mitgegeben werden?

CGI-Name:	noparams_on_error_url
Webservice-Name:	- (nicht benötigt)
Datentyp:	Boolean

Mit diesem Parameter können Sie einstellen, ob nach einer abgelehnten Transaktion die ipayment-Parameter an Ihren Shop übergeben werden sollen oder nicht. Beim Wert 1 (`true`) werden keine Parameter übergeben.

Funktion verhindert Browserprobleme

Einige Browser (wie der Internet Explorer 5) haben Beschränkungen in Bezug auf die Länge einer URL, wodurch es zu Problemen beim Redirect kommen kann. Verwenden Sie den Parameter `noparams_on_error_url`, um diese Probleme zu verhindern.

Dieser Parameter wird nur im **Silent-Modus** beachtet.

Fehler-Redirect-URL für Prepaid-Zahlungen

CGI-Name:	`backlink`
Webservice-Name:	`- (nicht benötigt)`
Datentyp:	`String`

Über diese URL kann Ihr Kunde im Fehlerfall in Ihren Shop zurückkehren. Dieser Parameter wird nur bei Prepaid-Zahlungen beachtet und kommt dann zum Einsatz, wenn der Käufer den Zahlungsvorgang abbricht. Wenn dieser Parameter nicht gesetzt ist, versucht ipayment, diesen mit der URL der aufgerufenen Seite (HTTP-Referer) zu füllen. Wenn der Wert HTTP-Referer nicht vorhanden ist, wird die vorherige Seite per JavaScript über die History-Funktion des Browsers aufgerufen.

Parameter für die Durchführung der Sicherheitsprüfungen

Mit Hilfe dieser Parameter können Sie bei jeder Transaktion bestimmen, ob umfangreiche Sicherheitsprüfungen durchgeführt werden sollen.

Sollen die Betrugsprüfungssysteme benutzt werden?

CGI-Name:	`check_fraudattack`
Webservice-Name:	`OptionData/checkFraudattack`
Datentyp:	`Boolean`

Die Prüfung auf einen eventuellen Betrugsversuch ist standardmäßig aktiviert. Wenn ein Betrugsversuch erkannt wird, wird die Transaktion mit einer entsprechenden Fehlermeldung abgelehnt. Welche Prüfungen durchgeführt werden sollen, können Sie für jede Anwendung in Ihrem ipayment-Konfigurationsmenü unter **Anwendungen** einstellen.

Mit dem Wert 0 (`false`) deaktivieren Sie die Prüfungen. Wir empfehlen, die Prüfung vor allem im Online-Shop-Umfeld durchzuführen.

Soll eine Prüfung auf Doppeltransaktionen durchgeführt werden?

CGI-Name:	`check_double_trx`
Webservice-Name:	`OptionData/checkDoubleTrx`
Datentyp:	`Boolean`

Das ipayment-System erkennt doppelte Zahlungsanfragen innerhalb von 2 Minuten und lehnt die zweite Zahlungsanfrage ab. Standardmäßig ist diese Prüfung aktiviert. Wenn als Wert 0 (`false`) eingetragen ist, wird die Prüfung auf Doppeltransaktionen vom ipayment-System deaktiviert.

Prüfung auf Doppeltransaktionen immer aktivieren

Wir empfehlen, die Prüfung immer aktiviert zu lassen, damit keine Doppelbuchungen vorkommen können. Schalten Sie die Prüfung nur dann aus, wenn Sie in Ihrer Anwendung sicherstellen können, dass Doppelbuchungen unmöglich sind oder wenn Sie eindeutige IPs mit der erweiterten Prüfung einsetzen.

Parameter für Einstellungen des Zahlungssystems

Für jede ipayment-Anwendung können Sie direkt im ipayment-Konfigurationsmenü einstellen, wie das System in bestimmten Situationen reagieren soll. Dies gilt jedoch nicht für alle Einstellungen. Die folgenden Parameter können Sie nur außerhalb des ipayment-Konfigurationsmenüs einstellen:

Sollen Details zu den Zahlungsdaten der Transaktion zurückgegeben werden?

CGI-Name:	`return_paymentdata_details`
Webservice-Name:	`- (nicht unterstützt)`
Datentyp:	`Boolean`

Über diesen Parameter können Sie einstellen, ob ipayment die benutzten Zahlungsdaten wieder an Ihre Anwendung zurückgeben soll. Sensible Daten, wie zum Beispiel eine Kreditkartennummer, werden entsprechend maskiert zurückgegeben. Die Namen der Rückgabeparameter beginnen mit `paydata_`. Danach folgt der Name der Eingabeparameter. Beispiel für einen Parameter: `paydata_cc_number` (maskierte Kreditkartennummer).

Der Parameter ist aktiv, wenn der Wert 1 (`true`) beträgt. Standardmäßig werden keine Zahlungsdaten zurückgegeben. Die Rückgabe der Zahlungsdaten ist besonders dann interessant, wenn Ihr Kunde bei Ihnen registriert ist, Sie wiederkehrende Zahlungen haben oder den Storage-Service von ipayment verwenden. Die Zahlungsdaten lassen den Kunden auf einen Blick erkennen, ob seine gespeicherten Daten noch aktuell sind.

Sollen die Daten der Transaktion länger gespeichert werden?

CGI-Name:	`trx_longsave`
Webservice-Name:	`OptionData/trxLongsave`
Datentyp:	`Boolean`

Über diesen Parameter stellen Sie ein, wie lange ipayment die personenbezogenen Daten der Transaktion speichert. Bei dem Wert 1 (`true`) wird von einem längeren Vertragsverhältnis als 3 Monate ausgegangen. In diesem Fall wird für die Transaktion der für den Account längstmöglichste Speicherzeitraum genutzt. Bei 0 (`false`) ist die Transaktion eine einmalige Zahlung. Das bedeutet, dass die personenbezogenen Daten nach maximal 3 Monaten gelöscht werden.

Bei wiederkehrenden Zahlungen (Recurring) oder Ratenzahlungen (Installment) wird automatisch der längste Speicherzeitraum gewählt, der für den Account erlaubt ist.

Wenn der Parameter gesetzt wurde, überschreibt er die entsprechenden Einstellungen der verwendeten Anwendung.

Parameter für die Integration in Shop-Systeme

Mit diesen Parametern können Sie ipayment mitteilen, welches Shop-System Sie einsetzen. Setzen Sie diese Parameter nur, wenn Sie ein Shop-System für verschiedene Händler betreiben.

Name und Version des verwendeten Shop-Systems

CGI-Name:	`client_name, client_version`
Webservice-Name:	`OptionData/client/clientName und OptionData/client/clientVersion`
Datentyp:	`String`

Mithilfe dieser Parameter können Sie den Namen Ihres Shop-Systems und die aktuelle Versionsnummer übermitteln. Anhand dieser Angaben kann der ipayment-Support bei einer Anfrage feststellen, von welcher Software die Transaktion gestartet wurde.

Tragen Sie im Parameter `client_name` Name und Version des verwendeten Shop-Systems ein. Im Parameter `client_version` können Sie die Version des Moduls für das ipayment-System angeben.

Weitere Parameter

Weitere Parameter für allgemeine Einstellungen finden Sie in diesem Abschnitt.

IP des Käufers

CGI-Name:	`from_ip`
Webservice-Name:	`OptionData/fromIp`
Datentyp:	`String, maximale Länge 15 Zeichen und Format wie IP`

Für Transaktionen, die aus einer Online-Anwendung stammen, muss die IP des Käufers übermittelt werden. Diese IP wird im ipayment-System gespeichert. Transaktionen aus einer Online-Anwendung können sein: **preauth**, **auth**, **check_save** oder **base_check**.

Bei Aktionen, die auf einer Transaktion basieren, die sich bereits im System befindet, muss die IP nicht angegeben werden. Mögliche Transaktionstypen: **re_auth**, **re_preauth**, **capture**, **reverse**, **refund_cap**, **grefund_cap**, **voice_auth** und **voice_grefund_cap**.

Es kann immer nur eine Anfrage von einer IP zur selben Zeit gestellt werden. Weitere Anfragen müssen warten. Im **Gateway-Modus** oder im **SOAP-Webservice** kann dieses Verhalten zu Problemen führen, weil mehrere Anfragen von einer Anwendung gleichzeitig bei ipayment eingehen können.

Sprache der Fehlermeldungen

CGI-Name:	`error_lang`
Webservice-Name:	`OptionData/errorLang`
Datentyp:	`String, Werte siehe Beschreibung`

Die Fehlermeldung wird in der eingestellten Sprache zurückgegeben. Möglich sind zur Zeit die Werte `de` (Deutsch) oder `en` (Englisch). Wenn Sie diesen Parameter setzen, wird die Einstellung der Anwendung ignoriert.

Security-Hash zur Absicherung der Aufruf-Parameter

CGI-Name:	`trx_securityhash`
Webservice-Name:	`- (nicht benötigt)`
Datentyp:	`String, maximal 32 Zeichen`

Security-Hash des CGI-Aufrufs. MD5-Hash einiger Übergabeparameter zusammen mit dem Transaktions-Security-Key, der im ipayment-Konfigurationsmenü unter **Anwendung** festgelegt wird. Der Hash wird über die Felder `trxuser_id`, `trx_amount`, `trx_currency`, `trxpassword` und dem Transaktions-Security-Key dieser Anwendung generiert. Die Übergabe erfolgt als String ohne Leer- und sonstige Trennzeichen. Somit ist eine Manipulation der Aufrufparameter nicht möglich. Sie können den MD5-Hash auf der Seite https://ipayment.de/technik/seckeygenerator.php4 generieren lassen.

Einen noch besseren Schutz vor Manipulationen erhalten Sie, wenn Sie eine Session vorgenerieren. Mehr dazu lesen Sie im Kapitel *Session-IDs vorgenerieren* auf Seite 42.

Soll der CGI-Silent-Modus benutzt werden?

CGI-Name:	`silent`
Webservice-Name:	`- (nicht benötigt)`
Datentyp:	`Boolean`

Mit diesem Parameter können Sie den **Silent-Modus** aktivieren. Wert 1 (`true`) bedeutet, dass der **Silent-Modus** aktiv ist. Weitere Informationen zum **Silent-Modus** lesen Sie unter *Zahlungen per CGI im Silent-Modus* auf Seite 21.

Soll der CGI-Gateway-Modus benutzt werden?

CGI-Name:	`gateway`
Webservice-Name:	`- (nicht benötigt)`
Datentyp:	`Boolean`

Mit diesem Parameter aktivieren Sie den **Gateway-Modus**. Wert 1 (`true`) bedeutet, dass der **Gateway-Modus** aktiv ist. Weitere Informationen zum **Gateway-Modus** lesen Sie unter *Zahlungen per CGI im Gateway-Modus* auf Seite 23.

Session-ID einer vorgenerierten Session

CGI-Name:	`ipayment_session_id`
Webservice-Name:	`- (nicht benötigt)`
Datentyp:	`String`

Wenn eine Session-ID im **Normalen Modus** oder **Silent-Modus** übermittelt wird, muss diese ID vorher generiert und zurückgegeben worden sein. Wie das funktioniert, können Sie im Kapitel *Session-IDs vorgenerieren* auf Seite 42 nachlesen.

Zahlungsdaten

Parameter für Kredit- und Debitkartenzahlungen

Parameter für die Abwicklung von Zahlungen mit Kreditkarten oder Debitkarten. Sie können diese Parameter im Simulationsmodus testen. Verwenden Sie dazu die Testkreditkartennummern der einzelnen Kartentypen, die Sie unter https://ipayment.de/ **> Technik** finden. Weitere Informationen zum Simulationsmodus finden Sie im Kapitel *ipayment-Funktionen testen (Simulationsmodus)* auf Seite 9.

Nummer der Kreditkarte oder Debitkarte

CGI-Name:	`cc_number`
Webservice-Name:	`PaymentData/ccData/ccNumber`
Datentyp:	`String`

Dieses Feld ist relativ fehlertolerant und akzeptiert auch Werte mit Leerzeichen oder anderen Zeichen. Vor der Zahlungsabwicklung werden alle Zeichen entfernt, die nicht in den Ziffernbreich von 0-9 fallen. Kreditkartennummern bestehen je nach Kartentyp aus 13-19 Stellen.

Verfalls- bzw. Gültigkeitsdatum der Kreditkarte oder Debitkarte

CGI-Name:	`cc_expdate_month, cc_expdate_year`
Webservice-Name:	`PaymentData/ccData/ccExpdateMonth, PaymentData/ccData/ccExpdateYear`
Datentyp:	`Integer`

Mit diesen Parametern können Sie das Gültigkeitsdatum der Kreditkarte (Monat und Jahr) übergeben. Beide Werte sind maximal zweistellig, gültige Werte liegen im Bereich von 1 bis 12 für Monatsangaben und 0 bis 99 für Jahresangaben. Eine Kreditkarte kann bis zu 20 Jahre gültig sein.

Kartenprüfnummer der Kreditkarte oder Debitkarte

CGI-Name:	`cc_checkcode`
Webservice-Name:	`PaymentData/ccData/ccCheckcode`
Datentyp:	`Integer`

Die Kartenprüfnummer für Mastercard, VisaCard, American Express, Diners Club, Discover und teilweise Maestro. Die Kartenprüfnummer wird manchmal auch als CVC2-Code oder CVV2-Code bezeichnet. Sie finden die Nummer je nach Kartentyp entweder auf der Kartenrückseite im Unterschriftsfeld oder auf der Vorderseite der Karte (bei American Express). Bei American Express hat die Kartenprüfnummer vier Stellen, bei allen anderen Kartentypen ist sie dreistellig.

Kartenprüfnummer nicht speichern

Der Wert der Kartenprüfnummer darf unter keinen Umständen in Ihrer Web-Anwendung gespeichert werden!

Ausgabe- bzw. Startdatum der Kreditkarte oder Debitkarte

CGI-Name:	`cc_startdate_month, cc_startdate_year`
Webservice-Name:	`PaymentData/ccData/ccStartdateMonth,` `PaymentData/ccData/ccStartdateYear`
Datentyp:	`Integer`

Ausgabedatum der Karte (Monat und Jahr). Dieses Feld ist nur bei englischen Solo-Karten und teilweise bei lokalen britischen Maestro-Karten nötig. Beide Werte sind maximal zweistellig, gültige Werte liegen im Bereich von 1 bis 12 für Monatsangaben und 0 bis 99 für Jahresangaben.

Issue-Nummer der Kreditkarte oder Debitkarte

CGI-Name:	`cc_issuenumber`
Webservice-Name:	`PaymentData/ccData/ccIssuenumber`
Datentyp:	`String, maximal 2 Zeichen`

Issue-Nummer der Karte. Dieses Feld ist nur bei englischen Maestro- oder Solo-Karten nötig.

Alle Werte übergeben

Die Issue-Nummer kann eine führende Null enthalten. Diese führende Null muss unbedingt auch übergeben werden.

Typ der Kreditkarte oder Debitkarte

CGI-Name:	`cc_typ`
Webservice-Name:	`- (Übergabe per OptionData/otherOptions möglich)`
Datentyp:	`String, Werte siehe Beschreibung`

Wenn Ihnen der Kreditkartentyp bekannt ist, können Sie diesen an ipayment übermitteln. Bei einer Angabe wird geprüft, ob die übergebene Karte tatsächlich von diesem Typ ist. Eine Transaktion wird nur dann durchgeführt, wenn der Kartentyp korrekt erkannt wird und mit der Angabe übereinstimmt.

Erlaubte Werte sind: MasterCard, VisaCard, AmexCard, DinersClubCard, JCBCard, SoloCard, DiscoverCard, MaestroCard.

Kartentyp-Fehler ignorieren

CGI-Name:	`ignore_cc_typ_mismatch`
Webservice-Name:	`- (Übergabe per OptionData/otherOptions möglich)`
Datentyp:	`Boolean`

Wenn Sie den Kartentyp im Parameter `cc_typ` übergeben haben und aufgrund der Kartennummer ein anderer Kartentyp erkannt wird, meldet die Transaktionsabwicklung einen Fehler. Sie können dies verhindern, indem Sie die Angabe des Kartentyps ignorieren. Übergeben Sie dazu den Parameter `ignore_cc_typ_mismatch` auf den Wert `1`.

Wenn Sie die Fehler bei der Angabe des Kartentyps ignorieren, können Sie einfach ein fehlertolerantes Kreditkartendaten-Formular anbieten.

Autorisierungsnummer für telefonische Autorisierungen oder Gutschriften

CGI-Name:	`cc_voice_authcode`
Webservice-Name:	`voiceAuthcode (in Methoden als Parameter, wenn nötig)`
Datentyp:	`String`

Bei telefonischen Autorisierungen (Transaktionstypen "voice_auth" und "voice_grefund_cap") können Sie über diesen Parameter den Autorisierungscode angeben. Rufen Sie den Genehmigungsdienst des Karteninstitutes an, um den Autorisierungscode in Erfahrung zu bringen.

Parameter für ELV-Zahlungen

Zur Abwicklung von Zahlungen mit dem elektronischen Lastschriftverfahren stehen die folgenden Parameter zur Verfügung:

BLZ der Bank

CGI-Name:	`bank_code`
Webservice-Name:	`PaymentData/elvData/bankCode`
Datentyp:	`String`

Bankleitzahl oder „sort code“ der Bank. Die Länge des Feldes und die erlaubten Inhalte sind vom Bankenland abhängig. Für bestimmte Länder, in denen es keine Bankleitzahl gibt, kann der Parameter auch leer bleiben oder gar nicht angegeben werden. Wenn die Bankleitzahl mit Leerzeichen angegeben wird, werden diese Leerzeichen automatisch entfernt.

Kontonummer der Bank

CGI-Name:	`bank_accountnumber`
Webservice-Name:	`PaymentData/elvData/bankAccountnumber`
Datentyp:	`String`

Kontonummer. Die Länge des Feldes und die erlaubten Inhalte sind vom Bankenland abhängig. Wenn die Kontonummer mit Leerzeichen angegeben wird, werden diese Leerzeichen automatisch entfernt.

Land der Bankverbindung

CGI-Name:	`bank_country`
Webservice-Name:	`PaymentData/elvData/bankCountry`
Datentyp:	`String, 2-stelliger ISO-Ländercode`

ISO-Country-Code des Bankenlands. Wenn dieser Parameter nicht gesetzt ist, wird das Land aus der Adresse verwendet. Ansonsten gilt „DE“ (Deutschland) als Standardland.

Name der Bank

CGI-Name:	bank_name
Webservice-Name:	PaymentData/elvData/bankName
Datentyp:	String

Name der Bank. ipayment versucht zusätzlich, den Banknamen anhand der BLZ zu ermitteln. Der Wert, der über diesen Parameter übermittelt wird, wird überschrieben, wenn ipayment den Namen erfolgreich ermitteln kann.

IBAN der Bank

CGI-Name:	bank_iban
Webservice-Name:	PaymentData/elvData/bankIban
Datentyp:	String

IBAN (International Bank Account Number) der Bankverbindung. Wenn dieser Parameter angegeben ist, können die Bankleitzahl, das Bankenland und die Bankkontonummer weggelassen werden. Diese Daten werden dann automatisch aus der IBAN ermittelt.

BIC der Bank

CGI-Name:	bank_bic
Webservice-Name:	PaymentData/elvData/bankBic
Datentyp:	String

Wenn eine IBAN verwendet wird, können Sie auch die BIC (Bank Interchange Code) angeben.

Parameter für Prepaid-Zahlungen

Die Abwicklung von Prepaid-Zahlungen ist nur über die CGI-Integrationsmethoden **Normaler Modus** und **Silent-Modus** möglich.

Business-Type-Parameter für paysafecard-Zahlungen

CGI-Name:	pp_paysafecard_businesstype
Webservice-Name:	- (nicht unterstützt)
Datentyp:	String

Businesstyp für Paysafecard-Zahlungen. Der Wert wird von Paysafecard im Rahmen des Akzeptanzvertrags festgelegt. Wenn Sie keinen Wert angeben, wird automatisch der Wert „0“ eingetragen.

Reporting-Criteria-Parameter für paysafecard-Zahlungen

CGI-Name:	pp_paysafecard_reportingcriteria
Webservice-Name:	- (nicht unterstützt)
Datentyp:	String

Hier wird das Reporting-Kriterium für Paysafecard-Zahlungen angegeben. Der zu verwendende Wert wird von Paysafecard im Rahmen des Akzeptanzvertrages festgelegt.

Gesicherte Rückmeldung erfolgreicher Transaktionen

Bei der Verwendung der CGI-Modi **Normaler Modus** oder **Silent-Modus** wird der Browser Ihres Kunden zum Ausführen der Transaktion direkt auf den ipayment-Server weitergeleitet. Damit die Bestellung durchgeführt werden kann, muss der Rücksprung auf Ihren Shop-Server funktionieren. Ansonsten wird zwar die Zahlung erfolgreich abgewickelt, aber Ihre Webanwendung erfährt davon nichts.

Für den Rücksprung können Sie Links oder Buttons verwenden, die Ihr Kunde aktiv anklicken muss (**Normaler Modus**). Dabei besteht jedoch die Gefahr, dass der Rücksprung nicht funkti-

oniert, zum Beispiel weil Ihr Kunde den Link oder Button nicht anklickt. Eine Alternative ist eine HTTP-Weiterleitung, die automatisch vom Browser ausgeführt wird (**Silent-Modus**). Eine erfolgreiche Weiterleitung kann auch bei diesem Verfahren nicht garantiert werden, weil die Umsetzung vom Browser des Kunden abhängig ist.

Damit die Abwicklung trotzdem reibungslos abläuft, bietet ipayment einen so genannten „Hidden Trigger" an. Sie übergeben die URL eines CGI-Scriptes (Hidden-Trigger-Script) an ipayment. Diese URL wird sofort nach der Abwicklung der Transaktion und vor dem Rücksprung in den Shop automatisch aufgerufen. Dadurch werden alle wichtigen Informationen wieder an Ihren Shop übermittelt. Damit die Bestellung auch bearbeitet wird, sollte Ihr Shop-System die Bestellung auf Basis dieses Hidden-Trigger-Scripts ausführen oder mindestens prüfen, ob das Hidden-Trigger-Script vor dem Rücksprung aufgerufen wurde.

Aus Sicherheitsgründen empfehlen wir zusätzlich zu überprüfen, ob der Aufruf der Hidden-Trigger-URL auch tatsächlich vom ipayment-Server kam. Nur in diesem Fall ist der Aufruf echt und eine Zahlung erfolgt. Die Prüfung können Sie anhand des Hostnamens vornehmen, der auf .ipayment.de enden muss. Außerdem können Sie die IP-Adressen des Servers für die Prüfung verwenden. Die offiziellen IP-Adressen der Server, von denen die Aufrufe kommen können, sind:

- 212.227.34.218
- 212.227.34.219
- 212.227.34.220

Wenn Sie den **Gateway-Modus** oder den **SOAP-Webservice** einsetzen, findet die Kommunikation direkt statt. Die Rückmeldung ist somit automatisch gesichert und das Hidden-Trigger-Script wird nicht benötigt. Hierzu müssen allerdings die Timeout-Einstellungen ausreichend hoch eingestellt werden (mindestens 5 bis 10 Minuten).

Parameter für die gesicherte Rückmeldung

Durch diese Parameter können Sie dem ipayment-System die URLs der Scripte im Shop-System übergeben, die bei erfolgreichen Zahlungen aufgerufen werden.

Angabe der Hidden-Trigger-URLs

CGI-Name:	`hidden_trigger_url[x]`
Webservice-Name:	`- (nicht benötigt)`
Datentyp:	`String`

Sie können mehrere Hidden-Trigger-URLs angeben. Diese URLs werden nach erfolgter Zahlung direkt vom ipayment-Server per HTTP-POST aufgerufen. Geben Sie die erste URL mit dem Parameter `hidden_trigger_url` an. Jede weitere URL können Sie als `hidden_trigger_url[x]` übermitteln, wobei `[x]` für eine Zahl steht. Geben Sie die Zahlen ab 1 aufsteigend an. Zusätzlich können Sie im ipayment-Konfigurationsmenü unter **Anwendungen** für jede Anwendung eine weitere URL hinterlegen. Diese URL wird immer als erstes aufgerufen.

Die angegebene URL muss per HTTP oder HTTPS über die Standard-Webserver-Ports 80 und 443 erreichbar sein. Andere Ports können aus Sicherheitsgründen nicht aufgerufen werden.

Session-IDs vorgenerieren

Im **Normalen Modus** und im **Silent-Modus** müssen gewisse feststehende Parameter an ipayment übermittelt werden. Das sind zum Beispiel Ihre ipayment Accountdaten, die Account-ID, die Anwendungs-ID und das Anwendungspasswort. Zusätzlich müssen Betrag und Währung übermittelt werden.

Ihr Kunde kann in seinem Browser den Link oder das Formular einsehen, das an ipayment geschickt wird. Dadurch kann er auch diese Daten manipulieren. Um solche Manipulationsversuche zu verhindern, können Sie diese Daten bereits an ipayment übermitteln, bevor Ihr Kunde auf den ipayment-Server weitergeleitet wird. Mit diesen Daten wird eine Session angelegt. Die Session-ID wird an Sie übermittelt und im Formular oder Link an ipayment für die Transaktion verwendet. Die Session wird mittels der Webservice-Funktion `createSession` generiert. Zur Generierung der Session-ID benötigen Sie die Parameter `accountData`, `transactionData`, `transactionType`, `paymentType`, `options` und `processorUrls`. Weitere Informationen zum Webservice finden Sie im Kapitel *Integration per SOAP-Webservice* auf Seite 24. Ein Beispiel eines Aufrufs können Sie im Abschnitt *Vorgenerierte Session zur besseren Absicherung gegen Manipulationen* ab Seite 75 einsehen.

Parameter für die Nutzung einer vorgenerierten Session

Eine per Webservice generierte Session-ID können Sie im **Normalen Modus** oder im **Silent-Modus** nutzen.

Angabe der Session-ID

CGI-Name:	`ipayment_session_id`
Webservice-Name:	`- (nicht benötigt)`
Datentyp:	`String`

Mit diesem Parameter übergeben Sie eine vorgenerierte Session-ID für den **Normalen Modus** oder den **Silent-Modus**.

Wenn dieser Parameter beim Aufruf übergeben wird, werden alle Parameter ignoriert, die bereits in der Session gesetzt sind. Eine nachträgliche Manipulation der Werte von außen ist nicht möglich. Die Session wird nach einem einmaligen Aufruf ungültig. Wenn Ihr Kunde also das Formular im **Silent-Modus** mit fehlerhaften Daten abschickt, wird die Transaktion von ipayment abgelehnt und Ihr Kunde wieder in Ihre Web-Anwendung weitergeleitet. Damit das Abschicken funktioniert, muss zuerst eine neue Session vorgeneriert werden.

Wenn Sie eine vorgenerierte Session verwenden, sollte bei der Anwendung das Feld „Transaktions-Security-Key“ freigelassen werden.

Überprüfung des Karteninhabers mit 3-D Secure

Die 3-D-Secure-Sicherheitsverfahren „Verified by Visa“, „Mastercard SecureCode“ und „Maestro SecureCode“ stellen sicher, dass es sich bei dem Käufer tatsächlich um den Kreditkarteninhaber handelt. Bei Nutzung von 3-D Secure profitieren Sie grundsätzlich von der Haftungsumkehr („Liability Shift“) für mit Visa, Mastercard und Maestro abgewickelte Verkäufe. Das bedeutet, dass für Sie das Rückbelastungsrisiko beim Widerruf von Transaktionen (Chargeback) aus Missbrauchsgründen entfällt.

Weitere Informationen zu den Sicherheitsverfahren finden Sie unter https://ipayment.de > **Sicherheit**.

"Verified by Visa" und "MasterCard/Maestro SecureCode"

Beide Verfahren beschreiben eine Authentifizierung des Kreditkarteninhabers zur Bezahlung und bieten dafür als Standardlösung ein einfaches Passwort an. Der Kreditkarteninhaber vergibt dieses bei seiner kartenausgebenden Bank und fügt einen eigenen Text hinzu, der bei der Passworteingabe angezeigt wird. Die Eingabe des Passwortes findet nicht auf den Seiten des Händlers statt, sondern auf den Seiten der kartenausgebenden Bank. Dadurch ist die Sicherheit des Passworts gewährleistet. Außerdem erhält der Karteninhaber durch den zusätzlichen

Text die Sicherheit, dass er sich auf der Seite seiner Bank befindet. Wenn zu einer Kreditkarte kein Passwort vergeben wurde, wird die Transaktion wie bisher sofort ausgeführt.

Bei Visa und MasterCard ist die Nutzung der Sicherheitsverfahren grundsätzlich optional. Bei internationalen Maestrozahlungen im Internet muss zwingend Maestro SecureCode verwendet werden. Beachten Sie hierzu auch die Vereinbarungen in Ihrem Akzeptanzvertrag und informieren Sie sich bei Ihrem Zahlungsanbieter.

In den nachfolgenden Abschnitten erhalten Sie technische Informationen zur Authentifizierung und zur Funktionsweise der Transaktion unter Verwendung der Sicherheitsverfahren.

Ablauf einer Zahlung mit Authentifizierung per 3-D Secure

Das Herzstück von 3-D-Secure-Transaktionen ist das so genannte MPI (Merchant Plugin). Das MPI ist bereits in ipayment integriert und kommt bei 3-D-Secure-Transaktionen automatisch zum Einsatz. Bei 3-D Secure besteht die Authentifizierung aus zwei Schritten:

Als Erstes wird die Anfrage mit der Kreditkartennummer an das MPI weitergeleitet. Das MPI fragt bei einem Visa- bzw. MasterCard-Directoryserver nach, ob diese Kreditkarte durch die kartenausgebende Bank für 3-D Secure freigeschaltet wurde. Freigeschaltet ist die Kreditkarte zum Beispiel dann, wenn ein Passwort vergeben wurde.

Wenn das System meldet, dass die Kreditkarte nicht für 3-D Secure freigeschaltet ist, findet die Zahlungsabwicklung wie bisher sofort statt. Diese Transaktion wird speziell gekennzeichnet. Die Kennzeichnung sagt aus, dass das 3-D-Secure-Verfahren vom Händler angeboten wurde, aber vom Karteninhaber nicht verwendet werden konnte.

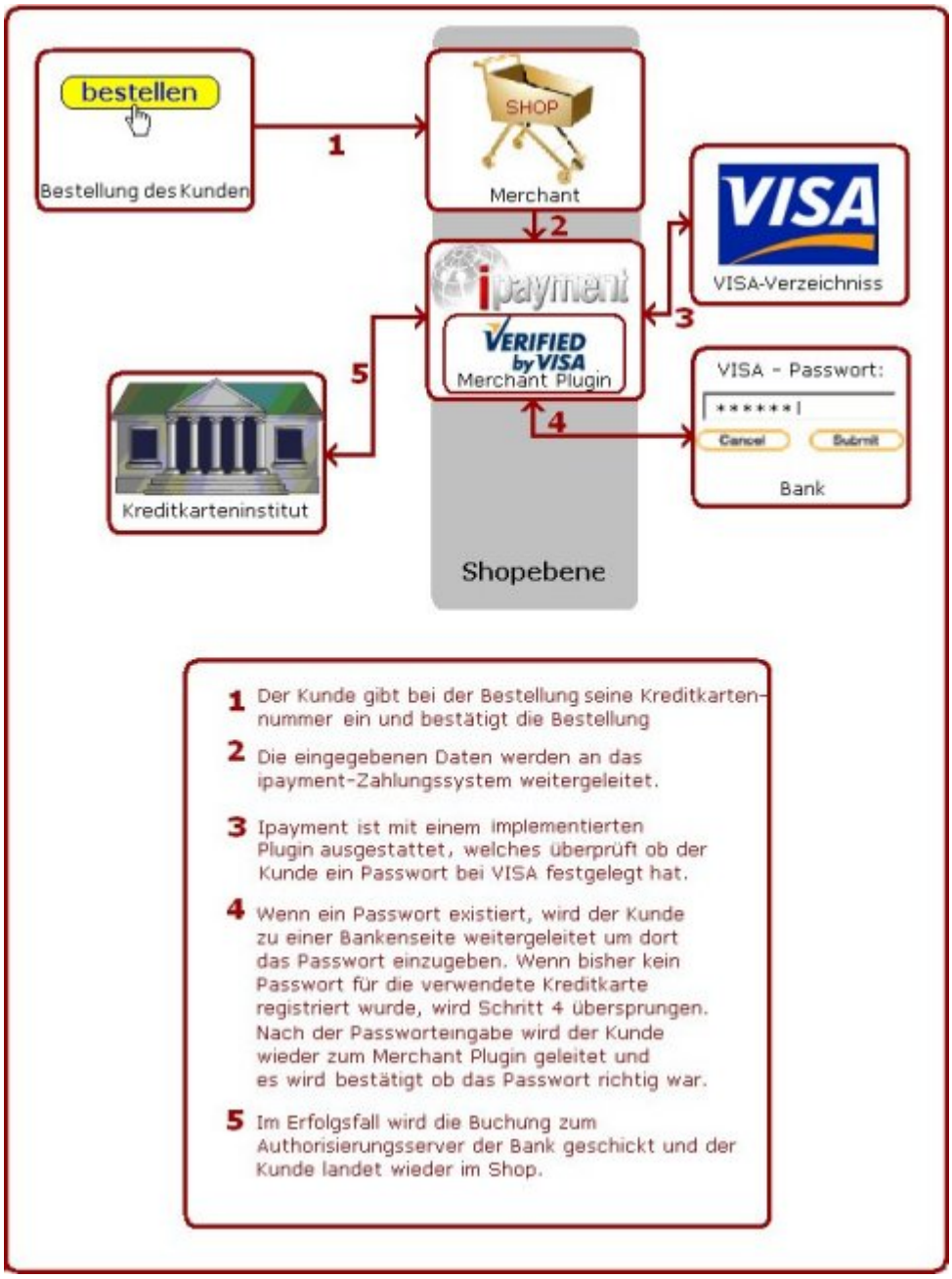
Wenn die Kreditkarte für 3-D Secure freigeschaltet ist, erhält das MPI zusätzlich vom Visa- bzw. MasterCard-Directoryserver die URL des Authentifizierungsscripts der entsprechenden kartenausgebenden Bank. Daraufhin wird der Käufer auf die Website seiner Bank weitergeleitet. Hier werden das Banklogo, der Betrag, der Händlername, die Kreditkartennummer, der gewählte Karteninhaber-Text und ein Feld für die Passworteingabe dargestellt. Nach der Eingabe des Passwortes wird dieses geprüft und im Fehlerfall nochmals abgefragt. Neben einem einfachen Passwort gibt es noch weitere Verfahren zur Authentifizierung, die von den Banken eingesetzt werden können. Am Ende kehrt der Käufer zu einer vorher definierten Seite im Shop oder im Zahlungssystem zurück, unabhängig davon, ob die Authentifizierung erfolgreich war oder nicht.

Mit der Antwort wird im Erfolgsfall noch der so genannte CAVV (Card Authentication Verification Value) mitgegeben, der in der nun folgenden Abwicklung der Transaktion mit zum entsprechenden Acquirer gesendet wird. Dieser CAVV-Wert wird von der Bank gemeinsam mit weiteren Antwort-Daten verschlüsselt an ipayment übergeben. Im Fall einer fehlerhaften Authentifizierung wird die Transaktion sofort vom Zahlungssystem abgelehnt. Eine fehlerhafte Authentifizierung kann zum Beispiel auftreten, wenn das Passwort mehrfach falsch eingegeben wurde oder wenn der Käufer den Prozess abbricht.

Bei der Prüfung nach der korrekten Freischaltung der Kreditkarte oder des korrekten Passworts kann es vorkommen, dass die Systeme von Visa und Mastercard melden, dass die Prüfung nicht möglich ist. Dafür kann es die folgenden Gründe geben: Technische Fehler in den Systemen von Visa, Mastercard oder den Banken oder eine fehlende Haftungsumkehr der benutzten Kreditkarte. In diesem Fall wird die Transaktion durch ipayment trotzdem weiter verarbeitet, aber nicht mit einer 3-D-Secure-Kennzeichnung versehen. Für diese Transaktionen gibt es somit keine Haftungsumkehr. In diesem Fall wird der Rückgabe-Parameter `trx_payauth_status` mit dem Wert `U` gefüllt.

Die kompletten technischen Spezifikationen zu 3-D Secure v1.0.2 können Sie auf der Visa-Website unter http://partnernetwork.visa.com/pf/3dsec/main.jsp aufrufen und herunterladen.

Im Folgenden sehen Sie noch einmal den Ablauf der 3-D-Secure-Transaktion in einem Schaubild am Beispiel von „Verified by Visa“:



Das müssen Sie tun, wenn Sie 3-D Secure verwenden

Mastercard und Visa stellen auf Ihren Webseiten Informationen zum technischen Ablauf der Sicherheitsverfahren bereit, unter anderem auch die sogenannten „Merchant Guidelines“ (Richtlinien für Händler). Diese Richtlinien müssen Sie auf jeden Fall beachten.

Die Informationen finden Sie hier:

- Visa: http://www.visaeurope.com/merchant/handlingvisapayments/cardnotpresent/verifiedbyvisa.jsp
- MasterCard/Maestro: http://www.mastercardmerchant.com/securecode/getstarted.html

Sobald Sie eines der Sicherheitsverfahren anbieten, sind Sie verpflichtet, das jeweilige Verfahrens-Logo auf Ihrer Webseite gut sichtbar anzuzeigen. Die genaue Positionierung der/des Logos können Sie den jeweiligen Händler-Richtlinien entnehmen.

Hier finden Sie die Logos zu den Sicherheitsverfahren:

- MasterCard/Maestro SecureCode: http://www.1und1.info/downloads/sc_infologo.zip
- Verified by Visa: http://www.1und1.info/downloads/vbv_infologo.zip

Integration in eigene Shopsysteme

Sie haben ipayment in eine eigene Shop-Lösung integriert? Wenn Sie den **Gateway CGI-Modus** oder den **SOAP-Webservice** verwenden, müssen Sie an Ihrem Shop-System Änderungen vornehmen, damit die Weiterleitung des Käufers auf die Banken-Website funktioniert. Außerdem müssen Sie das sogenannte PIT (Product Integration Testing) bei Visa absolvieren. Durch das PIT führen Sie zusätzlich einen Test über die Kommunikation Ihrer Anwendung mit dem ipayment-Zahlungssystem durch. Im folgenden Abschnitt finden Sie alle Informationen zu dem PIT.

Verified by Visa Product Integration Testing (PIT)

Wenn Sie ein Shopsystem von 1&1 oder ipayment im **Normalen Modus** oder **Silent-Modus** einsetzen, sind die Authentifizierungsverfahren "Verified by Visa", "Mastercard SecureCode" und „Maestro SecureCode“ bereits verfügbar. In diesem Fall müssen Sie keine umfangreichen Anpassungen oder Erweiterungen an Ihrem Shop vornehmen. Aktivieren Sie nur die Zahlungsauthentifizierung für Ihre benutzen Anwendungen. Zusätzlich müssen Sie die entsprechenden Logos der Sicherheitsverfahren in Ihrem Shop anzeigen.

Das PIT (Product Integration Testing) ist eine Anforderung von Visa und stellt sicher, dass nur funktionierende Implementierungen von 3-D Secure bzw. "Verified by Visa" im Einsatz sind. Für das PIT existiert ein Testsystem, das fast genauso funktioniert wie das 3-D-Secure-Livesystem. Anders als beim Livesystem können Sie aber über spezielle Kartennummern verschiedene Antworten des 3-D-Secure-Systems erzwingen und somit die Integration testen.

Jede Implementierung des 3-D-Secure-Sicherheitsverfahrens muss dieses PIT mindestens einmal durchlaufen.

Je nach Integrationsmodus entfallen einige Aufgaben aus dem 3-D-Secure-Ablauf auf Ihre Webanwendung. Diese Aufgaben müssen Sie gesondert testen, damit ein reibungsloser Ablauf sichergestellt werden kann.

Stellen Sie zunächst sicher, dass für die zu testenden Kartentypen die Freischaltung für „Verified by Visa“ aktiviert ist und erfolgreich von Visa abgeschlossen wurde. Nun benötigen Sie eine ipayment-Anwendung, mit der Sie die PIT-Transaktionen durchführen können. Diese Anwendung wird in einen speziellen Testmodus geschaltet, wodurch der Simulationsmodus aktiviert wird. Wir empfehlen deshalb, für das PIT eine neue ipayment-Anwendung anzulegen und diese in Ihrem Shopsystem nur für diese Tests zu benutzen. Beachten Sie auch, dass das PIT für diese Anwendung nicht mehr deaktiviert werden kann.

Unter https://dropit.3dsecure.net/PIT finden Sie die offizielle Webseite für das PIT. Melden Sie sich dort an. Sie benötigen dazu neben allgemeinen Angaben noch einige spezielle Daten, die Sie nach Absenden des Formulars unter https://ipayment.de/technik/vbvpit.php4 erhalten. Die von Ihnen hier angegebene Anwendung wird mit dem Erzeugen der PIT-Anmeldedaten sofort in den beschriebenen PIT-Testmodus versetzt und kann danach ausschließlich für diese Tests verwendet werden.

Das PIT besteht aus Pflicht-Testfällen, die absolviert werden müssen, und einigen optionalen Testfällen. Es gibt im PIT-Menü bei Visa eine Option, über die Sie alle benötigten Testfälle und deren bisherige Ergebnisse ansehen können („Review Required Results“). Wählen Sie dort im Feld "Mandatory By:" den Eintrag "E.U.", um die benötigten Testfälle für die Visa-EU-Region zu erhalten. Die exakten Beschreibungen der Testfälle mit den zu verwendenden Kartendaten und dem erwarteten Ergebnis sehen Sie im Testplan, der ebenfalls auf der PIT-Webseite zur Verfügung steht.

Jeder Testfall basiert auf einer Transaktion mit einer bestimmten Kartennummer. Diese Kartennummer löst die entsprechenden Aktionen aus. Nachdem Sie sich zum PIT angemeldet und Ihren Shop und die ipayment-Anwendung eingestellt haben, müssen Sie In Ihrem Shop eine Reihe von Bestellungen mit bestimmten Kartennummern durchführen. Die Werte für das Ablaufdatum und die Kartenprüfnummer sind hierbei egal, müssen aber angegeben werden. Das Ablaufdatum muss in der Zukunft liegen. Nach jedem Test können Sie im PIT-Menüpunkt "Review Test Activity" die Testergebnisse prüfen. Wenn der Test bestanden wurde, können Sie den nächsten absolvieren, ansonsten können Sie bei den Test-Details die Gründe für das Scheitern nachlesen. Sollte der PIT-Testfall zum Beispiel eine Ablehnung der Transaktion als Ergebnis erwarten, dann prüfen Sie unbedingt zusätzlich, ob diese Ablehnung vom ipayment-System zurückgegeben wurde. Wenn dies nicht der Fall ist, liegt ein Fehler bei der Abwicklung vor. Prüfen Sie auch im Erfolgsfall den zurückgegebenen 3-D-Secure-Status der Transaktion.

Der Test ist erfolgreich beendet, wenn alle benötigten Testfälle erfolgreich absolviert wurden. Um den Abschluss an Visa zu melden, nutzen Sie den Link "Conclude Testing" im PIT-Tool. Teilen Sie uns außerdem den Abschluss des Tests per E-Mail an support@ipayment.de mit und geben Sie in dieser E-Mail Ihre Account-ID an.

Parameter für 3-D Secure

Wenn Sie für eine Transaktion 3-D Secure verwenden möchten, wird Ihr Kunde zu einer Bankenseite weitergeleitet. Diese Bankenseite öffnet sich direkt im Browser Ihres Kunden. Im **Gateway-Modus** und dem SOAP-Webservice ist das jedoch nicht direkt möglich, da die Kommunikation per Script im Hintergrund stattfindet. Damit die Weiterleitung auch dort korrekt funktioniert, ist eine Erweiterung der bisherigen Abläufe notwendig.

Sie benötigen einige Daten des Browsers Ihres Kunden, die Sie bei der Anfrage an ipayment übergeben müssen. Verwenden Sie dazu die folgenden Parameter:

Verwendeter Browser des Käufers

CGI-Name:	`browser_user_agent`
Webservice-Name:	`OptionData/browser/browserUserAgent`
Datentyp:	`String, Länge bis zu 255 Zeichen`

In diesem Parameter muss der Wert der Environment-Variable `HTTP_USER_AGENT` des Kundenbrowsers an das ipayment-System übergeben werden.

Accept-Header des verwendeten Browsers

CGI-Name:	`browser_accept_headers`
Webservice-Name:	`OptionData/browser/browserAcceptHeaders`
Datentyp:	`String, Länge bis zu 2048 Zeichen`

In diesem Parameter muss der Wert der Environment-Variable `HTTP_ACCEPT` des Kundenbrowsers an das ipayment-System übergeben werden.

Zusätzliche Ergebnisparameter für 3-D Secure

Im **Gateway-Modus** und bei Nutzung des **SOAP-Webservices** gibt ipayment beim Aufruf einen Status und die Ergebnisparameter zurück. Auch wenn ein Redirect aufgrund von 3-D Secure stattfinden soll, wird als Fehlercode die „0“ (erfolgreiche Transaktion) zurückgegeben. Zusätzlich werden folgende Parameter zurückgegeben:

Soll ein Redirect ausgeführt werden?

CGI-Name:	`redirect_needed`
Webservice-Name:	`Wert “REDIRECT” im Feld PaymentReturn/status`
Datentyp:	`Boolean bei CGI, Status-Wert bei Webservice`

Dieser Parameter ist gesetzt, wenn die Zahlung nicht abgewickelt werden konnte, weil ein Redirect nötig ist.

Im **Gateway-Modus** hat der Parameter die Werte "0" oder "1", je nachdem ob ein Redirect stattfinden soll. Bei 3-D Secure-Zahlungen steht der Wert auf 1, weil eine Passwortprüfung notwendig ist.

Bei Nutzung des SOAP-Webservices wird die Durchführung eines Redirects durch den Wert `REDIRECT` im Status-Feld des Transaktionsergebnisses angezeigt.

Redirect-Daten

CGI-Name:	`redirect_data`
Webservice-Name:	`PaymentReturn/redirectDetails/redirectData`
Datentyp:	`String`

Dieses Feld enthält ein komplettes HTML-Formular. Wie Sie dieses Formular behandeln müssen, hängt von der angegebenen Aktion (`redirect_action`) ab. Das Formular enthält den Platzhalter `%REDIRECT_RETURN_SCRIPT%`, den Sie durch die URL zu einem eigenen Script ersetzen müssen. Dieses Script wird dann nach der Authentifizierung von der Bank mit bestimmten Parametern aufgerufen, die in einem zweiten CGI-Gateway bzw. Webservice-Aufruf an ipayment zur Prüfung und Zahlungsabwicklung übergeben werden müssen.

Die URL, die Sie einfügen, können Sie durch beliebige eigene Parameter ergänzen, wie zum Beispiel die Session-ID. Dazu müssen Sie zusätzlich zum Platzhalter auch das Fragezeichen nach dem Platzhalter ersetzen. Abgesehen vom Platzhalter darf das Formular nicht verändert werden, da sonst die Weiterverarbeitung durch die Bank nicht erfolgen kann. Ein Beispiel für ein solches Formular finden Sie im Kapitel *Zusätzliche Abläufe für 3-D Secure* auf Seite 49.

Redirect-Aktion

CGI-Name:	redirect_action
Webservice-Name:	PaymentReturn/redirectDetails/redirectAction
Datentyp:	String

Dieser Parameter gibt die durchzuführende Aktion zurück. Derzeit ist nur der folgende Wert möglich: `REDIRECT_POSTFORM`.

Das im Feld `redirect_data` übergebene Formular muss per JavaScript automatisch im gleichen Fenster abgesendet werden. Für den Fall, dass JavaScript deaktiviert ist, ist im Formular ein zusätzlicher Submit-Button verfügbar.

Zusätzliche Abläufe für 3-D Secure

Bei der Nutzung des **Gateway-Modus** und des **SOAP-Webservices** müssen von Ihrem Shopsystem weitere Aktionen ausgeführt werden. Wenn der Parameter `redirect_needed` anzeigt, dass ein Redirect zur Website der Bank notwendig ist, muss Ihr Shopsystem diese Weiterleitung vornehmen. Hierzu benötigen Sie den Parameter `redirect_data`.

Ein Beispiel für ein Formular in Ihrem Shop, in dem die zurückgegebenen Daten integriert werden:

```
<HTML>
  <HEAD>
  </HEAD>
  <BODY>
    <!-- Insert content of parameter redirect_data here -->
    <p>
       Or go <a href='javascript:history.back();'>back to
       the Shop</a>.
       <SCRIPT language="Javascript"><!--
         //auto-submit formular
         document.payauthForm.submit();
       //--></SCRIPT>
    </p>
  </BODY>
</HTML>
```

Durch diese Weiterleitung wird Ihr Kunde zur Website der Bank weitergeleitet. Dort muss er sein 3-D-Secure-Passwort eingeben. Nach erfolgreicher Eingabe oder bei einem Abbruch wird Ihr Kunde wieder in Ihren Shop zurückgeleitet. Dabei werden von der Bank zwei Parameter an das Script Ihres Shops übergeben: `PaRes` und `MD`. Das Script darf die Werte dieser Parameter nicht verändern. Wichtig ist auch, dass die Groß- und Kleinschreibung der Parameternamen beachtet wird. Anschließend führt Ihr Script einen zweiten Aufruf des ipayment-Systems durch, wobei nur diese beiden Parameter übergeben werden. Für den **Gateway-Modus** wird dasselbe Script wie für die anderen Aufrufe verwendet. Beim **SOAP-Webservice** müssen Sie die Methode `paymentAuthenticationReturn` mit diesen Parametern aufrufen.

Wert „MD“

CGI-Name:	MD
Webservice-Name:	ThreeDSecureData/MD
Datentyp:	String

Exakter Wert des Parameters `MD`, der Ihrem Script von der Bank übergeben wurde. Bitte beachten Sie die Groß- und Kleinschreibung des Parameternamens.

Wert „PaRes“

CGI-Name:	`PaRes`
Webservice-Name:	`ThreeDSecureData/PaRes`
Datentyp:	`String`

Exakter Wert des Parameters `PaRes`, der Ihrem Script von der Bank übergeben wurde. Bitte beachten Sie die Groß- und Kleinschreibung des Parameternamens.

Dieser Parameter kann auch spezielle Zeichen enthalten, wie zum Beispiel „+“. Um sicherzustellen, dass diese Werte korrekt an ipayment weitergegeben werden, müssen Sie diese Werte bei Verwendung des **Gateway-Modus** URL-kodiert weitergeben.

Nach der Weitergabe erkennt das ipayment-System die originale Buchung automatisch wieder, prüft alle Daten und führt die Buchung durch. Die zurückgegebenen Ergebnisparameter entsprechen den Parametern, die auch für Zahlungen ohne 3-D Secure verwendet werden.

Storage-Service zum Speichern von Zahlungsdaten

Viele Händler haben früher selbst die Kreditkartendaten Ihrer Kunden gespeichert, um Zahlungen abwickeln zu können. Die neuen Bestimmungen besagen jedoch, dass diese Systeme und Händler nun nach den PCI-DSS-Regeln zertifiziert werden müssen. Dadurch wird eine eigene Speicherung der Daten nahezu unmöglich, bzw. unwirtschaftlich.

Das ipayment-System hat aus diesem Grund einen neuen Storage-Service eingerichtet, der vollständig in das Transaktionssystem integriert ist. Da das ipayment-System nach den PCI-DSS-Regeln zertifiziert ist, können alle Händler, die ipayment nutzen, auch von diesem Service profitieren.

Wie funktioniert der Storage-Service?

Die Zahlungsdaten, die bei einer Anfrage an das ipayment-System benutzt werden, speichert der Storage-Service unter einer eindeutigen ID und gibt sie an Ihren Shop zurück. Für diese Speicherung können Sie eine weitere Referenz-Information (zum Beispiel die Kundennummer Ihres Kunden) und ein Ablaufdatum für die Datenspeicherung angeben.

Sie können den Storage-Service sowohl für eine Langzeit-Speicherung von Zahlungsdaten, als auch für eine kurzzeitige Speicherung verwenden. Eine kurzzeitige Speicherung kann sich zum Beispiel über die Dauer eines Bestellvorgangs im Shop erstrecken. Abgelaufene Zahlungsdaten werden automatisch zum angegebenen Zeitpunkt aus dem ipayment-System entfernt.

Damit die Zahlungsdaten gespeichert werden, müssen Sie den Parameter `use_datastorage` setzen. Bei dieser Anfrage wird der Parameter `storage_id` mit der eindeutigen ID des Datensatzes zurückgegeben.

Bei allen weiteren Anfragen zur Zahlungsabwicklung wird anstelle der Zahlungsdaten einfach die ID des Datensatzes im Parameter `from_datastorage_id` mitgegeben. Das ipayment-System liest anhand dieser ID die Zahlungsdaten aus der Datenbank und verwendet diese für die Zahlung.

Die vorhandenen Adressdaten werden ebenfalls im Storage-Service gespeichert.

Parameter für den Storage-Service

Diese Parameter stehen für die Nutzung des Storage-Services zur Verfügung.

Soll der Storage-Service verwendet werden?

CGI-Name:	use_datastorage
Webservice-Name:	PaymentData/storageData/useDatastorage
Datentyp:	Boolean

Beim Wert 1 (`true`) werden die Zahlungsdaten der aktuellen Anfrage in der Datenbank des Storage-Services gespeichert. Die eindeutige ID des Datensatzes wird im Parameter `storage_id` zurückgegeben.

ID des Storage-Service aus der die Zahlungsdaten gelesen werden sollen

CGI-Name:	from_datastorage_id
Webservice-Name:	PaymentData/storageData/fromDataStorageId
Datentyp:	Long

Dieser Parameter enthält die ID des Storage-Service-Datensatzes, aus dem die Zahlungsdaten für diese Transaktion gelesen werden sollen.

Ablaufdatum der gespeicherten Daten

CGI-Name:	datastorage_expirydate
Webservice-Name:	PaymentData/storageData/datastorageExpirydate
Datentyp:	String

Dieser optionale Parameter gibt das Ablaufdatum der Daten im Storage an. Die Daten werden nach diesem Datum automatisch gelöscht.

Als Wert wird das US-Datumsformat akzeptiert, wie zum Beispiel „2008/09/15“.

Verwendeten Datensatz als ungültig erklären

CGI-Name:	expire_datastorage
Webservice-Name:	PaymentData/storageData/expireDatastorage
Datentyp:	Boolean

Dieser optionale Parameter kann den momentan verwendeten Datensatz als ungültig erklären. Das ist sinnvoll, wenn Sie eine Zahlung durchführen möchten, danach aber diesen Datensatz nicht mehr benötigen. Der Parameter wird nur beachtet, wenn Sie einen gespeicherten Datensatz aus dem Storage-Service verwenden und die ID im Parameter `from_datastorage_id` angeben.

Händler-Referenz für die Datenspeicherung

CGI-Name:	datastorage_reference
Webservice-Name:	PaymentData/storageData/datastorageReference
Datentyp:	String

Diese Referenz kann mit den Zahlungsdaten im Storage-Service gespeichert werden. Der Parameter ist optional. Die Nutzung der Referenz ist je nach Wiederverwendungsmodus (siehe nächster Parameter) sinnvoll.

Wie sollen die Daten im Storage-Service gespeichert werden?

CGI-Name:	datastorage_reuse_method
Webservice-Name:	PaymentData/storageData/datastorageReuseMethod
Datentyp:	Integer

Mit diesem Parameter wird festgelegt, nach welchen Vorschriften die Daten gespeichert oder vorhandene IDs wiederverwendet werden. Der Wert wird als Bit-Maske ausgewertet. Die einzelnen Bits können miteinander kombiniert werden, wobei allerdings nicht alle Kombinationen sinnvoll sind.

Die Bits haben folgende Bedeutung:

- **1. Bit (Wert 1) Aktualisierung einer ID:** Die Daten, die im Storage-Service unter einer eindeutigen ID abgespeichert wurden, werden bei weiteren Anfragen aktualisiert. Das bedeutet, dass auch über mehrere Anfragen hinweg die Storage-ID identisch bleibt und immer aktualisiert wird, unabhängig davon, wie sich die Daten ändern. Dieser Wert wird standardmäßig verwendet wenn der Parameter `datastorage_reuse_method` nicht angegeben ist.
- **2. Bit (Wert 2) Immer neue ID:** Jede Zahlungsanfrage mit einer bestimmten Storage-ID erzeugt immer eine neue ID und gibt diese zurück. Das geschieht auch dann, wenn die Daten identisch sind. Geben Sie bei Nutzung dieser Methode unbedingt das Ablaufdatum der Daten an.
- **3. Bit (Wert 4) Eindeutige ID für komplette Zahlungsdaten:** Das System vergibt eine eindeutige Storage-ID für die Zahlungsdaten. Die Zahlungsdaten sind die vollständigen Kreditkartendaten (ohne Kartenprüfnummer) oder alle Bankdaten. Das bedeutet, dass eine weitere Anfrage mit den gleichen Zahlungsdaten, bei der eine Storage-ID vergeben werden soll, die gleiche Storage-ID erhält, wie die erste Anfrage. In den meisten Fällen können Sie davon ausgehen, dass identische Zahlungsdaten zum selben Kunden gehören.
- **4. Bit (Wert 8) Eindeutige ID für Basis-Zahlungsdaten:** Diese Option ist sehr ähnlich zur vorherigen, es werden jedoch nur die Basis-Zahlungsdaten benutzt. Das sind bei Kreditkartendaten nur die Kartennummer, bei Bankdaten die Bankleitzahl und die Kontonummer.
- **5. Bit (Wert 16) Eindeutige ID für die Adressdaten:** Das System vergibt je nach Adressdaten eine eindeutige Storage-ID. Die gleichen Adressdaten bei einer späteren Anfrage erhalten dieselbe Storage-ID wie bei der ersten Anfrage.
- **6. Bit (Wert 32) Eindeutige ID für die Händler-Referenz:** Das System vergibt je nach Händler-Referenz eine eindeutige Storage-ID. Wenn zum Beispiel eine Shop-Kundennummer als Händler-Referenz verwendet wird, stellt diese Option sicher, dass pro Kundennummer nur ein Datensatz im Storage-Service vorliegt. Dieser Datensatz wird immer aktualisiert.

Beispiele für Bit-Kombinationen:

- `datastorage_reuse_method=24`: Es wird ein neuer Datensatz angelegt, sobald die Adressdaten oder die Basis-Zahlungsdaten aktualisiert werden. Wenn bei der Kreditkarte nur die Gültigkeitsdauer verändert wird, aber die Kartennummer gleich bleibt, wird dabei immer dieselbe Storage-ID verwendet.
- `datastorage_reuse_method=36`: Es wird ein neuer Datensatz angelegt, wenn die kompletten Zahlungsdaten Ihres Kunden geändert werden oder wenn Sie einen neuen Wert im Parameter `datastorage_reference` übermitteln. Auf diese Weise können für einen Shop-Kunden zum Beispiel mehrere Zahlungsdaten gespeichert werden.
- `datastorage_reuse_method=52`: Es wird ein neuer Datensatz angelegt, wenn die Zahlungsdaten, die Adressdaten oder die Händlerreferenz bei der Anfrage nicht genau mit den gespeicherten Daten übereinstimmen. Das ist zum Beispiel der Fall, wenn Sie Ihrem Kunden eine neue Kundennummer geben möchten oder wenn Ihr Kunde seine Adressdaten oder Zahlungsdaten ändert.

Zusätzliche Ergebnisparameter des Storage-Service

Im hier aufgeführten Ergebnis-Parameter wird die Storage-ID zurückgegeben.

Storage-ID

CGI-Name:	`storage_id`
Webservice-Name:	`PaymentReturn/successDetails/retStorageId`
Datentyp:	`Long`

Dieser Parameter enthält die Storage-ID, die vom Storage-Service vergeben wurde.

Regelmäßige Zahlungen und Raten-Zahlungen

Was sind regelmäßige Zahlungen (Recurring Payments)?

Bei regelmäßigen Zahlungen (sogenannte Recurring Payments) handelt es sich meist um längere Vertragsverhältnisse, die nicht durch einmalige Zahlungen abgerechnet werden, sondern in regelmäßigen Abständen (zum Beispiel wöchentlich, monatlich oder halbjährlich) in Rechnung gestellt werden. Eine regelmäßige Zahlung kann zum Beispiel die Grundgebühr für das Mobiltelefon sein.

Regelmäßige Zahlungen bestehen in der Regel aus einer ersten Zahlung, die zum Beispiel als E-Commerce-Transaktion ausgeführt wird, und weiteren Zahlungen in den entsprechenden Abständen.

Vor allem bei Kreditkartenzahlungen ist es wichtig, die Transaktionen korrekt zu kennzeichnen, da in den Regelwerken der Karteninstitute festgelegt ist, dass die erste Zahlung und die Folgezahlungen unterschiedlich zu behandeln sind. Die Folgezahlungen werden dabei speziell gekennzeichnet und von den Banken anders behandelt als die erste Zahlung.

Was sind Ratenzahlungen (Installment Payments)?

Bei Ratenzahlungen handelt es sich um mehrere Zahlungen, denen ein Vertrag bzw. ein Produktkauf zugrunde liegt. Anders als bei der regelmäßigen Zahlung hält Ihr Kunde das Produkt bereits vollständig in den Händen, während er dieses noch abbezahlt (zum Beispiel ein Fernseher für 1000 EUR, der 10 Monate lang mit 105 EUR pro Monat bezahlt wird). Ratenzahlungen setzen sich aus einer Anzahlung und weiteren, festgelegten Zahlungen zusammen. Die Anzahl der Einzelzahlungen ist bereits zu Beginn festgelegt.

Vor allem bei Kreditkartenzahlungen ist es wichtig, die Transaktionen korrekt zu kennzeichnen, da in den Regelwerken der Karteninstitute festgelegt ist, dass die Anzahlung und die Folgezahlungen unterschiedlich zu behandeln sind. Die Folgezahlungen werden dabei speziell gekennzeichnet und von den Banken anders behandelt als die Anzahlung.

Die Parameter für Ratenzahlungen werden in den nachfolgenden Kapiteln genauer erläutert.

Wie werden solche Zahlungen über ipayment abgewickelt?

Ein wichtiges Thema bei regelmäßigen Zahlungen bzw. Ratenzahlungen ist die Sicherheit der Zahlungsdaten. Diese Daten werden für den Zeitraum der Zahlungsabwicklung gespeichert, manchmal für sehr lange Zeit. Wir empfehlen dafür das ipayment-System zu verwenden, da Sie sonst eine PCI-DSS-Sicherheitszertifizierung benötigen.

Bei der Verwendung des ipayment-Zahlungssystems für die Abwicklung solcher wiederkehrenden Zahlungen werden alle sensiblen Kreditkartendaten in den sicheren ipayment-Systemen gespeichert. Sie müssen keine Zahlungsdaten in Ihrer Datenbank speichern. Sie benötigen nur die ipayment-Transaktionsnummer, um regelmäßige Zahlungen oder Ratenzahlungen auszulösen.

So können Sie die Abwicklung von regelmäßigen Zahlungen oder Ratenzahlungen vornehmen:

- Führen Sie die erste Transaktion als Zahlung (**auth** oder **preauth**) oder als Zahlungsdatenprüfung (**check_save**) durch. Hierbei muss die Anfrage als Anzahlung („initiale Zahlung“) gekennzeichnet werden, indem der Parameter `recurring_typ` bzw. `installment_typ` den Wert `initial` erhält. Geben Sie auch die weiteren Parameter an, wie zum Beispiel `recurring_expiry`.

- Speichern Sie die zurückgegebene ipayment-Transaktionsnummer (Rückgabeparameter `ref_trx_number`) der Buchung in Ihrem System. Für den kompletten Zyklus der wiederkehrenden Zahlungen kann diese initiale Transaktionsnummer verwendet werden. Das ipayment-System sucht dann automatisch die Zahlungsdaten von der letzten noch verfügbaren Transaktion des Zyklus und benutzt diese.
- Nehmen Sie die Folgezahlungen mit den Transaktionstypen **re_auth** oder **re_preauth** unter Angabe der Original-Transaktionsnummer oder der Transaktionsnummer der letzten Buchung im Parameter `orig_trx_number` vor. Diese Transaktionen müssen Sie als Folgezahlungen kennzeichnen, indem Sie den Parameter `recurring_typ` bzw. `installment_typ` mit dem Wert `sequencial` nutzen.

Beachten Sie, dass die maximale Speicherzeit mindestens so lange sein muss wie der längste Folgezahlungs-Zyklus (siehe nächstes Kapitel). Die Transaktionsdaten werden nur für diesen Zeitraum gespeichert. Wenn Sie Zeiträume benötigen, die ein Jahr überschreiten, können Sie den Zeitraum durch die Nutzung des Storage-Services verlängern.

Keine automatische Folgezahlung

Die automatische Ausführung der Folgezahlungen wird von ipayment nicht unterstützt. Das bedeutet, dass Folgezahlungen aktiv von Ihnen über die verfügbaren Schnittstellen ausgeführt werden müssen.

Wie lange speichert ipayment die Zahlungsdaten?

Laut den Datenschutzbestimmungen des Teledienstgesetztes, das unter anderem für das ipayment-Zahlungssystem gilt, dürfen die personenbezogenen Daten einer Transaktion nur für bestimmte Zeiträume gespeichert werden. Personenbezogene Daten sind zum Beispiel Name, Adresse, Kreditkartennummer und Bankverbindung. Grundsätzlich ist die Speicherung für den Zeitraum erlaubt, der notwendig ist, um die Zahlungen zu einem Kaufvertrag abzuwickeln. In Hinblick auf die Standardzeiträume für Rückbelastungen (Chargebacks) beträgt die maximale Standard-Speicherzeit für die personenbezogenen Daten von ipayment-Transaktionen drei Monate.

Wenn Sie wiederkehrende Zahlungen oder Ratenzahlungen abwickeln, bei denen die Zeiträume zwischen den einzelnen Folgetransaktionen geringer sind als die Standard-Speicherzeit, reicht die maximale Speicherzeit für Ihre Zahlungen aus. Wenn die Transaktions-Zwischenzeiten länger als die Speicherzeit sind, können Sie über eine Zusatzvereinbarung eine längere Speicherzeit der Daten bei ipayment beantragen. Einen Vordruck für die Standardvereinbarung können Sie per E-Mail an support@ipayment.de anfordern. Senden Sie uns in diesem Fall diese Zusatzvereinbarung vollständig ausgefüllt an die auf dem Formular angegebene Faxnummer zurück.

Nach Ablauf der Speicherzeit werden die personenbezogenen Daten der Transaktionen gelöscht (anonymisiert). Danach sind keine Folgezahlungen mehr möglich, da die Zahlungsdaten der Transaktionen nicht mehr vorhanden sind. Bereits getätigte Transaktionen finden Sie ab dann auch nicht mehr in der Online-Transaktionsübersicht im ipayment-Konfigurationsmenü. Sie können aber weiterhin über den CSV-Export in der Transaktionsübersicht auf die restlichen Daten der anonymisierten Transaktionen zugreifen.

Bei Fragen zu Speicherzeiten oder zu wiederkehrenden Zahlungen oder Ratenzahlungen steht der ipayment-Support (support@ipayment.de) zur Verfügung.

Parameter für regelmäßige Zahlungen

Die folgenden Parameter sind für die Abwicklung von regelmäßige Zahlungen nötig.

Typ der regelmäßigen Zahlung

CGI-Name:	recurring_typ
Webservice-Name:	TransactionData/recurringData/reccurringTyp
Datentyp:	String, erlaubte Werte sind “initial” und “sequencial”

Wenn die genannte Transaktion eine regelmäßige Zahlung (Recurring Payment) ist, müssen Sie den Zahlungstyp setzen. Die erlaubten Werte sind `initial` für die erste Zahlung und `sequencial` für die Folgezahlungen. Die laufenden Folgezahlungen können nur mit den Transaktionstypen **re_preauth** und **re_auth** abgewickelt werden.

Abstand zwischen den regelmäßigen Zahlungen

CGI-Name:	recurring_frequency
Webservice-Name:	TransactionData/recurringData/recurringFrequency
Datentyp:	Integer

Bei der ersten Zahlung (`recurring_typ=“initial“`) muss der minimale Abstand zwischen zwei regelmäßigen Zahlungen in Tagen angegeben werden. Der Wert muss immer größer als 0 sein. Wenn die Zahlungen monatlich erfolgen sollen, geben Sie den Wert 28 an. Wenn der Buchungszeitraum zum Beispiel 3 Monate beträgt, geben Sie als Wert einfach 84 an (3x28).

Beim Ausführen der Folgezahlungen wird geprüft, ob der angegebene Abstand zwischen der letzten erfolgreichen Zahlung und dem neuen Zahlungsversuch eingehalten wurde. Wenn der Zeitraum unterschritten wird, wird die Transaktion mit einer Fehlermeldung abgelehnt.

Wenn bei der initialen Zahlung eine Transaktion des Typs `check_save` durchgeführt wurde, wurden die Zahlungsdaten nur geprüft, ohne dass eine wirkliche Zahlung stattgefunden hat. In diesem Fall können Sie ohne Beachtung des minimalen Zeitabstands die erste Folgezahlung sofort durchführen.

Endzeitpunkt der regelmäßigen Zahlungen

CGI-Name:	recurring_expiry
Webservice-Name:	TransactionData/recurringData/recurringExpiry
Datentyp:	String

Bei der ersten Zahlung (`recurring_typ=“initial“`) muss der Endzeitpunkt der Zahlungen angegeben werden. Der Endzeitpunkt darf hierbei das Verfallsdatum der angegebenen Kreditkarte nicht überschreiten. Durch den Parameter `recurring_allow_expiry_correction` kann das Datum auch automatisch korrigiert werden, so dass ein zu später Endzeitpunkt auch bei Kreditkartenzahlungen nicht zu einem Fehler führt.

Bei ELV-Zahlungen kann das Datum auch weit in der Zukunft liegen, angegeben werden muss es aber immer.

Als Wert wird das US-Datumsformatangaben akzeptiert, wie z.B. "2007/03/15".

Automatische Korrektur des Endzeitpunktes der Zahlungen

CGI-Name:	recurring_allow_expiry_correction
Webservice-Name:	TransactionData/recurringData/recurringAllowExpiryCorrection
Datentyp:	Boolean

Wenn vom verwendeten Zahlungsmedium eine maximale Expiry vorgegeben ist (wie zum Beispiel bei Kreditkarte das Verfallsdatum), kann das ipayment-System den korrekten Recurring-Expiry-Wert selbst ermitteln.

Stellen Sie dazu den Wert des Parameters `recurring_allow_expiry_correction` auf 1 (true). Ansonsten führen Recurring-Expiry-Werte, die über das Karten-Verfallsdatum hinausgehen, zu einem Fehler.

Verspätete Initialisierung von relgemäßigen Zahlungen

CGI-Name:	recurring_ignore_missing_initial
Webservice-Name:	TransactionData/recurringData/recurringIgnoreMissingInitial
Datentyp:	Boolean

Wenn Sie bisher regelmäßige Zahlungen ausgeführt haben, ohne diese explizit zu kennzeichnen, wurde noch kein Schema festgelegt. Das heißt es gab keine erste Zahlung, in der die Dauer für die Folgezahlung angegeben wurde. Dadurch können Sie diese Zahlungen nicht einfach umstellen.

Für diese Sonderfälle können Sie den Parameter `recurring_ignore_missing_initial` auf den Wert 1 setzen und zusätzlich alle weiteren nötigen Parameter, die Sie auch bei der ersten Zahlung angeben würden, übergeben. Somit wird die Initialisierung angenommen und Sie können die Folgezahlungen wie gewohnt abwickeln. Verwenden Sie die zurückgegebene Transaktionsnummer für alle Folgezahlungen.

Parameter für Ratenzahlungen

Die folgenden Parameter sind für die Abwicklung von Ratenzahlungen nötig:

Typ der Ratenzahlung

CGI-Name:	installment_typ
Webservice-Name:	TransactionData/installmentData/installmentTyp
Datentyp:	String, erlaubte Werte sind “initial” und “sequencial”

Wenn die Transaktion eine Ratenzahlung (Installment Payment) ist, muss hier der Typ dieser Zahlung gesetzt werden. Die erlaubten Werte sind `initial` für die Anzahlungen und `sequencial` für die folgenden Ratenzahlungen. Die laufenden Ratenzahlungen können nur mit den Transaktionstypen **re_preauth** und **re_auth** abgewickelt werden.

Anzahl der Ratenzahlungen

CGI-Name:	installment_max_number
Webservice-Name:	TransactionData/installmentData/installmentMaxNumber
Datentyp:	Integer

Bei der Anzahlung (`installment_typ=“initial“`) muss die Anzahl der vereinbarten Teilzahlungen angegeben werden. Der Wert muss größer als 1 sein.

Das ipayment-System prüft die maximale Anzahl der erfolgreichen Zahlungen bei der Abwicklung. Wenn die Anzahl überschritten wird, lehnt ipayment die Zahlung ab.

Verspätete Initialisierung von Ratenzahlungen

CGI-Name:	installment_ignore_missing_initial
Webservice-Name:	TransactionData/installmentData/installmentIgnoreMissingInitial
Datentyp:	Boolean

Wenn Sie bisher Ratenzahlungen ausgeführt haben, ohne diese explizit zu kennzeichnen, wurde noch kein Schema festgelegt. Das heißt es gab keine Anzahlung, in der die Anzahl der Folgezahlungen angegeben wurde. Dadurch können Sie diese Zahlungen nicht einfach umstellen. Für diese Sonderfälle können Sie den Parameter `installment_ignore_missing_initial` auf den Wert 1 setzen und zusätzlich alle weiteren nötigen Parameter, die Sie auch bei der Anzahlung angeben würden, übergeben. Somit wird die Initialisierung angenommen und Sie können die Folgezahlungen wie gewohnt abwickeln. Verwenden Sie die zurückgegebene Transaktionsnummer für alle Folgezahlungen.

Notwendige Parameter

Alle Zahlungsarten

Pflicht-Parameter

Parameter	Bemerkungen
account_id	Account-ID. Wird beim **Normalen Modus**, **Silent-Modus** und **Gateway-Modus** in der URL übermittelt.
trxuser_id	
trxpassword	
adminactionpassword	Wird nicht bei den folgenden Transaktionstypen benötigt: **auth**, **preauth**, **base_check** und **check_save**
trx_currency	
trx_amount	Alternativen: trx_amount_base oder trx_amount_decimal
redirect_url	Nur im **Normalen Modus** und im **Silent-Modus**

Kreditkarte

Pflicht-Parameter

Parameter	Bemerkungen
cc_number	Kreditkartennummer
cc_expdate_month und cc_expdate_year	Ablaufdatum der Kreditkarte

Optionale Parameter

Parameter	Bemerkungen
cc_checkcode	Dieser Parameter wird zum Pflicht-Parameter, wenn in den Einstellungen der Anwendung angegeben ist, dass CVV2 verlangt wird.
cc_startdate_month und cc_startdate_year	Nur bei bestimmten Kartentypen
cc_issuenumber	Nur bei bestimmten Kartentypen
cc_voice_authcode	Nur bei den Transaktionstypen **voice_auth** und **voice_grefund_cap**

Elektronisches Lastschriftverfahren

Pflicht-Parameter

Parameter	Bemerkungen
bank_accountnumber	Kontonummer
bank_code	Pflicht bei Bankverbindungen, bei denen es eine Bankleitzahl gibt.

Optionale Parameter

Parameter	Bemerkungen
bank_name	Name der Bank
bank_country	Wenn die Bankverbindung aus einem anderen Land wie addr_country ist. Wenn addr_country nicht angegeben wurde, wird DE verwendet.
bank_iban	Wenn dieser Wert angegeben wurde, ersetzt er bank_accountnumber, bank_code und bank_country.
bank_bic	

Pflichtparameter nach Transaktionstypen

	Zahlungsdaten	trx_amount	trx_currency	trx_typ	addr_name	account_id	trxpassword	adminactionpassword	orig_trx_number	cc_voice_authcode
preauth	✔	✔	✔	✔	✔	✔	✔			
auth	✔	✔	✔		✔	✔	✔			
base_check	✔	✔	✔	✔	✔	✔	✔			
check_save	✔	✔	✔	✔	✔	✔	✔			
re_preauth		✔	✔	✔		✔	✔	✔	✔	
re_auth		✔	✔	✔		✔	✔	✔	✔	
capture				✔		✔	✔	✔	✔	
reserve				✔		✔	✔	✔	✔	
refund_cap				✔		✔	✔	✔	✔	
grefund_cap	✔	✔	✔	✔	✔	✔	✔	✔		
voice_auth	✔	✔	✔	✔	✔	✔	✔	✔		✔
voice_grefund_cap	✔	✔	✔	✔	✔	✔	✔	✔		✔

Parameter für die Adressprüfung

Durch den Adressprüfungs-Service können Sie Adressen automatisch auf Korrektheit prüfen lassen. Schreibfehler werden erkannt und sofort korrigiert. Wenn die Korrektur nicht möglich ist, weil zu viele Alternativ-Vorschläge gefunden wurden, werden diese Vorschläge als Liste zurückgegeben. Wenn beim Aufruf keine Hausnummer mit angegeben wurde, versucht der Adressprüfungs-Service, diese anhand der Straße zu extrahieren. Wenn das nicht möglich ist, wird die Prüfung ohne die Hausnummer vorgenommen. Dieses Verfahren kann jedoch zu einem Fehler führen. Ergebnisse werden nur für die Felder zurückgegeben, die auch ausgefüllt wurden.

Sinnvolle Rückmeldungen nur bei sinnvollen Daten

Je fehlerhafter die zu prüfenden Adressfelder sind, desto fehlerhafter werden auch die Ergebnisse des Adressprüfungs-Services sein. Der Dienst dient dazu, die Felder zu prüfen und einfache Fehler und/oder Verschreiber vor allem bei Straßen- oder Ortsnamen zu erkennen und zu korrigieren. Scherzeingaben werden nicht als solche erkannt.

Zusätzlich zu den speziellen Parametern für die Adressprüfung müssen alle Basisparameter angegeben werden. Informationen dazu finden Sie im Abschnitt *Basisparameter* ab Seite 28. Außerdem gelten dieselben Datentypen wie bei den Parametern zur Zahlungsabwicklung. Mehr dazu können Sie im Abschnitt *Verwendete Datentypen* ab Seite 28 nachlesen.

Für die Nutzung des Adressprüfungs-Service werden folgende Parameter benötigt:

Anzahl der zurückzugebenden Korrekturvorschläge

CGI-Name:	`max_suggestions`
Webservice-Name:	`maxSuggestions (in Methoden als Parameter, wenn nötig)`
Datentyp:	`Integer`

Über diesen Parameter können Sie angeben, wie viele Auswahlmöglichkeiten maximal zurückgegeben werden, wenn die Adresse nicht korrekt ist. Wenn dieser Wert nicht angegeben wird, werden standardmäßig maximal 5 Ergebnisse zurückgeliefert. Der Wert sollte zwischen 3 und 10 liegen. Höhere Werte als 10 werden automatisch auf 10 heruntergesetzt.

Eindeutige ID der Addressprüfungsanfrage

CGI-Name:	`request_id`
Webservice-Name:	`requestId (in Methoden als Parameter, wenn nötig)`
Datentyp:	`String`

Eindeutige ID eines Bestellablaufs im Shop. Erfolgreiche Adressprüfungen werden für eine gleichbleibende Adresse innerhalb einer bestimmten Zeit nur einmal abgerechnet. Diese eindeutige ID sollte deshalb für einen Bestellablauf unverändert bleiben, so dass auch ein mehrmaliges Vor- und Zurückspringen nur eine einmalige Adressprüfungs-Abrechnung gewährleistet. Zusätzlich zur ID müssen aber dann auch die Adressdaten übereinstimmen.

Rückgabeparameter Zahlungsabwicklung

Rückgabeparameter zum Transaktionsergebnis

Diese Parameter informieren über das Ergebnis einer Transaktion. Mit diesen Parametern können Sie zum Beispiel feststellen, ob die Transaktion erfolgreich ausgeführt wurde oder ein Fehler aufgetreten ist.

Ergebnis-Status der Transaktion

CGI-Name:	`ret_status`
Webservice-Name:	`PaymentReturn/status`
Datentyp:	`String`

Die möglichen Werte sind:

- **SUCCESS:** Die Transaktion wurde erfolgreich abgewickelt.
- **ERROR**: Bei der Transaktionsabwicklung gab es einen Fehler.
- **REDIRECT:** Zur weiteren Abwicklung muss ein Redirect ausgeführt werden (siehe *Überprüfung des Karteninhabers mit 3-D Secure* ab Seite 43.)

Fehlercode der Transaktion

CGI-Name:	`ret_errorcode`
Webservice-Name:	`PaymentReturn/errorDetails/retErrorcode (wenn PaymentReturn/status den Wert "ERROR" zurückgibt)`
Datentyp:	`Integer`

Fehlernummer der Transaktion.

Der Fehlercode 0 bedeutet, dass die Transaktion erfolgreich ausgeführt wurde. Wenn in einem CGI-Integrationsmodus der Parameter `redirect_needed` mit dem Wert 1 zurückgegeben wird, bedeutet das nur, dass alle Daten korrekt sind und ein Redirect ausgeführt werden muss. Der Rückgabewert ist erst nach einem zweiten Aufruf aussagekräftig.

Eine Liste der möglichen Fehlercodes finden Sie unter https://ipayment.de **> Technik**

Ist ein technischer Fehler oder eine Ablehnung aufgrund der Zahlungsdaten aufgetreten?

CGI-Name:	`ret_fatalerror`
Webservice-Name:	`PaymentReturn/errorDetails/retFatalerror (wenn PaymentReturn/status den Wert "ERROR" zurückgibt)`
Datentyp:	`Boolean`

Dieser Wert wird nur zurückgegeben, wenn ein Fehler aufgetreten ist.

Basierend auf diesem Wert kann Ihr Shop dem Käufer die Möglichkeit bieten, die Zahlungsdaten zu korrigieren, solange kein fataler Fehler vorliegt. Fatale Fehler sind meist Störungen im Banken-Netzwerk oder andere Probleme, bei denen ein neuer Versuch voraussichtlich keine Verbesserung bringt. Ihrem Kunden können Sie in diesem Fall eine spezielle Fehlermeldung anzeigen.

Fehlermeldung

CGI-Name:	`ret_errormsg`
Webservice-Name:	`PaymentReturn/errorDetails/retErrorMsg (wenn PaymentReturn/status den Wert "ERROR" zurückgibt)`
Datentyp:	`String`

Dieser Wert wird nur zurückgegeben, wenn ein Fehler aufgetreten ist.

Die Fehlermeldung wird in verständlichen Sätzen zurückgegeben, also nicht nur ein numerischer Code, den Ihr Kunde nicht versteht. Die Sprache der Fehlermeldung können Sie über den Parameter `error_lang` angeben. Wenn Sie die Sprache nicht explizit angeben, wird die Fehlermeldung auf Deutsch zurückgeliefert.

Zusätzliche Fehlerinformation

CGI-Name:	`ret_additionalmsg`
Webservice-Name:	`PaymentReturn/errorDetails/retAdditionalMsg (wenn PaymentReturn/status den Wert "ERROR" zurückgibt)`
Datentyp:	`String`

Dieser Wert wird nur zurückgegeben, wenn ein Fehler aufgetreten ist.

Zusatztext zur Fehlermeldung mit ergänzenden Details zum aufgetretenen Fehler. Dieser Text wird stets auf englisch zurückgegeben und ist nicht zur Anzeige an Endkunden geeignet, jedoch kann Sie dieser Text bei der Fehlersuche unterstützen.

Rückgabeparameter zu erfolgreichen Transaktionen

In diesen Parametern werden Informationen zu erfolgreichen Transaktionen zurückgegeben, zum Beispiel der Zeitpunkt der Transaktion, die eindeutige Transaktionsnummer und weitere Informationen.

Datum und Uhrzeit der Transaktion

CGI-Name:	`ret_transdate, ret_transtime`
Webservice-Name:	`PaymentReturn/successDetails/retTransDate und PaymentReturn/successDetails/retTransTime (wenn PaymentReturn/status den Wert "SUCCESS" zurückgibt.)`
Datentyp:	`String`

Datum und Uhrzeit des Transaktionszeitpunktes.

Eindeutige Buchungsnummer/Transaktionsnummer der Transaktion

CGI-Name:	`ret_trx_number`
Webservice-Name:	`PaymentReturn/successDetails/retTrxNumber (wenn PaymentReturn/status den Wert "SUCCESS" zurückgibt.)`
Datentyp:	`String`

Eindeutige Transaktionsnummer (Buchungsnummer) des ipayment-Systems. Diese Nummer wird in der Form „x-xxxxxxx" zurückgegeben, wobei x für einzelne Ziffern steht. Mit dieser Transaktionsnummer können Sie weitere Aktionen wie Abbuchungen oder Stornierungen ausführen.

Autorisierungsnummer der Transaktion

CGI-Name:	`ret_authcode`
Webservice-Name:	`PaymentReturn/successDetails/retAuthCode (wenn PaymentReturn/status den Wert "SUCCESS" zurückgibt.)`
Datentyp:	`String`

Autorisierungsnummer des Zahlungsanbieters für diese Transaktion oder eine andere eindeutige Identifikation der Zahlung beim Zahlungsanbieter. Der Parameter kann in bestimmten Fällen auch leer sein.

Ergebnis der AVS-Prüfung des Zahlungsanbieters (wenn unterstützt)

CGI-Name:	`trx_issuer_avs_response`

Webservice-Name:	PaymentReturn/successDetails/trxIssuerAvsResponse
Datentyp:	String

In diesem Parameter wird, wenn möglich, die AVS-Antwort der kartenausgebenden Bank zurückgegeben. Beim AVS (Address Verification Service) werden Teile der Karteninhaberadresse an die Banken übermittelt und dort gegen die Kartendaten geprüft.

Es gibt momentan zwei verschiedene AVS-Systeme: Eins aus den USA und eins aus Großbritannien. Die beiden Systeme unterscheiden sich im Umfang der geprüften Adressdaten. Der Spalte „Land" der folgenden Tabelle können Sie entnehmen, welches AVS-System zum Einsatz kam. Der Karteninhaber selbst muss nicht aus den genannten Ländern stammen.

Folgende Werte werden in diesem Parameter zurückgegeben:

Wert	Land	Bedeutung
A	US	Straßenfeld stimmt überein, aber PLZ stimmt nicht überein
B	US	Adressdaten wurden nicht für das AVS-System übermittelt
E	US	Fehler beim AVS
F	UK	Adresse stimmt überein
G	US	Kartenausgebende Bank ist nicht aus den USA
N	US + UK	Straßenfeld und PLZ stimmen nicht überein
P	US	AVS kann für diese Transaktion nicht verwendet werden
R	US	Das System ist momentan nicht erreichbar, bitte nochmals versuchen
S	US	AVS wird von der kartenausgebenden Bank nicht unterstützt
U	US + UK	Es sind keine Adressdaten für den Karteninhaber gespeichert
W	US	Straßenfeld stimmt nicht überein, aber 9-stellige PLZ stimmt überein
X	US	Straßenfeld und 9-stellige PLZ stimmen überein
Y	US	Straßenfeld und 5-stellige PLZ stimmen überein
Z	US	Straßenfeld stimmt nicht überein, aber 5-stellige PLZ stimmt überein.

Ob ein AVS-Ergebnis zur Verfügung steht, ist von zwei Faktoren abhängig: Die kartenausgebende Bank muss AVS unterstützen und das Karteninstitut muss die Daten weitergeben können. Derzeit stehen AVS-Rückgabewerte nur für das ipayment-Terminalsystem (derzeit nur mit Acquirer Euroconex und American Express), Protx, Authorize.net und Worldpay zur Verfügung.

Status der 3-D-Secure-Prüfung

CGI-Name:	trx_payauth_status
Webservice-Name:	PaymentReturn/successDetails/trxPayauthStatus
Datentyp:	String

In diesem Parameter ist der Status der 3-D-Secure-Prüfung enthalten. Die folgenden Werte sind möglich:

- I (Issuer Authenticated): Vollständige und erfolgreiche Authentifizierung des Karteninhabers. Eine Haftungsumkehr besteht.
- M (Merchant Attempted):Die Kreditkarte war nicht für das Sicherheitsverfahren freigeschaltet. Der Händler hat das Sicherheitsverfahren angeboten, die Haftungsumkehr besteht.
- U (Unavailable): Eine Prüfung ist nicht möglich, es besteht keine Haftungsumkehr. Diese Transaktion sollte sorgfältig geprüft werden!

Checksumme der CGI-Rückgabeparameter

CGI-Name:	ret_param_checksum
Webservice-Name:	- (nicht benötigt)
Datentyp:	String

Wenn Sie für eine Transaktion einen Security-Key verwenden und somit den Aufruf der Transaktion mit einer MD5-Checksumme gesichert haben, bekommen Sie in `ret_param_checksum` auch eine MD5-Checksumme zurückgeliefert. Mithilfe dieser Checksumme können Sie die Richtigkeit der Rückgabeparameter prüfen. Der Hash wird über die Felder `trxuser_id`, `trx_amount`, `trx_currency`, `ret_authcode`, `ret_booknr` und dem Transaktions-Security-Key der Anwendung generiert. Diese Felder werden dabei ohne Leerzeichen oder sonstige Trennzeichen als String aneinandergehängt. Es werden die Werte der Felder verwendet, die in den Rückgabeparametern enthalten sind. Wenn eins der Felder leer ist oder nicht zurückgegeben wird, wird dieses Feld beim Aneinanderhängen der Werte wie ein Leerstring behandelt. Die Checksumme steht nur im Erfolgsfall einer Transaktion (`ret_errorcode=0` und `redirect_needed=0`) zur Verfügung.

Mit dieser Checksumme können Sie die Richtigkeit der Antwort verifizieren und mögliche Manipulationen erkennen. Noch mehr Sicherheit gegen Manipulationen können Sie erreichen, indem Sie den Parameter `ret_url_checksum` einsetzen.

Checksumme der CGI-Rücksprungs-URL

CGI-Name:	ret_url_checksum
Webservice-Name:	- (nicht benötigt)
Datentyp:	String

Wenn Sie für eine Transaktion eine Anwendung mit einem Security-Key verwendet haben, wird dieser Parameter mit einem MD5-Hash an die Rücksprungs-URL angehängt.

Für die Bildung des Hash wird an die Rücksprungs-URL ein & und der Transaktions-Security-Key der Anwendung angehängt. Für diese Zeichenkette wird die MD5-Prüfsumme generiert. Der ermittelte Hash wird als Parameter `ret_url_checksum` an die Rücksprungs-URL hinter alle anderen Parameter an das Ende angehängt.

Um die Prüfsumme zu überprüfen müssen Sie den Parameter `ret_url_checksum` von der vollständigen URL des aufgerufenen Scriptes abschneiden, den Transaktions-Security-Key anhängen und dann die MD5-Prüfsumme ermitteln. Wenn die Prüfsumme nicht mit dem Wert des Parameters `ret_url_checksum` übereinstimmt, liegt vermutlich eine Manipulation der URL vor.

Weitere Rückgabeparameter

Land der Zahlungsdaten der Transaktion

CGI-Name:	trx_paymentdata_country
Webservice-Name:	PaymentReturn/trxPaymentdataCountry
Datentyp:	String

In diesem Parameter wird, wenn möglich, der ISO-Code des Landes zurückgegeben, zu dem die Zahlungsdaten gehören. Das Feld enthält zum Beispiel bei Kreditkartenzahlungen das Land der kartenausgebenden Bank und bei ELV-Zahlungen das Bankenland.

Genauigkeit der Daten

Die Daten, die über diesen Parameter zurückgegeben werden, haben durch ständige Änderungen bei den Banken eine Genauigkeit von ca. 99%.

Land der verwendeten IP der Transaktion

CGI-Name:	trx_remoteip_country
Webservice-Name:	PaymentReturn/trxRemoteIpCountry
Datentyp:	String

In diesem Parameter wird, wenn möglich, der ISO-Code des Landes zurückgegeben, in dem die zur Zahlung verwendete IP-Adresse vergeben wurde. Neben den offiziellen ISO-Codes können folgende Werte zurückgegeben werden:

- A0 für AOL-Client-IPs und AOL-Proxies, wenn die IP nicht einem bestimmten Land zugeordnet werden kann.
- A1 für anonyme Proxies
- A2 für Satelliten-Provider
- EU für Europa
- AP für Asien/Pazifik-Region

Genauigkeit der Daten

Die Daten, die über diesen Parameter zurückgegeben werden, haben durch ständige Änderungen der IP-Bereiche und der Registrare eine Genauigkeit von ca. 99%.

Verwendete IP der Transaktion

CGI-Name:	ret_ip
Webservice-Name:	- (nicht unterstützt)
Datentyp:	String

In diesem Parameter wird die IP des Kunden zurückgegeben, der die Bestellung durchgeführt hat.

Zahlungsmedium der Transaktion

CGI-Name:	trx_paymentmethod
Webservice-Name:	PaymentReturn/paymentMethod
Datentyp:	String

In diesem Parameter wird der Name des verwendeten Zahlungsmediums zurückgegeben. Das kann zum Beispiel ein Kreditkartentyp sein (wie VisaCard oder MasterCard) oder ELV.

Rückgabe der maskierten Zahlungsdaten der Transaktion

CGI-Name:	paydata_*
Webservice-Name:	- (nicht unterstützt)
Datentyp:	String

Wenn der Eingabeparameter return_paymentdata_details gesetzt ist, werden in den Parametern, die mit paydata_ beginnen, die verwendeten Zahlungsdaten maskiert zurückgegeben. Die Namen haben nach paydata_ die gleichen Namen wie die Eingabeparameter (zum Beispiel die Kartennummer: paydata_cc_number).

Ergebnis des Adresschecks der Transaktion

CGI-Name:	addr_check_result
Webservice-Name:	PaymentReturn/addresscheckResult
Datentyp:	String

Ein Statuswert für das Ergebnis der Adressprüfung mit folgenden möglichen Werten: `UNCHECKED`, `OK`, `ERROR`, `CORRECTED`, `NORMALIZED`, `SUGGESTIONS`. (Die Bedeutung der Felder können Sie in den nachfolgenden Kapitel nachlesen).

Wenn keine Adressprüfung für die Transaktion verwendet wurde, wird der Wert `UNCHECKED` zurückgegeben.

Nur Adressdaten werden geprüft

Die Adressprüfung kann nur die angegebenen Adressdaten anhand des Datenbestands der Deutschen Post prüfen, jedoch nicht vergleichen, ob die angegebenen Daten tatsächlich mit der Adresse des Karteninhabers übereinstimmen.

Rückgabeparameter Adressprüfung

Pro Adressdaten-Feld werden folgende Daten zurückgegeben:

Feldname der zurückgegeben Daten in dieser Zeile

CGI-Name:	`ret_field`
Webservice-Name:	`- (spezielle Felder in AddresscheckReturn)`
Datentyp:	`String`

Gibt das Feld an, das die Ergebnisse dieser Zeile beinhaltet. Der Inhalt entspricht den möglichen Aufrufparametern für die folgenden Adressfelder: `addr_street`, `addr_street_number`, `addr_street2`, `addr_city`, `addr_zip`, `addr_state`

Adressprüfungs-Status der ganzen Anfrage bzw. eines speziellen Adressfeldes

CGI-Name:	`status`
Webservice-Name:	`AddresscheckReturn/status bzw. AddresscheckReturn/addr*/status`
Datentyp:	`String`

Gibt den Status dieses Feldes an. Der Status kann folgende Werte annehmen:

- `OK`: Feld ist in Ordnung und der Wert des Feldes ist korrekt.
- `NORMALIZED`: Der Wert wurde eindeutig korrigiert und das Ergebnis befindet sich als einzige Auswahlmöglichkeit im Ergebnisparameter `suggestionlist`. In diesem Fall wurde der Wert normalisiert. Dies bedeutet, dass sich der originale und der zurückgegebene Wert zum Beispiel nur in der Schreibweise und dem Setzen von Bindestrichen oder Leerzeichen unterscheiden. Ein Feld mit diesem Status hat keine Auswirkung auf den Gesamtstatus der Prüfung.
- `CORRECTED`: Der Wert wurde eindeutig korrigiert und das Ergebnis befindet sich als einzige Auswahlmöglichkeit im Ergebnisparameter `suggestionlist`.
- `SUGGESTIONS`: Der Wert stimmt nicht und im Parameter `suggestionlist` befinden sich mehrere Auswahlvorschläge für korrekte Adressen.
- `ERROR`: Der Wert stimmt nicht und es konnten auch keine Auswahlmöglichkeiten gefunden werden. Dieser Fehler tritt auf, wenn die Daten so falsch sind, dass Sie keinen Sinn ergeben (zum Beispiel willkürlich gewählte Straßennamen wie „Teststraße“).
- `UNCHECKED`: Das Feld wurde nicht geprüft. Das kann auftreten, wenn das Land nicht unterstützt wird oder wenn ein Feld angegeben wurde, das in dem Land keine Bedeutung hat (zum Beispiel in Deutschland das Zusatzfeld für die Straße). Ein Feld mit diesem Status hat keine Auswirkung auf den Gesamtstatus der Prüfung.

Originalwert des Feldes bei der Anfrage

CGI-Name:	origvalue
Webservice-Name:	AddresscheckReturn/addr*/origValue
Datentyp:	String

Dieser Rückgabeparameter enthält den beim Aufruf übergebenen Originalwert.

Liste mit Korrekturvorschlägen

CGI-Name:	suggestionlist
Webservice-Name:	AddresscheckReturn/addr*/suggestionList
Datentyp:	String

Dieser optionale Rückgabeparameter enthält bei Bedarf eine Liste mit Auswahlvorschlägen. Bei einem Aufruf per CGI sind die einzelnen Auswahlvorschläge durch das Pipe-Zeichen (|) getrennt. Bei Nutzung der Webservice-Schnittstelle werden die Auswahlvorschläge als Array (Liste) zurückgegeben. Dieser Parameter ist bei Feldern mit den Status `CORRECTED, NORMALIZED` und `SUGGESTIONS` vorhanden.

Detailinformationen zum Status

CGI-Name:	statusdetail
Webservice-Name:	AddresscheckReturn/addr*/statusDetail
Datentyp:	String

Dieser optionale Rückgabeparameter enthält bei Bedarf weitere Informationen zum Status des geprüften Feldes. Der Inhalt kann zum Beispiel nähere Informationen zu vorgenommenen Korrekturen enthalten. Der Wert wird immer auf Englisch ausgegeben und ist nicht zur Anzeige für Endkunden gedacht.

Sichere Integration von ipayment

Dieses Kapitel widmet sich der Einbindung des ipayment-Systems in Ihre Webanwendungen. Die verschiedenen Möglichkeiten der Integration werden zuerst anhand von Nutzungsbeispielen beschrieben. Im Anschluss stellen wir noch weitere Anwendungsfälle und -möglichkeiten vor, die verschiedene Integrationsmethoden sinnvoll kombinieren. Anhand dieser Anwendungsfälle sehen Sie die verfügbaren Möglichkeiten für die optimale und sichere Integration von ipayment in Ihre Web-Anwendung.

Ausführen von Zahlungen

Zur Abwicklung von Zahlungen muss Ihr Kunde seine Zahlungsdaten irgendwo eingeben, damit diese verarbeitet werden können. Da die eingegebenen Daten möglicherweise sehr sensibel sind, müssen Sie sich genaue Gedanken darüber machen, auf welche Weise die Daten eingegeben, verarbeitet oder gespeichert werden. Je nachdem wie die Eingabe der Zahlungsdaten realisiert wird, kann eine Zertifizierung nach den PCI-DSS-Regeln nötig werden. Nähere Informationen zur PCI-Zertifizierung finden Sie im Kapitel *Was ist PCI DSS?* auf Seite 11.

Im Folgenden werden der **Normale Modus** und der **Silent-Modus** von ipayment genauer betrachtet, da bei diesen beiden Integrations-Methoden die Zahlungsdaten ausschließlich vom ipayment-System verarbeitet werden. Dadurch hat Ihre Anwendung keinen direkten Kontakt mit den Zahlungsdaten und die Zertifizierungspflicht entfällt. Außerdem ist die Zahlungsauthentifizierung mittels 3-D Secure ohne weitere Änderungen sofort möglich.

Zahlungen mit dem Normalen CGI-Modus

Im **Normalen Modus** wird Ihr Kunde von Ihrer Webanwendung zur Eingabe der Zahlungsdaten für die Transaktion auf den Webserver von ipayment weitergeleitet. Nach der Durchführung der Zahlung wird Ihr Kunde wieder automatisch zurück auf Ihren Server geleitet.

Die folgenden Beispiele nutzen den ipayment-Testaccount. Nähere Informationen dazu finden Sie im Kapitel *ipayment-Funktionen testen (Simulationsmodus)* auf Seite 9. So können Sie eigene Tests durchführen: Ersetzen Sie einfach die Werte der Parameter `trxuser_id` und `trxpassword` sowie die Account-ID in der URL des Formulars durch die Werte Ihres ipayment-Accounts und aktivieren Sie den Simulationsmodus in der ipayment-Konfiguration.

Beispiel für ein sehr einfaches Formular zur Durchführung einer Zahlung über 129,89 EUR mittels Kreditkarte:

```
<form method="post"
      action="https://ipayment.de/merchant/99999/processor/2.0/">
  <!—- Base Parameter -->
  <input type="hidden" name="trxuser_id" value="99999">
  <input type="hidden" name="trxpassword" value="0">

  <!—- Amount and Currency for Payment -->
  <input type="hidden" name="trx_amount" value="12989">
  <input type="hidden" name="trx_currency" value="EUR">

  <!—- URL and Parameter for Redirect back to Shop -->
  <input type="hidden" name="redirect_url"
         value="https://your_domain/payment_success.php">
  <input type="hidden" name="redirect_action" value="POST">

  <!—- Paymenttyp: CC to make creditcard payment -->
  <input type="hidden" name="trx_paymenttyp" value="cc">
```

```
  <!—- Submit Button -->
  <input type="submit" name="form_submit"
    value="Process payment">
</form>
```

Das HTML-Formular enthält nur einige Parameter für das ipayment-Zahlungssystem. Wenn Sie zum Beispiel nur ein Produkt haben, kann es für alle Zahlungen gleich aussehen. Über den Parameter `trx_paymenttyp` wird dem ipayment-System mitgeteilt, welches Formular für die Zahlungsdaten (hier Kreditkarte) angezeigt werden soll.

Alternativ kann das Formular auf Ihrem Server auch dynamisch generiert werden. Dadurch kann zum Beispiel der aktuelle Preis passend zu den gewählten Produkten Ihres Shops im Formular ausgefüllt werden oder die möglichen Zahlungsarten können über eine Auswahlliste eingebunden werden.

Wenn die Zahlung erfolgreich war, bekommt Ihr Kunde einen "Weiter"-Button angezeigt, über den er zurück auf Ihren Server geführt wird. Geben Sie dazu im Parameter `redirect_url` die URL der gewünschten Seite an. Die ipayment-Rückgabeparameter werden anschließend per `POST` an diese URL zurückgegeben.

Sie können das Layout der Seiten zur Eingabe der Zahlungsdaten und der Ergebnisseite beeinflussen. Klicken Sie dazu im ipayment-Konfigurationsmenü unter **Anwendungen** auf **Template-Texte** und passen Sie die gewünschten Seiten nach Ihren Bedürfnissen an. Je nachdem, welche Änderungen Sie vornehmen möchten, benötigen Sie dazu teilweise umfangreiche HTML-Kenntnisse.

Zahlungen mit dem Silent-Modus (CGI)

Der **Silent-Modus** ähnelt dem **Normalen Modus**. Allerdings haben Sie im **Silent-Modus** volle Kontrolle über das Aussehen aller Zahlungsseiten, im **Normalen Modus** dagegen nicht. Das Formular mit den Feldern zur Eingabe der Zahlungsdaten wird auf Ihrem Server angezeigt, die eingegebenen Zahlungsdaten werden jedoch direkt an ipayment übermittelt. Somit werden die Zahlungsdaten weiterhin nur vom ipayment-System verarbeitet. Der ipayment-Server selbst gibt keine Webseite aus, sondern leitet Ihren Kunden je nach Erfolg oder Fehler auf eine von Ihnen angegebene URL auf Ihrem Server weiter.

Dieser Integrationsmodus setzt voraus, dass die Zahlungsseite dynamisch generiert wird. Dadurch kann im Fehlerfall die zurückgegebene Fehlermeldung angezeigt werden und die weiteren Werte, die vom ipayment-Server zurückgegeben werden, können korrekt verarbeitet oder gespeichert werden.

Beispiel für ein Formular mit dem Silent-Modus

Das folgende Formular basiert auf dem Beispiel-Formular zum **Normalen Modus** aus dem letzten Abschnitt. Im Unterschied zu diesem kommen für den **Silent-Modus** zusätzliche versteckte Parameter und die entsprechenden Felder für die Eingabe der Zahlungsdaten dazu. Die weiteren möglichen Felder für alle von ipayment unterstützten Zahlungsdaten finden Sie im Abschnitt *Zahlungsdaten* ab Seite 38.

Alle Änderungen am Formular sind fett hervorgehoben.

```
<form method="post"
      action="https://ipayment.de/merchant/99999/processor/2.0/">
  <!—- Base Parameter -->
  <input type="hidden" name="trxuser_id" value="99999">
  <input type="hidden" name="trxpassword" value="0">

  <!—- Amount and Currency for Payment -->
  <input type="hidden" name="trx_amount" value="12989">
```

```
  <input type="hidden" name="trx_currency" value="EUR">

  <!—- Paymenttyp: CC to make creditcard payment -->
  <input type="hidden" name="trx_paymenttyp" value="cc">

  <!—- URL and Parameter for Redirect back to Shop -->
  <input type="hidden" name="redirect_url"
         value="https://your_domain/payment_success.php">
  <input type="hidden" name="silent" value="1">
  <input type="hidden" name="silent_error_url"
         value="https://your_domain/enter_payment_data.php">

  <!—- Credit card data fields -->
  Cardholder name: <input type="text" name="addr_name"
         value=""><br>
  Credit card no.: <input type="text" name="cc_number"
         value=""><br>
  Card Check Code: <input type="text" name="cc_checkcode"
         value=""><br>
  Card Expire date: <select name="cc_expdate_month">
                       <option>01</option>
                       <option>02</option>
                       <option>03</option>
                       <option>04</option>
                       <option>05</option>
                       <option>06</option>
                       <option>07</option>
                       <option>08</option>
                       <option>09</option>
                       <option>10</option>
                       <option>11</option>
                       <option>12</option>
         </select>
          / 
         <select name="cc_expdate_year">
                       <option>2007</option>
                       <option>2008</option>
                       <option>2009</option>
                       <option>2010</option>
                       <option>2011</option>
                       <option>2012</option>
                       <!-- More Years … -->
         </select><br>

  <!—- Submit Button -->
  <input type="submit" name="form_submit"
         value="Process payment">
</form>
```

Wenn das Formular abgesendet wird, erfolgt nach der Verarbeitung der Daten durch ipayment automatisch ein Rücksprung auf Ihren Server. Je nach Ergebnis wird dabei die im Formular angegebene Fehler-URL (`silent_error_url`) oder die Erfolgs-URL (`redirect_url`) aufgerufen. In beiden Fällen werden einige Parameter an Ihre Web-Anwendung zurückgegeben. Ihre Web-Anwendung muss dem Besucher eventuelle Fehlermeldungen anzeigen, damit Ihr Kunde weiß, weshalb die Zahlung nicht funktioniert hat. Idealerweise geht der Rücksprung im Fehlerfall wieder zu der Seite mit dem Formular für die Eingabe der Zahlungsdaten.

Die an die URL angehängten Parameter können zum Beispiel einfach mittels PHP ausgelesen werden. Ein Beispiel:

```
<?php
  while (list ($key, $val) = each ($_GET)) {
    echo $key.": ".htmlentities($val)."<br>";
  }
?>
```

Zur Integration von ipayment im **Silent-Modus** benötigen Sie zusätzlich Kenntnisse in einer Programmiersprache, um die vom ipayment-System zurückgegeben Daten korrekt verarbeiten zu können.

Wichtige Funktionen und weitere Parameter zur Fehlerbehandlung, zur Vermeidung von Doppeltransaktionen bzw. die gesicherte Rückmeldung bei erfolgreichen Zahlungen werden im folgenden Abschnitt beschrieben.

Empfohlene zusätzliche Parameter für Zahlungen

ipayment bietet die Möglichkeit, zusätzliche Parameter zu verwenden, durch die Sie Ihre Anwendung individueller gestalten können. So können Sie zum Beispiel erreichen, dass die Zahlungen manipulationssicher zum Server übertragen werden. Doppelte Zahlungen können vermieden werden, und bei erfolgreichen Zahlungen können Sie eine Meldung an Ihre Web-Anwendung zurückgeben lassen. Wir empfehlen, die folgenden Parameter in Ihrer Web-Anwendung zu nutzen.

Manipulation der Parameter verhindern

Damit Ihr Kunde die Inhalte des Zahlungsformulars nicht manipulieren kann, können Sie eine Prüfsumme über einige Felder und einem selbstgewählten Passwort (Transaktions-Security-Key) bilden. Diese Prüfsumme wird im Aufruf an das ipayment-System übergeben. Sie können die Prüfsumme bilden, in dem Sie die Felder `trxuser_id`, `trx_amount`, `trx_currency`, `trxpassword` und den Security-Key in genau dieser Reihenfolge aneinanderhängen und aus dieser Zeichenkette einen MD5-Hash generieren. Dafür muss das Formular bei der Nutzung von unterschiedlichen Preisen dynamisch generiert werden, um den Hash-Wert jedes Mal neu zu berechnen. Diese Generierung kann in PHP zum Beispiel mit folgender Zeile geschehen:

```
$sec_key= "qundhft67dnft"; // Security-Key
$trx_securityhash=

md5($trxuser_id.$trx_amount.$trx_currency.$trxpassword.$sec_key);
```

Zur Verifizierung geben Sie im ipayment-Konfigurationsmenü unter **Anwendungen** bei der entsprechenden Anwendung im Feld „Transaktions-Security-Key“ denselben Inhalt ein, den Sie auch in der Variable `sec_key` verwenden. Benutzen Sie dazu am besten eine zufällige Zeichenkette.

Wenn Sie den Transaktions-Security-Key verwenden und beim Aufruf der Transaktion bereits einen Hash übermittelt haben, wird im Falle einer erfolgreichen Transaktion der Parameter `ret_param_checksum` mit einem Hash-Wert zurückgegeben. Der Hash wird gebildet, indem die Felder `trxuser_id`, `trx_amount`, `trx_currency`, `ret_authcode`, `ret_trx_number` und der Transaktions-Security-Key der Anwendung aneinander gehängt werden. Aus diesem String wird wiederum ein MD5-Hash generiert. Dadurch können Sie feststellen, ob der Aufruf tatsächlich vom ipayment-System kam und nicht manipuliert wurde, da der benutzte Transaktions-Security-Key nur Ihnen und ipayment bekannt ist. Eine Prüfung des Hash-Wertes könnte zum Beispiel wie folgt aussehen:

```
$security_key= "qundhft67dnft";
$return_checksum="";
if (isset($_GET["trxuser_id "]))
  $return_checksum.= $_GET["trxuser_id"];
if (isset($_GET["trx_amount "]))
  $return_checksum.= $_GET["trx_amount"];
if (isset($_GET["trx_currency "]))
  $return_checksum.= $_GET["trx_currency"];
if (isset($_GET["ret_authcode "]))
  $return_checksum.= $_GET["ret_authcode"];
if (isset($_GET["ret_trx_number "]))
  $return_checksum.= $_GET["ret_trx_number"];
$return_checksum.= $security_key;

if ($_GET["ret_param_checksum "]!=md5($return_checksum)) {
  // Error because hash do not match!
  exit;
}
```

Sobald ein Transaktions-Security-Key verwendet wird, wird im **Silent-Modus** noch ein zweiter Hash beim Rücksprung mitgegeben. In diesem Fall wird aus der Rücksprungs-URL inklusive aller Ergebnis-Parameter und dem Transaktions-Security-Key der Anwendung eine MD5-Summe gebildet. Dieser Hash wird dann im Parameter `ret_url_checksum` an die URL angehängt. Sie können den Hash wie im folgendem Beispiel prüfen:

```
$security_key= "qundhft67dnft";
$url= "https://".$_SERVER["SERVER_NAME"].$_SERVER["REQUEST_URI"];
$url_without_checksum=
  substr($url, 0, strpos($url, "&ret_url_checksum") + 1);
if ($_REQUEST['ret_url_checksum'] !=
  md5($url_without_checksum.$security_key)) {
    // Error because hash does not match
}
else {
  // URL ok
}
```

Wo Sie die einzelnen Werte der Webserverumgebungsvariablen finden, hängt von der Konfiguration des Webservers ab. Im obigen Beispiel wurde ein Apache 2 verwendet. Wenn der Aufruf immer gleich ist, können Sie den Servernamen auch festlegen, ohne ihn dynamisch zu ermitteln. Die Rücksprungsadresse wird in dem Parameter `redirect_url` festgelegt.

Um das Beispiel zu testen, können Sie die folgende Rückspungs-URL aufrufen:

```
http://localhost/test_ret_url_checksum.php?param_x=1&param_y=2&ret_url_checksum=1daa0a7f275c9a917bfd8d7114813e3a
```

Nur diese URL ist gültig, bei jedem Manipulationsversuch wird eine Fehlermeldung zurückgeliefert.

Doppelzahlungen verhindern

Es kann passieren, dass Ihr Kunde zweimal auf den Bestellbutton klickt oder mit dem Zurück-Button des Browsers navigiert. Dadurch wird das Formular mit den Zahlungsdaten eventuell mehrmals an das ipayment-System abgesendet.

Das ipayment-System versucht, doppelte Transaktionen innerhalb von 2 Minuten zu erkennen und lehnt doppelte Anfragen mit einer Fehlermeldung ab. Wenn die doppelten Transaktions-

anfragen aber in einem Abstand von mehr als 2 Minuten eintreffen, werden sie als separate Transaktionen behandelt. Um das zu verhindern, können Sie bei einem Aufruf des ipayment-Systems im Parameter `shopper_id` eine eindeutige ID für diese Zahlung übergeben. Diese ID kann zum Beispiel eine eindeutige Warenkorb- oder Bestellnummer sein und bis zu 255 alphanumerische Zeichen enthalten. Damit die Erkennung funktioniert, müssen Sie zusätzlich den Parameter `advanced_strict_id_check` mit dem Wert `1` übergeben.

Wenn es bereits eine Transaktion mit den gleichen Daten und der gleichen `shopper_id` im ipayment-System gibt, wird die erneute Zahlungsanfrage nicht nochmals ausgeführt. Stattdessen werden die Transaktionsdaten der bereits im System vorhandenen Transaktion als Ergebnis zurückgegeben. Ihrer Anwendung wird also vorgespielt, dass die Transaktion erfolgreich verlaufen ist und es werden die gleichen Ergebnisdaten (zum Beispiel die Buchungsnummer) wie bei der ersten Anfrage zurückgegeben. So kann Ihr Kunde im normalen Bestellprozess weiter fortfahren, ohne dass Sie eine Ausnahmebehandlung in Ihrer Anwendung programmieren müssen. Zudem können keine Doppeltransaktionen mehr vorkommen.

Nur in dem Fall, dass der neue Zahlungsversuch andere Zahlungsdaten oder einen anderen Betrag als die bereits ausgeführte Transaktion aufweist, wird hier von ipayment ein Fehler generiert. In dem Fall war die ID nicht eindeutig.

Wenn Sie für eine Bestellung beim Aufruf von ipayment eine eindeutige ID verwenden, sollten Sie diese auch am Ende der Zahlung wieder überprüfen, falls Sie ipayment im **Normalen Modus** oder im **Silent-Modus** einsetzen. Wenn Sie diese Prüfung nicht vornehmen, besteht die Möglichkeit, Ihrem Shop mittels einer sogenannten „Replay-Attacke" nach einer erfolgreichen Zahlung weitere erfolgreiche Zahlungen für weitere Bestellungen vorzuspielen, indem Sie die Rücksprungparameter der ersten Bestellung benutzen. Durch Korrektheits-Prüfung der IDs und die gesicherte Rückmeldung über Hidden-Trigger-URLs können Sie solche Attacken verhindern.

Hidden-Trigger-URL

Im **Normalen Modus** und im **Silent-Modus** besteht die Gefahr, dass Zahlungen ausgeführt werden, ohne dass Ihr Kunde korrekt auf den Server zurückgeleitet wird. Dieser Fall kann aus verschiedenen Gründen vorkommen: Ihr Kunde kann zum Beispiel den Browser schließen, die Internet-Verbindung kann unterbrochen werden oder der Browser Ihres Kunden kann Probleme mit der korrekten Ausführung der Redirects haben. Wenn sich Ihre Anwendung zum Beispiel beim Generieren der Bestellung auf den Aufruf der im Parameter `redirect_url` angegebenen Erfolgs-URL verlässt, kann das zu erfolgreichen Zahlungen ohne einer zugehörigen Bestellung führen.

Solche Probleme können Sie vermeiden, indem Sie den sogenannten „Hidden Trigger" verwenden. Dazu übergeben Sie dem ipayment-System einfach eine Hidden-Trigger-URL. Ein durch diese URL aufgerufenes Script auf Ihrem Server kann zum Beispiel den Warenkorb speichern und die Bestellung auslösen, aber auch einen Downloadlink oder ein Passwort per E-Mail versenden. Auch andere Aktionen sind möglich. Die Logik der zu speichernden Bestellung wird von der Ergebnis-Seite in das Hidden-Trigger-Script verlagert. Die Ergebnisseite erzeugt stattdessen zum Beispiel nur noch die HTML-Ausgabe für Ihren Kunden, um ihn über den erfolgreichen Abschluss der Bestellung zu informieren.

Alternativ können im Shop-System Informationen zur erfolgreichen Transaktion gespeichert werden, damit Sie den Fall später prüfen können, falls der Redirect nicht erfolgreich war.

Die Hidden-Trigger-URL wird bei einer erfolgreichen Zahlung vom ipayment-Server aus aufgerufen. Das geschieht, bevor Ihr Kunde auf Ihren Server zurückgeleitet wird. Ihr Kunde bekommt nicht mit, dass das Hidden-Trigger-Script aufgerufen wird. Das Script sollte nur dann eine Aktion ausführen, wenn es vom einem der ipayment-Rechner aufgerufen wurde. Zusätzlich können Sie noch den im vorletzten Abschnitt vorgestellten Security-Hash für den Aufruf prüfen.

Das folgende Hidden-Trigger-Script schreibt eine Logzeile in eine Datei. Die Daten werden von ipayment per HTTP-POST gesendet. Deshalb ist es ausreichend, nur diese Daten auszuwerten. Es muss geprüft werden, ob der Aufruf tatsächlich vom ipayment-System und nicht von einem anderen Server kam.

Beispiel für ein Hidden-Trigger-Script

```
<?
  // check whether the call came from the ipayment system
  if (! preg_match('/\.ipayment\.de$/',
    gethostbyaddr($_SERVER["REMOTE_ADDR"])))
    exit();
  if ($_POST["ret_status"]!="SUCCESS")
    exit();
  // system variables usually always exists and are from type
  // Array, so we only check if the array consists of more
  // than 0 elements.
  if (count($_POST) > 0)
    $params= $_POST;
  else
    $params= array();
  // params (if there were submited any) are stored in
  // $params now
  $string="Payment on ".$params['ret_transdate']." at "
    $params['ret_transtime']." from ";
  // the name is stored in addr_name
  if (isset($params['addr_name']))
    $string.= $params['addr_name'];
  else
    $string.= "someone without name";
  // evaluate some other parameters (only a subset - there are
  // many more that can be used)
  $string.= " of "
    .sprintf("%.2f", $params['trx_amount'] / 100)
    ." ".$params['trx_currency']
    ." :: payment successful"
    ." transaction no ".$params['ret_trx_number'] : "")
    ."\n";
  // write the log line into a log file.
  $log_file= fopen("hidden_trigger.log", "a+");
  if ($log_file) {
    fputs ($log_file, $string);
    fclose ($log_file);
  }
?>
```

Als Ergebnis wird eine Logzeile zurückgegeben. Jede Zeile enthält einige Details aus der Transaktion. Die Logzeile hat das folgende Format:

```
  Payment on 18.09.03 at 11:42:17 from name of 129.89 EUR ::
  payment successful transaction no 1-10601273
```

Die Logzeile ist im Beispiel sehr vereinfacht. Wenn mehrere Benutzer gleichzeitig bezahlen, wird das Hidden-Trigger-Script auch von mehreren Prozessen vom ipayment System aus aufgerufen. Um in solchen Fällen keine Daten beim Schreiben zu verlieren, müssen Sie eine Dateisperre benutzen, so dass immer nur ein Prozess zu einem Zeitpunkt schreiben kann. Jede Programmiersprache bietet hierfür entsprechende Funktionen an.

Mehr Informationen zur Nutzung von Hidden-Trigger-URLs lesen Sie im Abschnitt *Gesicherte Rückmeldung erfolgreicher Transaktionen* ab Seite 41.

Einfaches Zuordnen der Auszahlungen

Mithilfe des Parameters `invoice_text` können Sie zusätzliche Referenzinformationen an die Zahlungsanbieter übertragen. Diese Referenzinformationen werden von den Zahlungsanbietern meistens auf den Auszahlungs-Belegen ausgedruckt bzw. teilweise auch direkt an Ihren Kunden weitergeleitet. Sie können somit eine Bestell- oder Rechnungsnummer an ipayment übergeben. Dadurch ist es einfach möglich, die späteren Auszahlungen den ausgeführten Transaktionen im ipayment-System und den Bestellungen in Ihren Systemen zuzuordnen.

Vollständiges Formular der empfohlenen Parameter

Das Beispiel-Formular für den **Silent-Modus** aus dem letzten Kapitel kann mit den zuvor genannten Sicherheitsoptionen und Parametern erweitert werden. Im untenstehenden Beispiel haben wir als Transaktions-Security-Key das Wort „testtest“ und den entsprechenden Test-Account verwendet.

Beispiel für ein vollständiges Formular mit den empfohlenen Parametern

```
<form method="post"
      action="https://ipayment.de/merchant/99999/processor/2.0/">
  <!—- Base Parameter -->
  <input type="hidden" name="trxuser_id" value="99998">
  <input type="hidden" name="trxpassword" value="0">

  <!—- Amount and Currency for Payment -->
  <input type="hidden" name="trx_amount" value="12989">
  <input type="hidden" name="trx_currency" value="EUR">

  <!—- Paymenttyp: CC to make creditcard payment -->
  <input type="hidden" name="trx_paymenttyp" value="cc">

  <!—- URL and Parameter for Redirect back to Shop -->
  <input type="hidden" name="redirect_url"
         value="https://your_domain/payment_success.php">
  <input type="hidden" name="silent" value="1">
  <input type="hidden" name="silent_error_url"
         value="https://your_domain/enter_payment_data.php">

  <!—- Hidden-Trigger URL -->
  <input type="hidden" name="hidden_trigger_url"
         value="https://your_domain/hidden_trigger.php">

  <!—- Security Hash for Parameter -->
  <input type="hidden" name="trx_securityhash"
         value="237701009c405a243ac821343cfbf9ec">

  <!—- Unique shopper_id and enable Advanced-ID-Check mode -->
  <input type="hidden" name="shopper_id"
         value="<some_unique_id>">
  <input type="hidden" name="advanced_strict_id_check" value="1">

  <!—- Invoice-text for payment -->
  <input type="hidden" name="invoice_text"
         value="<invoice_number>">

  <!—- Credit card data fields -->
  Cardholder name: <input type="text" name="addr_name"
                          value=""><br>
  Credit card no.: <input type="text" name="cc_number"
                          value=""><br>
  Card Check Code: <input type="text" name="cc_checkcode"
                          value=""><br>
```

```
  Card Expire date: <select name="cc_expdate_month">
                      <option>01</option>
                      <option>02</option>
                      <option>03</option>
                      <option>04</option>
                      <option>05</option>
                      <option>06</option>
                      <option>07</option>
                      <option>08</option>
                      <option>09</option>
                      <option>10</option>
                      <option>11</option>
                      <option>12</option>
  </select>
   / 
  <select name="cc_expdate_year">
                      <option>2007</option>
                      <option>2008</option>
                      <option>2009</option>
                      <option>2010</option>
                      <option>2011</option>
                      <option>2012</option>
  </select><br>

  <!—- Submit Button -->
  <input type="submit" name="form_submit"
    value="Process payment">
</form>
```

Vorgenerierte Session zur besseren Absicherung gegen Manipulationen

Neben der Absicherung der Basisparameter über den Security-Hash können Sie auch mit einem vorgenerierten Hash arbeiten. Rufen Sie dazu die Webservice-Methode `createSession` mit den statischen Parametern auf.

So könnte ein solcher Aufruf aussehen:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <createSession>
      <accountData>
        <accountId>99999</accountId>
        <trxuserId>99999</trxuserId>
        <trxpassword>0</trxpassword>

        <adminactionpassword>
          5cfgRT34xsdedtFLdfHxj7tfwx24fe</adminactionpassword>
      </accountData>
      <transactionData>
        <trxAmount>117</trxAmount>
        <trxCurrency>EUR</trxCurrency>
      </transactionData>
      <transactionType>auth</transactionType>
      <paymentType>cc</paymentType>
      <processorUrls>
        <redirectUrl>
          https://your-domain/payment_success.php
```

```
      </redirectUrl>
      <silentErrorUrl>
        https://your-domain/enter_payment_data.php
      </silentErrorUrl>
    </processorUrls>
  </createSession>
 </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Die Antwort mit der Session-ID sieht so aus:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <ns1:createSessionResponse
      xmlns:ns1="https://ipayment.de/service_v3/binding">
      <sessionId>ac131118BWlJuaXMhlQQk3bGSJ7djPV0</sessionId>
    </ns1:createSessionResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Mit dieser generierten Session-ID ist das benutzte Formular um einiges einfacher:

```
<form method="post"
     action="https://ipayment.de/merchant/99999/processor/2.0/">
<!—- Session-ID and integration mode -->
<input type=”hidden” name=”ipayment_session_id”
     value=”ac131118Jo05tyKqUkWEjvbafBoWfEk0”>
<input type="hidden" name="silent" value="1">
<!—- Credit card data fields -->
Cardholder name:
<input type="text" name="addr_name" value=""><br>
Credit card no.:
<input type="text" name="cc_number" value=""><br>
Card Check Code:
<input type="text" name="cc_checkcode" value=""><br>
Card Expire date:
<select name="cc_expdate_month">
  <option>01</option>
  <option>02</option>
  <option>03</option>
  <option>04</option>
  <option>05</option>
  <option>06</option>
  <option>07</option>
  <option>08</option>
  <option>09</option>
  <option>10</option>
  <option>11</option>
  <option>12</option>
</select>
   
<select name="cc_expdate_year">
  <option>2007</option>
  <option>2008</option>
  <option>2009</option>
  <option>2010</option>
```

```
  <option>2011</option>
  <option>2012</option>
</select><br>
<!—- Submit Button -->
<input type="submit" name="form_submit" value="Process payment">
</form>
```

Rückgabewerte (Betrugserkennung)

Die Rückgabeparameter im **Normalen Modus** und **Silent-Modus** werden an die Rücksprungs-URL per `GET` oder `POST` angehängt. Um an die einzelnen Werte zu gelangen, müssen die entsprechenden Umgebungsvariablen ausgewertet werden.

Eine Rücksprungs-URL kann zum Beispiel so aussehen:

```
https://your_domain/script.php?trx_currency=EUR&trx_typ=auth&
trx_paymenttyp=cc&trx_amount=1275&trxuser_id=99999&
addr_name=Hans+Mustermann&ret_transdate=29.08.07&

ret_transtime=10%3A19%3A54&ret_errorcode=0&ret_authcode=H67D8&
ret_ip=212.227.116.72&ret_trx_number=1-20630916&redirect_needed=0&
trx_paymentmethod=AmexCard&trx_paymentdata_country=US&
trx_remoteip_country=DE&ret_status=SUCCESS
```

Wenn beim Aufruf der Zahlung der Parameter `return_paymentdata_details` mit dem Wert 1 übergeben wurde, werden zusätzlich noch die maskierten Zahlungsdetails des Besuchers zurückgegeben:

```
https://your_domain/script.php?trx_currency=EUR&trx_typ=auth&
trx_paymenttyp=cc&trx_amount=1278&trxuser_id=99999&
addr_name=Hans+Mustermann&ret_transdate=29.08.07&
ret_transtime=10%3A28%3A24&ret_errorcode=0&ret_authcode=H67D8&
ret_ip=212.227.116.72&ret_trx_number=1-20631050&redirect_needed=0&
trx_paymentmethod=AmexCard&trx_paymentdata_country=US&
trx_remoteip_country=DE&paydata_cc_cardowner=Hans+Mustermann&
paydata_cc_number=XXXXXXXXXXX8431&paydata_cc_expdate=0608&
paydata_cc_typ=AmexCard&ret_status=SUCCESS
```

Die folgenden Parameter sind besonders wichtig. Sie sollten bei jeder Bestellung überprüft, bzw. als Zahlungsdetails in Ihren Systemen gespeichert werden. In bestimmten Fällen sind einzelne Werte nicht vorhanden. Beachten Sie zusätzlich die Informationen zu den einzelnen Parametern im Kapitel *Rückgabeparameter Zahlungsabwicklung* ab Seite 60.

Parameter	Beschreibung
`ret_status`	An dem Parameter kann einfach abgelesen werden, ob die Transaktion erfolgreich war (`SUCCESS`), oder es einen Fehler gegeben hat (`ERROR`). In speziellen Fällen kommt hier auch der Wert `REDIRECT` zurück, wenn Ihr Kunde zu einer anderen Seite weitergeleitet werden muss.
`ret_errorcode`	Übermittlung des Fehlercodes. Ein Fehler tritt immer dann auf, wenn der Wert des Parameters nicht 0 lautet. Ihre Anwendung sollte diesen Parameter prüfen und nur korrekte Aufrufe akzeptieren. Eine komplette Liste aller Fehlercodes finden Sie unter http://ipayment.de > **Technik**.
`trx_amount, trx_currency`	Betrag in der kleinsten Währungseinheit und Währung der Transaktion. Diese Werte sollten mit den Werten übereinstim-

Parameter	Beschreibung
	men, die zur Zahlungsabwicklung an ipayment übergeben wurden.
ret_trx_number	Transaktionsnummer für die aktuelle Zahlung im ipayment-System. Mit dieser Nummer wird die Zahlung eindeutig identifiziert. Dieser Wert sollte in Ihrem System gemeinsam mit der Bestellung abgespeichert werden. Die Transaktionsnummer wird benötigt, um die Transaktion im ipayment-System zu finden bzw. um weitere Aktionen durchzuführen, wie zum Beispiel Abbuchungen oder Stornierungen.
ret_authcode	Autorisierungsnummer, die vom Zahlungsanbieter zurückgegeben wurde. Diese Nummer erscheint normalerweise auch auf den Monatsabrechnungen, die Sie von Ihrem Zahlungsanbieter erhalten. Dadurch können Sie die Zahlungen entsprechend zuordnen.
addr_street, addr_city, addr_zip, addr_country, addr_street2, addr_state, addr_telefon, addr_telefax	Adressdaten des Kunden. Diese Daten sind nur interessant, wenn Sie diese vorher nicht schon in Ihrem Bestellablauf erfasst haben, bzw. wenn Ihr Kunde diese bei der Zahlung über ipayment ändern kann.
paydata_cc_number, paydata_cc_expdate, paydata_cc_typ, ...	Verwendete Kreditkartennummer (maskiert), Ablaufdatum und Kartentyp der Zahlung. Durch diese Werte können Sie die ausgezahlten Beträge später besser zuordnen. Welche Parameter zurückgegeben werden, ist von der Zahlungsart abhängig.
ret_ip	Die IP des Kunden. Dieser Wert sollte mit der Bestellung gespeichert werden, um bei Betrugsfällen zusätzliche Informationen zu haben.
trx_remoteip_country	Das Land, in dem die IP des Kunden registriert ist. Dieses sollte mit dem Land der Adresse und eventuell auch mit Land der Zahlungsdaten übereinstimmen.
trx_paymentdata_country	Das Land, aus dem die Zahlungsdaten stammen. Bei der Kreditkarte das Land, in dem die Karte ausgegeben wurde. Dieses sollte mit dem Land der Adresse und eventuell auch mit dem Land der IP übereinstimmen.
trx_issuer_avs_response	Ergebnis der AVS-Prüfung. Der Ergebniswert gibt zusätzliche Hinweise, ob die Adresse zur angegebenen Kreditkarte passt. Voraussetzung für den Einsatz der AVS-Prüfung ist die Angabe der kompletten Adresse und die Unterstützung des AVS-Verfahrens von Zahlungspartner und der kartenausgebenden Bank.
trx_payauth_status	Status der 3-D-Secure-Prüfung. Wenn keine Prüfung möglich war (Wert U), sollten Sie die Transaktion genauer prüfen, da keine Haftungsumkehr besteht.

Vor allem die fünf zuletzt genannten Rückgabewerte geben sehr wichtige Informationen über die Bestellung. Deshalb sollten sie unbedingt gespeichert und auch genau geprüft werden. Wenn die Werte der verschiedenen Länderangaben stark abweichen, sollte die Zahlung und die bestellten Waren oder Dienstleistungen genauestens geprüft werden. Zusätzlich sollten Sie

versuchen, den Käufer zu kontaktieren oder die Adressdaten mit dem Zahlungsanbieter zu klären.

Benutzen Sie die AVS- und 3-D-Secure-Werte ebenfalls, um Anhaltspunkte zu erhalten, ob eine Transaktion bzw. Bestellung genau geprüft werden soll. Wenn Sie diese Werte gemeinsam mit der Bestellung speichern, können Sie Bezahldaten mit der Bestellung verknüpfen und somit gezielt auffällige Bestellungen auf Betrugsverdacht prüfen. Weitere Informationen zur Prüfung von Transaktionen und Erkennung von betrügerischen Zahlungen finden Sie in der ipayment-FAQ unter http://faq.ipayment.de > **Anwendungen – Fragen zur Nutzung von ipayment** > **Welche Anzeichen können auf Betrugsabsichten hinweisen?**

Die in diesem Abschnitt genannten Daten können auch zu einem späteren Zeitpunkt aus dem ipayment-System exportiert werden.

Backend-Aktionen

Die Schnittstellen des ipayment-Systems können sowohl aus einer Webanwendung, als auch aus dem Backend-Umfeld angesprochen werden. Eine Backend-Aktion kann zum Beispiel eine Verbuchung von Vorautorisierungen direkt vor dem Versand der Ware sein, oder das Rückgängigmachen einer Zahlung durch Stornierung oder Gutschrift. Backend-Aktionen können Sie durchführen, ohne dass eine Interaktion durch Ihren Kunden stattfindet. Das ipayment-System bietet dazu verschiedene Schnittstellen an.

SOAP-Webservice

Mit Hilfe der verfügbaren WSDL-Beschreibung des SOAP-Webservices können die Funktionen des Webservices in den meisten Programmiersprachen sehr einfach aufgerufen werden.

Der SOAP-Webservice eignet sich besonders für die Ausführung von Folgeaktionen zu Transaktionen, wenn keine sensiblen Zahlungsdaten übertragen werden müssen. Wenn Sie auch sensible Zahlungsdaten übertragen möchten, muss Ihr System nach den PCI-DSS-Regeln zertifiziert werden. Deshalb ist es wichtig, dass Sie den SOAP-Webservice nur für Aktionen einsetzen, bei denen keine sensiblen Daten übertragen und verarbeitet werden. Mehr Informationen zu PCI DSS finden Sie im Kapitel *Sicherheit für Zahlungen* ab Seite 11.

Das folgende Beispiel stellt eine mögliche Abfolge von Backend-Aktionen dar. Dabei wird davon ausgegangen, dass zu einer Bestellung in einem Online-Shop eine Zahlung über 12,76 EUR vorautorisiert wurde (`trx_typ=preauth`) und diese Zahlung nun tatsächlich verbucht werden soll (`capture`). Die Transaktionsnummer der im Shop ausgeführten Transaktion lautet 1-18232427 und muss als originale Transaktionsnummer angegeben werden. Wenn der gesamte Betrag der Autorisierung abgebucht werden soll, muss der Betrag nicht übergeben werden.

Beispiel für eine SOAP-Anfrage (wichtige Felder sind fett markiert)

Keine manuellen Zeilenumbrüche verwenden

Die Zeilenumbrüche innerhalb der Tags dienen nur der besseren Veranschaulichung der Beispiele. Achten Sie darauf, dass Sie in Ihrer Anwendung keine manuellen Zeilenumbrüche innerhalb der Tags einfügen, da diese als Sonderzeichen übermittelt werden und somit zum Beispiel das `adminactionpassword` nicht mehr stimmt.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
```

```
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <capture>
      <accountData>
        <accountId>99999</accountId>
        <trxuserId>99999</trxuserId>
        <trxpassword>0</trxpassword>
        <adminactionpassword>
           5cfgRT34xsdedtFLdfHxj7tfwx24fe</adminactionpassword>
      </accountData>
      <origTrxNumber>1-25949395</origTrxNumber>
      <transactionData>
        <trxAmount>119</trxAmount>
        <trxCurrency>EUR</trxCurrency>
      </transactionData>
    </capture>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Beispiel für eine Antwort auf diese Anfrage

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <ns1:captureResponse
      xmlns:ns1="https://ipayment.de/service_v3/binding">
      <ipaymentReturn>
        <status>SUCCESS</status>
        <successDetails>
          <retTransDate>25.07.08</retTransDate>
          <retTransTime>17:08:08</retTransTime>
          <retTrxNumber>1-25949407</retTrxNumber>
          <retAuthCode></retAuthCode>
        </successDetails>
        <addressData>
          <addrStreet>Ernst-Frey-Str. 9</addrStreet>
          <addrCity>Karlsruhe</addrCity>
          <addrZip>76135</addrZip>
          <addrCountry>DE</addrCountry>
        </addressData>
        <addresscheckResult>UNCHECKED</addresscheckResult>
        <paymentMethod>VisaCard</paymentMethod>
        <trxPaymentDataCountry>US</trxPaymentDataCountry>
      </ipaymentReturn>
    </ns1:captureResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Die Verbuchung wurde ausgelöst und im Beispiel unter der Transaktionsnummer 1-20632827 im ipayment-System gespeichert. Speichern Sie die Transaktionsnummer auf jeden Fall, da Sie diese für weitere Abfragen verwenden können.

In unserem Beispiel sendet der Käufer nach ein paar Tagen einen der gelieferten Artikel zurück. Der Teilbetrag für diesen Artikel (in dem Fall 10,71 EUR) soll nun erstattet werden.

Die Stornierung muss auf die zuvor verbuchte Transaktion erfolgen, nicht auf die Vorautorisierung. Dazu können Sie folgende SOAP-Nachricht verwenden:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV=http://schemas.xmlsoap.org/soap/envelope/
  xmlns:xsd=http://www.w3.org/2001/XMLSchema
  xmlns:xsi=http://www.w3.org/2001/XMLSchema-instance>
  <SOAP-ENV:Body>
    <refund>
      <accountData>
        <accountId>99999</accountId>
        <trxuserId>99999</trxuserId>
        <trxpassword>0</trxpassword>
        <adminactionpassword>
          5cfgRT34xsdedtFLdfHxj7tfwx24fe</adminactionpassword>
      </accountData>
      <origTrxNumber>1-25778595</origTrxNumber>
      <transactionData>
        <trxAmount>1071</trxAmount>
        <trxCurrency>EUR</trxCurrency>
      </transactionData>
      <options></options>
    </refund>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Die Antwort auf diese Anfrage sieht folgendermaßen aus:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <SOAP-ENV:Body>
     <ns1:refundResponse
xmlns:ns1="https://ipayment.de/service_v3/binding">
       <ipaymentReturn>
         <status>SUCCESS</status>
         <successDetails>
           <retTransDate>16.07.08</retTransDate>
           <retTransTime>11:05:05</retTransTime>
           <retTrxNumber>1-25791685</retTrxNumber>
           <retAuthCode></retAuthCode>
         </successDetails>
         <paymentMethod>VisaCard</paymentMethod>
         <trxPaymentDataCountry>US</trxPaymentDataCountry>
       </ipaymentReturn>
     </ns1:refundResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

In diesem Fall wurde nur ein Teilstorno durchgeführt. Deshalb können Sie dem Kunden noch weitere Gutschriften gewähren. Insgesamt können maximal 115% des verbuchten Betrags gutgeschrieben werden. Sollte ein weiterer Teilstorno versucht werden, bei dem der Gesamtbetrag überschritten wird, kommt es zu folgendem Fehler:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <SOAP-ENV:Body>
     <ns1:refundResponse
xmlns:ns1="https://ipayment.de/service_v3/binding">
       <ipaymentReturn>
         <status>ERROR</status>
         <errorDetails>
           <retErrorcode>10033</retErrorcode>
           <retFatalerror>0</retFatalerror>
           <retErrorMsg>Refund nicht möglich.</retErrorMsg>
           <retAdditionalMsg>Not enough funds left (17) for
            this refund.</retAdditionalMsg>
         </errorDetails>
       </ipaymentReturn>
     </ns1:refundResponse>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Der ursprüngliche Betrag, der in der Vorautorisierung reserviert und abgebucht wurde, betrug 12,76 EUR. Daraufhin wurden 10,71 EUR storniert. Da vom Gesamtbetrag insgesamt 115% gutgeschrieben werden können, also maximal 14,67 EUR, sind noch 3,96 EUR offen, die dem Kunden zurückerstattet werden könnten.

Sie können einstellen, wie lange der SOAP-Client auf die Antwort des Servers wartet. Stellen Sie hier mindestens 300 bis 600 Sekunden als Timeout ein. Die Antworten vom ipayment-Server kommen im Normalfall innerhalb weniger Sekunden, es kann allerdings in seltenen Fällen auch zu längeren Zeiten kommen, die 5 bis 10 Minuten betragen können.

Wenn das Timeout für die Kommunikation zu kurz eingestellt ist, kann es passieren, dass Sie zu einer erfolgreichen Aktion keine Antwort erhalten.

Zur Integration des Webservices in Ihre Anwendungen benötigen Sie einen SOAP-Client für Ihre Programmiersprache. Diese sind teilweise bereits in den Programmiersprachen integriert oder lassen sich einfach integrieren. Für PHP existiert zum Beispiel neben der in PHP 5 enthaltenen SOAP-Erweiterung die SOAP-Library „nusoap“ (http://sourceforge.net/projects/nusoap/). Für andere Programmiersprachen gibt es vergleichbare Bibliotheken.

Gateway-Modus

Der **Gateway-Modus** funktioniert ähnlich wie der Webservice, mit dem Unterschied, dass hier keine SOAP-Nachrichten im XML-Format ausgetauscht werden. Stattdessen werden die Daten per `HTTP POST` direkt an den ipayment-Server gesendet. Das Ergebnis wird als Body der HTTP-Antwort zurückgegeben.

Im **Gateway-Modus** werden alle Daten, die an ipayment gesendet werden, von Ihrem Server verarbeitet und übertragen. Deshalb eignet sich diese Methode vor allem für Backend-Aktionen, weil bei diesen keine sensiblen Zahlungsdaten übertragen werden müssen. In der Regel werden nur die Transaktionsnummern aus dem ipayment System verwendet.

Der Aufruf zur Stornierung der Transaktion 1-20632827 aus dem vorhergehenden Kapitel sieht für den Gateway-Modus wie folgt aus. Die wichtigen Parameter sind fett markiert:

```
gateway=1&trxuser_id=99999&trxpassword=0&
adminactionpassword=5cfgRT34xsdedtFLdfHxj7tfwx24fe&
```

```
trx_typ=refund_cap&orig_trx_number=1-20632827&trx_amount=1071&
trx_currency=EUR
```

Die Kommunikation mit dem ipayment-Server muss SSL-gesichert erfolgen. Dazu gibt es auf den meisten Systemen Hilfsprogramme, wie zum Beispiel „curl“ oder andere Bibliotheken, die eine SSL-gesicherte Kommunikation ermöglichen. Je nach Programmiersprache gibt es verschiedene Möglichkeiten, wie die Kommunikation aussehen kann. In unserer FAQ finden Sie zwei dieser Möglichkeiten für PHP unter http://ipayment.de > **Fragen zur individuellen Anbindung von ipayment** > **Wie benutzt man Curl mit PHP**. Außerdem sind dort weitere Parameter und mögliche Problemlösungen bei der Integration beschrieben.

Die Rückgabe könnte im Erfolgsfall so aussehen:

```
Status=0
Params=trxuser_id=99999&trx_typ=refund_cap&trx_amount=1071&
trx_currency=EUR&trx_paymenttyp=cc&ret_transdate=20.07.08&
ret_transtime=12%3A37%3A31&ret_errorcode=0&ret_authcode=&
ret_ip=127.0.0.1&ret_trx_number=1-20633075&redirect_needed=0&
trx_paymentmethod=AmexCard&trx_paymentdata_country=US&
addr_name=Hans+Mustermann&addr_street=Ernst-Frey-Str.+9&
addr_zip=76135&addr_city=Karlsruhe&addr_country=DE&addr_email=support%40
ipayment.de&addr_check_result=UNCHECKED&ret_status=SUCCESS
```

Im Fehlerfall könnte die folgende Rückmeldung auftreten:

```
Status=-1
Params=trxuser_id=99999&trx_typ=refund_cap&trx_amount=1071&
trx_currency=EUR&trx_paymenttyp=cc&ret_errorcode=10033&ret_fatalerror=0&
ret_errormsg=Refund+nicht+m%F6glich.&
ret_additionalmsg=Not+enough+funds+left+%280%29+for+this+refund.&
ret_ip=127.0.0.1&redirect_needed=0&ret_status=ERROR
```

Das Ergebnis ist auf zwei Zeilen verteilt. In der ersten Zeile steht der Status der Transaktion, nämlich erfolgreich (Status=`0`) oder fehlerhaft (Status=`-1`). In der zweiten Zeile stehen alle Rückgabewerte als Parameterstring zusammengefasst und URL-kodiert.

Da im **Gateway-Modus** die Kommunikation direkt stattfindet, muss sichergestellt sein, dass das verwendete Kommunikations-Timeout in Ihrem System ausreicht. Tragen Sie mindestens 300 bis 600 Sekunden ein.

Im **Gateway-Modus** kann auch eine Hidden-Trigger-URL verwendet werden. In der Regel ist das allerdings nicht notwendig, da die Kommunikation direkt erfolgt. Sie können jedoch über ein Hidden-Trigger-Script die erfolgreichen Aktionen loggen, so dass Sie bei einem Timeout das Ergebnis einer solchen Anfrage im Nachhinein prüfen können.

Fortgeschrittene Aktionen und Anwendungsfälle

Die bisher erklärten Integrationsmethoden werden bei den folgenden Anwendungsfällen so kombiniert, dass sichere und zertifizierungsfreie Abläufe entstehen. Dadurch sind auch komplexe Anwendungsfälle sicher umsetzbar.

Regelmäßige Zahlungen

Regelmäßige Zahlungen sind Vorgänge, bei denen Ihr Kunde nicht nur einmal bezahlt, sondern mehrere Zahlungen leistet. Das ist zum Beispiel bei Abonnements der Fall. Dabei wird in regelmäßigen Abständen ein Betrag fällig, bis das Abonnement erlischt. Zusätzlich gibt es auch noch Ratenzahlungen, bei denen im Voraus festgelegt wird, wie lange eine bestimmte Zah-

lung geleistet werden muss. Regelmäßige Zahlungen und Ratenzahlungen werden gegenüber den Banken entsprechend gekennzeichnet, so dass die Banken eine solche Zahlung sofort von Einmalzahlungen unterscheiden können.

Damit Sie regelmäßige Zahlungen durchführen können, müssen die Kunden- und Zahlungsdaten gespeichert werden. Dabei ist es wichtig, wo diese Daten gespeichert werden. Damit die Daten nicht auf Ihrem Server lagern müssen, kann das ipayment-System die Speicherung dieser sensiblen Zahlungsdaten übernehmen. Dadurch bleibt Ihnen die Zertifizierung nach den PCI-DSS-Regeln erspart.

Anwendungsfall „monatliche Abonnement-Zahlungen“

Im folgenden Beispiel zeigen wir Ihnen, wie Sie ein Abonnement starten können.

Ein Kunde bestellt auf einer Website einen Service, der monatlich 12 Euro kostet. Auf der Website gibt der Kunde seine Kreditkartendaten einmalig bei der Anmeldung ein. Später muss der Händler die monatlichen Beträge einzeln über ein Script einziehen.

Bei einem Abonnement müssen zuerst die Zahlungsdaten erfasst werden. Diese Erfassung erfolgt auf der Website unter Verwendung des **Silent-Modus**, da bei diesem die Daten nur vom ipayment-System verarbeitet werden. Zuerst überprüft das System mit einer Testbuchung, ob die Kreditkarte gültig ist und eine Autorisierung erfolgen kann. In unserem Beispiel haben wir für die Testbuchung einen Testbetrag von 5 Euro verwendet. Für die Testbuchung müssen Sie den Transaktionstyp **check_save** setzen.

Das Formular ist ähnlich aufgebaut wie das Formular für einmalige Zahlungen. Es werden jedoch einige Parameter hinzugefügt, um dem ipayment-System mitzuteilen, dass mit diesen Daten regelmäßige Zahlungen abgewickelt werden sollen. Diese zusätzlichen Parameter sind im Beispiel fett hervorgehoben.

```
<form method="post"
      action="https://ipayment.de/merchant/99999/processor/2.0/">
  <input type="hidden" name="trxuser_id" value="99999">
  <input type="hidden" name="trxpassword" value="0">
  <input type="hidden" name="trx_paymenttyp" value="cc">
  <input type="hidden" name="redirect_url"
      value="https://your_domain/payment_success.php">
  <input type="hidden" name="silent" value="1">
  <input type="hidden" name="silent_error_url"
      value="https://your_domain/enter_payment_data.php">
  <input type="hidden" name="hidden_trigger_url"
      value="https://your_domain/hidden_trigger.php">
  <input type="hidden" name="shopper_id"
      value="<some_unique_id>">
  <input type="hidden" name="advanced_strict_id_check" value="1">

  <input type="hidden" name="trx_amount" value="500">
  <input type="hidden" name="trx_currency" value="EUR">

  <input type="hidden" name="trx_typ" value="check_save">
  <input type="hidden" name="recurring_typ" value="initial">
  <input type="hidden" name="recurring_frequency" value="28">
  <input type="hidden" name="recurring_allow_expiry_correction"
      value="1">

  <!—- Credit card data fields -->
  …
  <!—- Submit Button -->
  <input type="submit" name="form_submit"
      value="Process payment">
</form>
```

Nach der Eingabe und Prüfung der Kreditkartendaten erhält Ihr System als Rückgabewert des Parameters `redirect_url` eine Transaktionsnummer, die innerhalb des Parameters `ret_trx_number` übermittelt wird. Speichern Sie diese Transaktionsnummer anstelle der Zahlungsdaten gemeinsam mit Ihren Kundendaten ab. Basierend auf dieser Transaktionsnummer können Sie alle monatlichen Abrechnungen für diesen Kunden durchführen.

Der Wert 28 im Parameter `recurring_frequency` bedeutet, dass zwischen den einzelnen regelmäßigen Zahlungen mindestens 28 Tage vergehen müssen. Wenn Sie vor Ablauf dieser 28 Tage versuchen, eine solche Zahlung auszuführen, erhalten Sie vom ipayment-System eine Fehlermeldung.

Die Folgezahlungen können Sie mithilfe des SOAP-Webservices durchführen. Ein Aufruf könnte zum Beispiel so aussehen:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <reAuthorize>
      <accountData>
        <accountId>99999</accountId>
        <trxuserId>99999</trxuserId>
        <trxpassword>0</trxpassword>
        <adminactionpassword>
          5cfgRT34xsdedtFLdfHxj7tfwx24fe</adminactionpassword>
      </accountData>
      <origTrxNumber>1-25949836</origTrxNumber>
      <transactionData>
        <trxAmount>119</trxAmount>
        <trxCurrency>EUR</trxCurrency>
        <recurringData>
          <recurringTyp>sequencial</recurringTyp>
        </recurringData>
      </transactionData>
      <options></options>
    </reAuthorize>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Als Ergebnis wird eine neue Transaktion ausgeführt. Diese kann erfolgreich ablaufen oder eine Fehlermeldung zurückgeben. Ihr Script muss entsprechend auf das Ergebnis der Transaktion reagieren können.

Wenn die Kosten für das Abonnement zum Beispiel halbjährlich bezahlt werden sollen, muss bei der ersten Anfrage der Parameter `recurring_frequency` mit dem Wert 168 (6*28 Tage) angegeben werden

Beachten Sie: Das ipayment-System speichert die Transaktions- und Zahlungsdaten standardmäßig maximal 3 Monate. Da für eine Folgezahlung die Zahlungsdaten der letzten erfolgreichen Zahlung benötigt werden, ist damit standardmäßig die maximale Abonnement-Frequenz auf 3 Monate begrenzt. Wenn Sie eine längere Abonnement-Frequenz benötigen, können Sie die maximale Datenspeicherzeit auf bis zu 12 Monate verlängern lassen. Längere Abstände zwischen den einzelnen wiederkehrenden Zahlungen sind nur durch die Nutzung des Storage-Services möglich.

Der Storage-Service

Neben der reinen Zahlungsabwicklung kann das ipayment-System auch sensible Zahlungsdaten sicher speichern. Die Zahlungsdaten können dabei mit Hilfe der Storage-ID angesprochen werden, die bei allen Aufrufen zur Abwicklung von Zahlungen anstelle der Zahlungsdaten verwendet werden kann. Das ipayment-System liest die unter dieser ID gespeicherten Daten aus einer internen Datenbank und benutzt diese für die Zahlungen.

Anwendungsfall „Kurzfristiges Speichern der Zahlungsdaten für einen E-Shop“

Ein mögliches Beispiel ist die kurzfristige Speicherung der Kartendaten. Das kann zum Beispiel für die Dauer eines Bestellvorgangs sein, um die Dateneingabe und Ausführung der Zahlung in einem Online-Shop trennen zu können. In den meisten Online-Shops wird nach der Eingabeseite für die Zahlungsdaten vor dem Abschluss der Bestellung noch die Zusammenfassung aller Informationen angezeigt. Die Zahlung soll nach Bestätigung dieser Informationen durch den Kunden ausgeführt werden.

Wenn die Zahlungsdaten vorher eingegeben werden sollen, bedeutet das im Standardfall, dass der Online-Shop die Kreditkartendaten zwischenspeichern oder zumindest kurz verarbeiten muss. Dadurch entsteht eine Pflicht zur PCI-DSS-Zertifizierung. Das können Sie umgehen, indem Sie den Storage-Service verwenden. Bei der Eingabe der Zahlungsdaten wird der Datenspeicher angelegt. Sobald die Bestellung bestätigt wird, werden die Daten aus dem Datenspeicher verwendet, um eine Transaktion auszulösen. Die Zahlungsdaten werden dabei nicht noch einmal an ipayment übermittelt.

Im Beispiel verwenden wir den sicheren **Silent-Modus**. Die Daten werden direkt an das ipayment-System gesendet, ohne vorher auf Ihrem System zwischengespeichert oder verarbeitet zu werden. Mit dem Transaktionstyp `base_check` werden die übergebenen Daten auf Plausibilität geprüft. Alternativ könnten Sie auch wie im letzten Beispiel den Transaktionstyp `check_save` verwenden, um die Daten noch genauer zu prüfen. Die zusätzlichen Parameter sind im folgenden Beispiel fett hervorgehoben:

```
<form method="post"
      action="https://ipayment.de/merchant/99999/processor/2.0/">
  <input type="hidden" name="trxuser_id" value="99999">
  <input type="hidden" name="trxpassword" value="0">
  <input type="hidden" name="trx_paymenttyp" value="cc">
  <input type="hidden" name="redirect_url"
      value="https://your_domain/payment_success.php">
  <input type="hidden" name="silent" value="1">
  <input type="hidden" name="silent_error_url"
      value="https://your_domain/enter_payment_data.php">
  <input type="hidden" name="hidden_trigger_url"
      value="https://your_domain/hidden_trigger.php">
  <input type="hidden" name="shopper_id"
      value="<some_unique_id>">
  <input type="hidden" name="advanced_strict_id_check" value="1">

  <input type="hidden" name="trx_amount" value="500">
  <input type="hidden" name="trx_currency" value="EUR">

  <input type="hidden" name="trx_typ" value="base_check">
  <input type="hidden" name="return_paymentdata_details"
      value="1">
  <input type="hidden" name="use_datastorage" value="1">
  <input type="hidden" name="datastorage_expirydate"
       value="2007/09/25 14:30:00">

  <!—- Credit card data fields -->

  ...
```

```
  <!—- Submit Button -->
  <input type="submit" name="form_submit" value="Process payment">
</form>
```

Das Ablaufdatum des Datenspeichers (`datastorage_expirydate`) kann zum Beispiel auf zwei Stunden oder einen Tag gesetzt werden. Nach Ablauf dieser Zeit werden die Daten gelöscht.

Durch Nutzung des Parameters `return_paymentdata_details` werden die maskierten Zahlungsdetails an den Online-Shop zurückgegeben. Diese Daten können gemeinsam mit dem Verfallsdatum auf der Bestellübersichts-Seite angezeigt werden. Die Verarbeitung und Speicherung dieser Daten ist sicherheitstechnisch unbedenklich.

Zum Ausführen der Zahlung beim Abschließen der Bestellung benutzen wir im Beispiel wieder den SOAP-Webservice. Anstelle der Kreditkartendaten wird die Storage-ID des Datenspeichers übergeben.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:ns7156="https://ipayment.de/service_v3/binding">
  <SOAP-ENV:Body>
    <preAuthorize>
      <accountData>
        <accountId>99999</accountId>
        <trxuserId>99999</trxuserId>
        <trxpassword>0</trxpassword>
        <adminactionpassword>
          5cfgRT34xsdedtFLdfHxj7tfwx24fe</adminactionpassword>
      </accountData>
      <paymentData>
        <storageData>
          <fromDatastorageId>3873222</fromDatastorageId>
          <expireDatastorage>true</expireDatastorage>
        </storageData>
        <addressData>
          <addrName>Hans Muster</addrName>
        </addressData>
      </paymentData>
      <transactionData>
        <trxAmount>500</trxAmount>
        <trxCurrency>EUR</trxCurrency>
      </transactionData>
      <options></options>
    </preAuthorize>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Der Bestellbetrag wird mit diesem Aufruf auf der Kreditkarte reserviert und kann später über die **capture**-Funktion vor dem Versenden der Ware abgebucht werden. Natürlich können Sie die Zahlung auch am Ende der Bestellung unter Benutzung der Storage-ID komplett abwickeln.

Im zuletzt gezeigten SOAP-Aufruf wird der Datenstorage durch den Parameter `expire_datastorage` sofort für ungültig erklärt, da dieser bei einer Einzel-Bestellung nicht länger benötigt wird. Wenn Sie die Ablaufzeit bereits vorher recht niedrig gesetzt haben, können Sie hier auch auf den Parameter `expire_datastorage` verzichten.

Durch diese beiden Möglichkeiten wird verhindert, dass die Daten für immer bei ipayment gespeichert werden.

Anwendungsfall „Längeres Speichern der Zahlungsdaten für registrierte Shop-Kunden"

Der zuletzt beschriebene Anwendungsfall erzeugt einen Datenspeicher, der die Zahlungsdaten für den Ablauf eines Bestellvorgangs sicher speichert. Dadurch kann der Bestellablauf so flexibel wie möglich gestaltet werden. Der Datenspeicher kann auch für die längere Speicherung der Zahlungsdaten benutzt werden.

Wenn es sich beim Käufer zum Beispiel um einen registrierten Kunden handelt, für den die Zahlungsdaten im Online-Shop über einen längeren Zeitraum hinweg gespeichert werden sollen, kann im Parameter `datastorage_reference` die Kundennummer angegeben werden. In diesem Fall sollten Sie kein explizites Ablaufdatum (`datastorage_expirydate`) setzen.

Wenn zusätzlich noch der Parameter `datastorage_reuse_method` mit dem Wert 32 in Zusammenhang mit einer Storage-Referenz-Nummer gesetzt ist, wird bei ipayment für diesen Kunden eine ID benutzt, auch wenn später die Zahlungsdaten geändert werden. Ebenso wird in diesem Anwendungsfall der Datenspeicher nicht mit der Zahlung als ungültig erklärt.

Anwendungsfall „Abonnement-Zahlungen mit Zeitabständen über 12 Monaten"

Sie können den Datenspeicher auch für regelmäßige Zahlungen nutzen, deren Abonnement-Frequenzen 12 Monate überschreiten.

Sie müssen dazu bei der Anmeldung des Kunden im **Silent-Modus** die zusätzlichen Parameter für die regelmäßigen Zahlungen und die des Datenspeichers kombinieren. Die Abwicklung der Folgezahlungen funktioniert dann genauso, wie es im Anwendungsfall zu den regelmäßigen Zahlungen beschrieben ist.

Weitere Funktionen des ipayment-Systems

In diesem Kapitel werden noch einige Beispiele der weiteren Möglichkeiten des ipayment-Systems anhand des SOAP-Webservices erläutert.

Adressprüfung mit dem SOAP-Webservice

Bei einer Zahlung kann die Anschrift des Kunden auf postalische Korrektheit geprüft werden, wenn Sie die Option in der ipayment-Anwendung aktiviert haben. Über den Webservice können Sie durch die Funktion `checkAddress` außerdem reine Adressprüfungen vornehmen, ohne dass eine Zahlung durchgeführt wird.

Beispiel für die Adressprüfung ohne Zahlung:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <checkAddress>
      <accountData>
        <accountId>99999</accountId>
        <trxuserId>99999</trxuserId>
        <trxpassword>0</trxpassword>
        <adminactionpassword>
        5cfgRT34xsdedtFLdfHxj7tfwx24fe</adminactionpassword>
      </accountData>
      <addressData>
```

```
        <addrStreet>Ernst-Frey-Str. 9</addrStreet>
        <addrCity>Karlsruhe</addrCity>
        <addrZip>76135</addrZip>
        <addrCountry>DE</addrCountry>
      </addressData>
      <maxSuggestions>5</maxSuggestions>
      <requestId>4711</requestId>
    </checkAddress>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Neben den Elementen `accountData` und `addressData` müssen Sie noch zwei weitere Parameter setzen. Das Element `maxSuggestions` beschränkt die Liste der Alternativvorschläge, wenn eine Adresse nicht eindeutig ist und es zu einem Adressteil mögliche Korrekturvorschläge gibt. Im Element `requestId` müssen Sie eine eindeutige ID für den aktuellen Befehlsablauf übergeben.

Als Antwort erhalten Sie bei einer eindeutigen Adresse folgende XML-Nachricht:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <ns1:checkAddressResponse
      xmlns:ns1="https://ipayment.de/service_v3/binding">
      <addresscheckReturn>
        <status>CORRECTED</status>
        <addrStreet>
          <origValue>Ernst-Frey-Str. 9</origValue>
          <status>CORRECTED</status>
          <suggestionList>
            <suggestion>Ernst-Frey-Str.</suggestion>
          </suggestionList>
          <statusDetail>OKAY</statusDetail>
        </addrStreet>
        <addrStreetNumber>
          <origValue></origValue>
          <status>CORRECTED</status>
          <suggestionList>
            <suggestion>9</suggestion>
          </suggestionList>
          <statusDetail>OKAY</statusDetail>
        </addrStreetNumber>
        <addrCity>
          <origValue>Karlsruhe</origValue>
          <status>OK</status>
        </addrCity>
        <addrZip>
          <origValue>76135</origValue>
          <status>OK</status>
        </addrZip>
      </addresscheckReturn>
    </ns1:checkAddressResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Hier wurde nur der Straßenname normalisiert und die Hausnummer in ein eigenes Feld geschrieben. Der Status der betroffenen Felder lautet `CORRECTED`. Bei den unveränderten Feldern wird der Status `OK` zurückgegeben.

Das nachfolgende Beispiel zeigt das Ergebnis einer Anfrage, bei der die eingegebene Adresse so nicht existiert und die Adresse nicht eindeutig korrigiert werden konnte:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <ns1:checkAddressResponse
xmlns:ns1="https://ipayment.de/service_v3/binding">
    <addresscheckReturn>
     <status>SUGGESTIONS</status>
      <addrStreet>
       <origValue>Ernst-Key-Str.</origValue>
       <status>SUGGESTIONS</status>
       <suggestionList>
        <suggestion>Ebertstr.</suggestion>
        <suggestion>Edgar-von-Gierke-Str.</suggestion>
        <suggestion>Ehrmannstr.</suggestion>
        <suggestion>Eisenlohrstr.</suggestion>
        <suggestion>Ernst-Frey-Str.</suggestion>
       </suggestionList>
       <statusDetail>street/suggestions found</statusDetail>
      </addrStreet>
      <addrStreetNumber>
       <origValue>9</origValue>
       <status>OK</status>
      </addrStreetNumber>
      <addrCity>
       <origValue>Karlsruhe</origValue>
       <status>OK</status>
      </addrCity>
      <addrZip>
       <origValue>76135</origValue>
       <status>OK</status>
      </addrZip>
     </addresscheckReturn>
    </ns1:checkAddressResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Das Feld `addrStreet` hat den Status `SUGGESTIONS`. Es werden 5 Alternativvorschläge angeboten, nämlich Ebertstr., Edgar-von-Gierke-Str., Ehrmannstr., Eisenlohrstr. und Ernst-Frey-Str.

Momentan können nur Adressen aus Deutschland auf ihre postalische Korrektheit geprüft werden. Der Adresscheck kann auch nicht überprüfen, ob Ihr Kunde tatsächlich an der angegebenen Adresse wohnt. Beachten Sie, dass Sie den Adresscheck nicht mit den Testaccounts testen können. Sie benötigen dafür einen eigenen Account mit aktivierter Adressprüfung.

Prüfung von E-Mail-Adressen mit dem SOAP-Webservice

Die Funktion `checkEmail` prüft eine angegebene E-Mail-Adresse auf Syntaxfehler und testet, ob die angegebene Domain E-Mails empfangen kann. Wenn es für die Domain keinen sogenannten „MX Record“ gibt, wird versucht, den Server auf Port 25 zu kontaktieren. Die Prü-

fung des Ports können Sie bei der Anfrage mit der Funktion `checkPort` deaktivieren, um längere Wartezeiten zu vermeiden.

Beispiel für eine Anfrage mit einer existierenden E-Mail-Adresse:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <checkEmail>
      <accountData>
        <accountId>99999</accountId>
        <trxuserId>99999</trxuserId>
        <trxpassword>0</trxpassword>
        <adminactionpassword>
          5cfgRT34xsdedtFLdfHxj7tfwx24fe</adminactionpassword>
      </accountData>
      <email>support@ipayment.de</email>
      <checkPort>false</checkPort>
    </checkEmail>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Antwort des ipayment-Servers:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <ns1:checkEmailResponse
xmlns:ns1="https://ipayment.de/service_v3/binding">
      <emailcheckReturn>
        <status>OK</status>
      </emailcheckReturn>
    </ns1:checkEmailResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Die E-Mail-Prüfung wird auch für E-Mail-Adressen verwendet, die während des Zahlungsvorgangs von Ihrem Kunden angegeben werden.

Platzhalter der Templates

Website und E-Mails

Wenn das ipayment-System im **Normalen Modus** eingebunden wird, erfolgt die Eingabe der Zahlungsdaten direkt auf dem ipayment-Server. Das Formular für die Zahlungsdaten kann individuell per HTML angepasst werden. Editieren Sie dazu einfach im ipayment-Konfigurationsmenü unter **Anwendungen** bei der entsprechenden Anwendung die Templates. Die Templates zu einer bereits angelegten Anwendung finden Sie in der Konfiguration unter dem Link **Template-Texte.**

Das Template für "E-Mail an den Kunden" wird in allen Integrationsmodi verwendet.

Nachfolgend finden Sie alle im ipayment-System verfügbaren Platzhalter, die in den Templates durch den entsprechenden Wert ersetzt werden. Wenn Sie eigene Parameter an das ipayment-System übergeben, sind diese ebenso über ihren Namen im Template verfügbar. Ein Platzhalter muss im Template in der Form `{--variable_name--}` angegeben werden, damit er ersetzt wird.

Platzhalter, die in allen Templates verfügbar sind

Diese Platzhalter sind in allen Templates verfügbar.

Zusätzlich zu diesen Platzhaltern sind die Parameter verfügbar, die das Shopsystem zusätzlich übergeben hat.

Platzhalter	Beschreibung
`transaction_amount`	Betrag der Transaktion (als Ganzzahl in der kleinsten Währungseinheit, zum Beispiel 1234 Cent statt 12,34 Euro)
`transaction_amount_formated`	Formatierter Betrag mit Währungskürzel (zum Beispiel 12,34 EUR)
`transaction_amount_formated_us`	Formatierter Betrag mit Währungskürzel im US-Format (zum Beispiel EUR 12.34)
`merchant_name`	Firmenname des Händlers
`merchant_email`	E-Mail-Adresse des Händlers
`customer_name`	Name des Karteninhabers oder Kontoinhabers
`custom_values`	Dieser Platzhalter muss unbedingt innerhalb des Formulars in den Templates vorhanden sein.
`addr_street, addr_zip, addr_city, addr_email, ...`	Adressdaten des Kunden.

Platzhalter „custom_values" ist notwendig

Dieser Platzhalter darf nicht gelöscht werden, da die darin übermittelten Daten vom ipayment-System benötigt werden.

Platzhalter in Templates zur Eingabe der Zahlungsdaten

Diese Platzhalter sind im Formular zur Zahlungsdateneingabe im **Normalen Modus** verfügbar.

Platzhalter	Beschreibung
backlink	Link zurück in Ihre Anwendung. Diesen Link können Sie als Parameter übergeben. Ansonsten versucht das System, den Link automatisch zu ermitteln.
selection_cc_expdate_month	Für Kreditkartenzahlung: Auswahlliste der Monate (2-stellig) für das Ablaufdatum. Zur Vorauswahl können Sie den Parameter cc_expdate_month übergeben.
selection_cc_expdate_month_options	Für Kreditkartenzahlung: Liste der Einträge im <option> Tag für die Auswahlliste der Monate beim Ablaufdatum. Zur Vorauswahl können Sie den Parameter cc_expdate_month übergeben. Wenn dieser Platzhalter verwendet wird, darf selection_cc_expdate_month nicht angegeben sein. Das Tag <select name="cc_expdate_month"> müssen Sie in diesem Fall selbst setzen, können so aber JavaScript oder CSS nutzen.
selection_cc_expdate_year	Für Kreditkartenzahlung: Auswahlliste der Jahre (4-stellig) für das Ablaufdatum, zur Vorauswahl können Sie den Parameter cc_expdate_year übergeben.
selection_cc_expdate_year_options	Für Kreditkartenzahlung: Liste der Einträge im <option> Tag für die Auswahlliste der Jahre beim Ablaufdatum. Zur Vorauswahl können Sie den Parameter cc_expdate_year übergeben. Wenn dieser Platzhalter verwendet wird, darf selection_cc_expdate_year nicht angegeben sein. Das Tag <select name="cc_expdate_year"> müssen Sie in diesem Fall selbst setzen, können so aber JavaScript oder CSS nutzen.
selection_cc_startdate_month	Für Kreditkartenzahlung (nur Solo-Karten): Auswahlliste der Monate (2-stellig) für Ausgabedatum. Zur Vorauswahl können Sie den Parameter cc_startdate_month übergeben.
selection_cc_startdate_month_options	Für Kreditkartenzahlung: Liste der Einträge im <option> Tag für die Auswahlliste der Monate beim Ausgabedatum. Zur Vorauswahl können Sie den Parameter cc_startdate_month übergeben. Wenn dieser Platzhalter verwendet wird, darf selection_cc_startdate_month nicht angegeben sein. Das Tag <select name="cc_startdate_month"> müssen Sie in die-

Platzhalter	Beschreibung
	sem Fall selbst setzen, können so aber JavaScript oder CSS nutzen.
selection_cc_startdate_year	Für Kreditkartenzahlung (nur Solo-Karten): Auswahlliste der Jahre (4-stellig vom aktuellen Jahr bis 20 Jahre zurück) für das Ausgabedatum. Zur Vorauswahl können Sie den Parameter cc_startdate_year übergeben.
selection_cc_startdate_year_options	Für Kreditkartenzahlung: Liste der Einträge im <option> Tag für die Auswahlliste der Jahre beim Ablaufdatum. Zur Vorauswahl können Sie den Parameter cc_expdate_year übergeben. Wenn dieser Platzhalter verwendet wird, darf selection_cc_expdate_year nicht angegeben sein. Das Tag <select name="cc_expdate_year"> müssen Sie in diesem Fall selbst setzen, können so aber JavaScript oder CSS nutzen.
selection_addr_country_elv	Nur für ELV-Zahlungen: ELV ist zurzeit in folgenden Ländern möglich: Deutschland (DE) und Österreich (AT). Diese Länder werden mit vollem Ländernamen ausgegeben. Je nach Sprache der Fehlermeldung (können Sie im Parameter error_lang definieren) wird die Liste auf Deutsch oder Englisch ausgegeben. Zur Vorauswahl können Sie den Parameter addr_country übergeben.
errormessage	Fehlertext, der erscheint, wenn die Transaktion abgelehnt wurde und die Eingabeseite zum zweiten Mal angezeigt wird.

Platzhalter für Bestellbestätigung und Ergebnis-Seite

Diese Platzhalter sind für die Bestellbestätigungs-E-Mail und die Ergebnis-Seite im **Normalen Modus** verfügbar.

Platzhalter	Beschreibung
transaction_date	Datum der Transaktion
transaction_amount_formated_simple	Formatierte Gleitkommazahl des Betrags
transaction_number	ipayment-Transaktionsnummer
transaction_authcode	Autorisierungsnummer, die vom Zahlungsanbieter zurückgegeben wurde.
transaction_paymentmethod	Zahlungsmethode, zum Beispiel VisaCard, Mastercard oder ELV.
remote_ip_country	Herkunftsland des Käufers (2-stelliger Ländercode, siehe https://ipayment.de/ **> Technik**)
paymentdata_country	Herkunftsland der Zahlungsdaten des Käufers (zwei-

Platzhalter	Beschreibung
	stelliger Ländercode).
issuer_avs_response	Ergebnis der AVS-Prüfung des Zahlungsanbieters. Es sind die gleichen Werte möglich wie beim Parameter trx_issuer_avs_response.
payauth_status	Status der 3-D-Secure-Prüfung. Es sind die gleichen Werte möglich wie beim Parameter trx_payauth_status.
trx_user_comment	Interner Kommentar zur Zahlung
shopper_id	Externe IDs, die Sie beim ipayment-Aufruf übergeben können.

Platzhalter nur im Template "Webseite-Ergebnis"

Diese Platzhalter sind für Ergebnis-Seiten im **Normalen Modus** verfügbar. Die Inhalte der jeweiligen Platzhalter hängen davon ab, welche Werte in den Parametern `redirect_url` und `redirect_action` angegeben wurden.

Platzhalter	Beschreibung
redirect_form	Durch diesen Platzhalter wird das Ergebnis-Formular um das Tag `<form>` erweitert, wenn im Parameter `redirect_action` der Wert `POST` steht. Wenn in `redirect_url` eine URL angegeben wurde, wird diese im Attribut `action` des `<form>`-Tags eingetragen.
continue_link	Je nach der in `redirect_action` angegebenen Methode (`GET` oder `POST`) wird die Ergebnis-URL durch diese Variable als Schaltfläche (`POST`) oder Link (`GET`) ausgegeben. Die Sprache der Beschriftung ist von den Angaben beim Parameter `error_lang` abhängig. Ein Klick auf die Schaltfläche oder den Link führt auf die Seite, die in `redirect_url` angegeben wurde.

Verwendung von eigenen Bildern

Die ipayment-Seiten werden über eine SSL-gesicherte Verbindung aufgerufen. Dadurch ist es nicht ohne weiteres möglich, Bilder von anderen Servern über ungesicherte HTTP-Verbindungen einzubinden. Diese Bilder werden von den Browsern meistens als unsicher erkannt und deshalb abgelehnt. Um diese Bilder dennoch anzeigen zu lassen, können Sie in den ipayment-Templates ein spezielles Format verwenden.

Ein Beispiel für die Einbindung eines Bildes, das auf einem anderen Server liegt:

```
<img src="{--img="http://www.meine-domain.de/bild.jpg"--}" border=0>
```

Ersetzen Sie hierbei die URL hinter `img=` durch die tatsächliche URL zum Bild auf Ihrem Server. Das Tag `{--img=" "--}` ändert die URL des Bildes so ab, dass es auch bei Verwendung von SSL von Ihrem Server geladen und ohne Warnmeldung angezeigt wird. Wenn Ihr Server ebenfalls ein gültiges SSL-Zertifikat besitzt, benötigen Sie diesen speziellen Platzhalter nicht. Binden Sie die Bild-URL in diesem Fall einfach direkt über HTTPS ein.

Index der Parameter-Namen

CGI-Namen

Webservice-Namen

Anhang

Technische Dokumentation des SOAP-Webservices

In diesem Kapitel wird der SOAP-Webservice technisch dokumentiert. Die Informationen wurden automatisch generiert und beschreiben die angebotenen Funktionen, Nachrichten, Parameter und Datentypen.

Informationen zu den Funktionen und zum Aufbau

services	bindings	porttypes	messages
IpaymentWebserviceV3.0	**ipaymentBinding**	**ipaymentPortType**	**addresscheckResponse**
			authorizeRequest
			basecheckRequest
			captureRequest
			checkAddressRequest
			checkEmailRequest
			checksaveRequest
			createSessionRequest
			createSessionResponse
			emailcheckResponse
			generalRefundRequest
			ipaymentResponse
			paymentAuthenticationReturnRequest
			preAuthorizeRequest
			reAuthorizeRequest
			refundRequest
			rePreAuthorizeRequest
			reverseRequest
			voiceAuthorizeCCRequest
			voiceGeneralRefundCCRequest

service **IpaymentWebserviceV3.0**

ports **ipayment**

binding **tns:ipaymentBinding**

binding **ipaymentBinding**

type **tns:ipaymentPortType**

operations **authorize**

input `<soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>`

output `<soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>`

checksave

input `<soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>`

output `<soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>`

generalRefund
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

basecheck
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

capture
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

preAuthorize
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

reAuthorize
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

rePreAuthorize
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

reverse
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

refund
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

paymentAuthenticationReturn
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

checkAddress
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

checkEmail
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

voiceAuthorizeCC
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

voiceGeneralRefundCC
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

createSession
input <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>
output <soap:body use="literal" namespace="https://ipayment.de/service_v3/binding"/>

used by Service **IpaymentWebserviceV3.0** in Port **ipayment**

porttype **ipaymentPortType**

operations

authorize
- input **tns:authorizeRequest**
- output **tns:ipaymentResponse**

checksave
- input **tns:checksaveRequest**
- output **tns:ipaymentResponse**

generalRefund
- input **tns:generalRefundRequest**
- output **tns:ipaymentResponse**

basecheck
- input **tns:basecheckRequest**
- output **tns:ipaymentResponse**

capture
- input **tns:captureRequest**
- output **tns:ipaymentResponse**

preAuthorize
- input **tns:preAuthorizeRequest**
- output **tns:ipaymentResponse**

reAuthorize
- input **tns:reAuthorizeRequest**
- output **tns:ipaymentResponse**

rePreAuthorize
- input **tns:rePreAuthorizeRequest**
- output **tns:ipaymentResponse**

reverse
- input **tns:reverseRequest**
- output **tns:ipaymentResponse**

refund
- input **tns:refundRequest**
- output **tns:ipaymentResponse**

paymentAuthenticationReturn
- input **tns:paymentAuthenticationReturnRequest**
- output **tns:ipaymentResponse**

checkAddress
- input **tns:checkAddressRequest**
- output **tns:addresscheckResponse**

checkEmail
- input **tns:checkEmailRequest**
- output **tns:emailcheckResponse**

voiceAuthorizeCC
- input **tns:voiceAuthorizeCCRequest**
- output **tns:ipaymentResponse**

voiceGeneralRefundCC

input **tns:voiceGeneralRefundCCRequest**
output **tns:ipaymentResponse**

createSession
input **tns:createSessionRequest**
output **tns:createSessionResponse**
used by **binding ipaymentBinding**

message **basecheckRequest**
parts **accountData**
type **tns:AccountData**

paymentData
type **tns:PaymentData**

transactionData
type **tns:TransactionData**
used by **PortType ipaymentPortType in Operation basecheck**

message **checksaveRequest**
parts **accountData**
type **tns:AccountData**

paymentData
type **tns:PaymentData**

transactionData
type **tns:TransactionData**

options
type **tns:OptionData**
used by **PortType ipaymentPortType in Operation checksave**

message **preAuthorizeRequest**
parts **accountData**
type **tns:AccountData**

paymentData
type **tns:PaymentData**

transactionData
type **tns:TransactionData**

options
type **tns:OptionData**
used by **PortType ipaymentPortType in Operation preAuthorize**

message **captureRequest**
parts **accountData**
type **tns:AccountData**

origTrxNumber
type **xsd:string**

transactionData
type **tns:TransactionData**

options
type **tns:OptionData**

used by **PortType ipaymentPortType in Operation capture**

message **refundRequest**

parts **accountData**
type **tns:AccountData**

origTrxNumber
type **xsd:string**

transactionData
type **tns:TransactionData**

options
type **tns:OptionData**

used by **PortType ipaymentPortType in Operation refund**

message **reverseRequest**

parts **accountData**
type **tns:AccountData**

origTrxNumber
type **xsd:string**

transactionData
type **tns:TransactionData**

options
type **tns:OptionData**

used by **PortType ipaymentPortType in Operation reverse**

message **authorizeRequest**

parts **accountData**
type **tns:AccountData**

paymentData
type **tns:PaymentData**

transactionData
type **tns:TransactionData**

options
type **tns:OptionData**

used by **PortType ipaymentPortType in Operation authorize**

message **reAuthorizeRequest**

parts
- **accountData**
  - type **tns:AccountData**
- **origTrxNumber**
  - type **xsd:string**
- **transactionData**
  - type **tns:TransactionData**
- **options**
  - type **tns:OptionData**

used by PortType **ipaymentPortType** in Operation **reAuthorize**

message **voiceAuthorizeCCRequest**

parts
- **accountData**
  - type **tns:AccountData**
- **paymentData**
  - type **tns:PaymentData**
- **voiceAuthcode**
  - type **xsd:string**
- **transactionData**
  - type **tns:TransactionData**
- **options**
  - type **tns:OptionData**

used by PortType **ipaymentPortType** in Operation **voiceAuthorizeCC**

message **paymentAuthenticationReturnRequest**

parts
- **threedsecureData**
  - type **tns:ThreeDSecureData**

used by PortType **ipaymentPortType** in Operation **paymentAuthenticationReturn**

message **rePreAuthorizeRequest**

parts
- **accountData**
  - type **tns:AccountData**
- **origTrxNumber**
  - type **xsd:string**
- **transactionData**
  - type **tns:TransactionData**
- **options**
  - type **tns:OptionData**

used by PortType **ipaymentPortType** in Operation **rePreAuthorize**

message **generalRefundRequest**

parts
- **accountData**
  - type **tns:AccountData**
- **paymentData**
  - type **tns:PaymentData**
- **transactionData**
  - type **tns:TransactionData**
- **options**
  - type **tns:OptionData**

used by
- PortType **ipaymentPortType** in Operation **generalRefund**

message **voiceGeneralRefundCCRequest**

parts
- **accountData**
  - type **tns:AccountData**
- **paymentData**
  - type **tns:PaymentData**
- **voiceAuthcode**
  - type **xsd:string**
- **transactionData**
  - type **tns:TransactionData**
- **options**
  - type **tns:OptionData**

used by
- PortType **ipaymentPortType** in Operation **voiceGeneralRefundCC**

message **ipaymentResponse**

parts
- **ipaymentReturn**
  - type **tns:PaymentReturn**

used by
- PortType **ipaymentPortType** in Operation **authorize**
- PortType **ipaymentPortType** in Operation **checksave**
- PortType **ipaymentPortType** in Operation **generalRefund**
- PortType **ipaymentPortType** in Operation **basecheck**
- PortType **ipaymentPortType** in Operation **capture**
- PortType **ipaymentPortType** in Operation **preAuthorize**
- PortType **ipaymentPortType** in Operation **reAuthorize**
- PortType **ipaymentPortType** in Operation **rePreAuthorize**
- PortType **ipaymentPortType** in Operation **reverse**
- PortType **ipaymentPortType** in Operation **refund**
- PortType **ipaymentPortType** in Operation **paymentAuthenticationReturn**
- PortType **ipaymentPortType** in Operation **voiceAuthorizeCC**
- PortType **ipaymentPortType** in Operation **voiceGeneralRefundCC**

message **checkAddressRequest**

parts
- **accountData**
  - type **tns:AccountData**
- **addressData**
  - type **tns:AddresscheckData**

maxSuggestions
type **xsd:int**

requestId
type **xsd:string**
used by **PortType ipaymentPortType in Operation checkAddress**

message **addresscheckResponse**
parts **addresscheckReturn**
type **tns:AddresscheckReturn**
used by **PortType ipaymentPortType in Operation checkAddress**

message **checkEmailRequest**
parts **accountData**
type **tns:AccountData**

email
type **xsd:string**

checkPort
type **xsd:boolean**
used by **PortType ipaymentPortType in Operation checkEmail**

message **emailcheckResponse**
parts **emailcheckReturn**
type **tns:EmailcheckReturn**
used by **PortType ipaymentPortType in Operation checkEmail**

message **createSessionRequest**
parts **accountData**
type **tns:AccountData**

transactionData
type **tns:TransactionData**

transactionType
type **tns:TransactionType**

paymentType
type **tns:PaymentType**

options
type **tns:OptionData**

processorUrls
type **tns:ProcessorUrlData**
used by **PortType ipaymentPortType in Operation createSession**

message **createSessionResponse**

parts **sessionId**

type **xsd:string**

used by **PortType ipaymentPortType in Operation createSession**

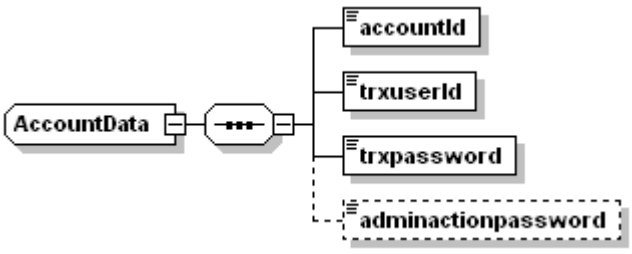
Informationen zu den definierten Datentypen

targetNamespace: **https://ipayment.de/service_v3/extern**

Complex types	Simple types
AccountData	**AddresscheckFieldstatus**
AddresscheckData	**EmailcheckStatus**
AddresscheckFieldresult	**InstallmentTyp**
AddresscheckReturn	**PaymentReturnStatus**
AddressData	**PaymentType**
BrowserData	**RecurringTyp**
CCData	**TransactionType**
ClientData	
ELVData	
EmailcheckReturn	
InstallmentData	
OptionData	
OptionHash	
OptionHashItem	
PaymentData	
PaymentReturn	
PaymentReturnError	
PaymentReturnRedirect	
PaymentReturnSuccess	
ProcessorUrlData	
RecurringData	
StorageData	
SuggestionArray	
ThreeDSecureData	
TransactionData	

complexType **AccountData**

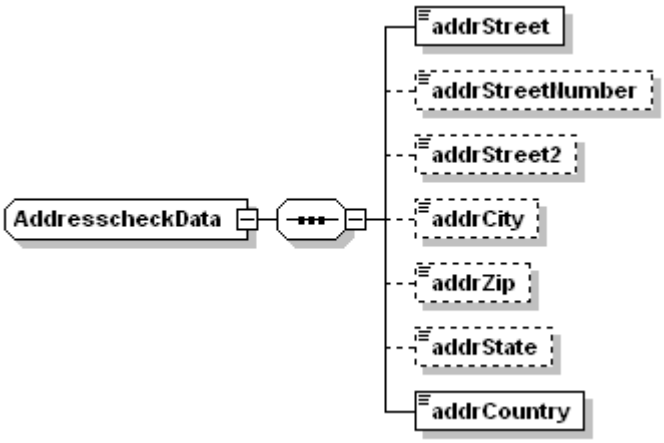
diagram



namespace https://ipayment.de/service_v3/extern

complexType **AddresscheckData**

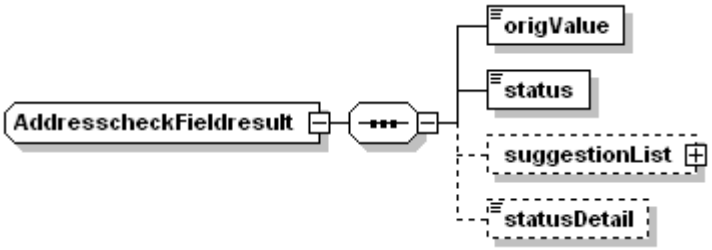
diagram



namespace https://ipayment.de/service_v3/extern

complexType **AddresscheckFieldresult**

diagram

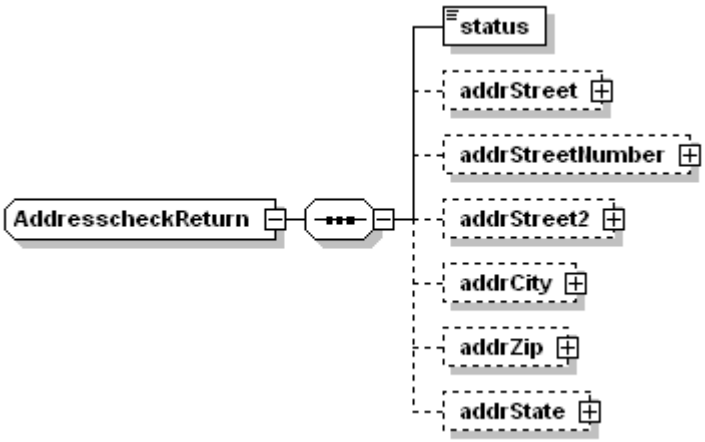


namespace https://ipayment.de/service_v3/extern

used by elements **AddresscheckReturn/addrCity AddresscheckReturn/addrState AddresscheckReturn/addrStreet AddresscheckReturn/addrStreet2 AddresscheckReturn/addrStreetNumber AddresscheckReturn/addrZip**

complexType **AddresscheckReturn**

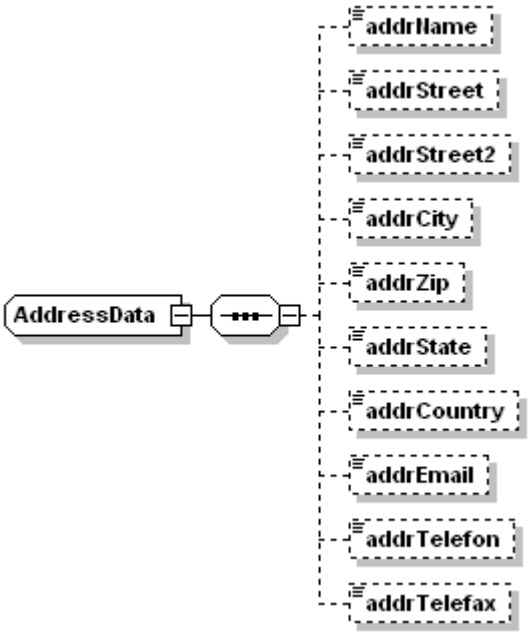
diagram



namespace https://ipayment.de/service_v3/extern

complexType **AddressData**

diagram

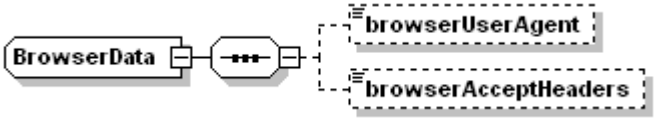


namespace https://ipayment.de/service_v3/extern

used by elements **PaymentData/addressData PaymentReturn/addressData**

complexType **BrowserData**

diagram

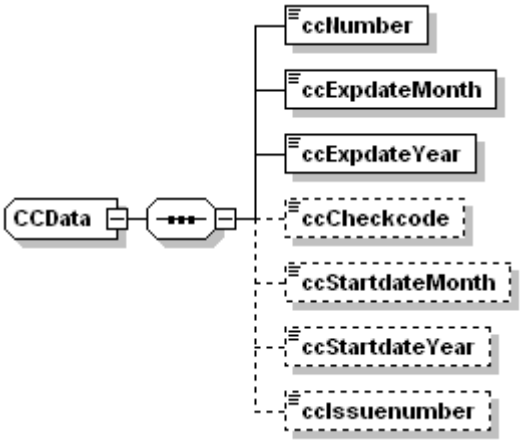


namespace https://ipayment.de/service_v3/extern

used by element **OptionData/browserData**

complexType **CCData**

diagram

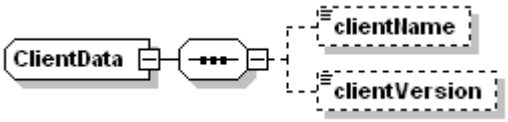


namespace https://ipayment.de/service_v3/extern

used by element **PaymentData/ccData**

complexType **ClientData**

diagram

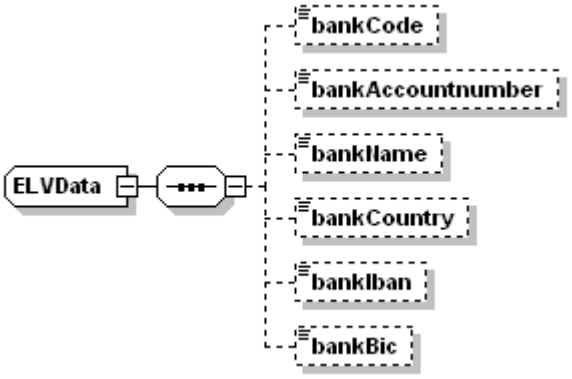


namespace https://ipayment.de/service_v3/extern

used by element **OptionData/clientData**

complexType **ELVData**

diagram

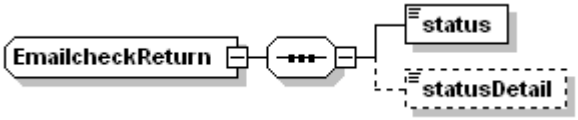


namespace https://ipayment.de/service_v3/extern

used by element **PaymentData/elvData**

complexType **EmailcheckReturn**

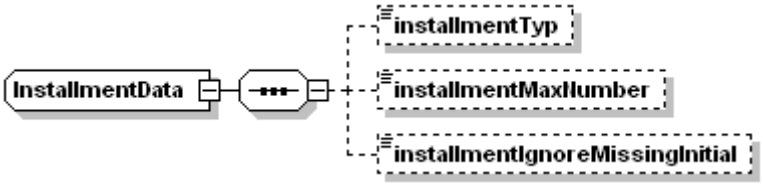
diagram



namespace https://ipayment.de/service_v3/extern

complexType **InstallmentData**

diagram

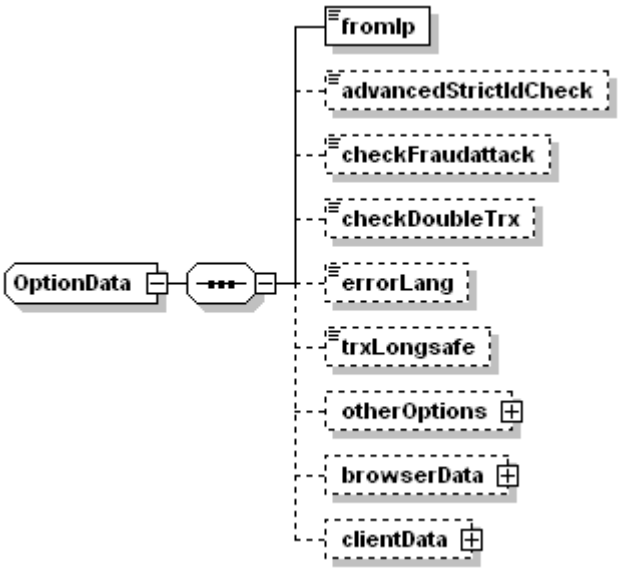


namespace https://ipayment.de/service_v3/extern

used by element **TransactionData/installmentData**

complexType **OptionData**

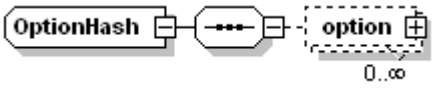
diagram



namespace https://ipayment.de/service_v3/extern

complexType **OptionHash**

diagram

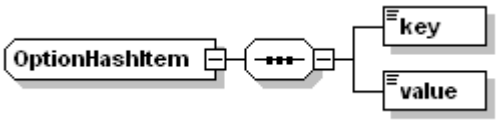


namespace https://ipayment.de/service_v3/extern

used by element **OptionData/otherOptions**

complexType **OptionHashItem**

diagram

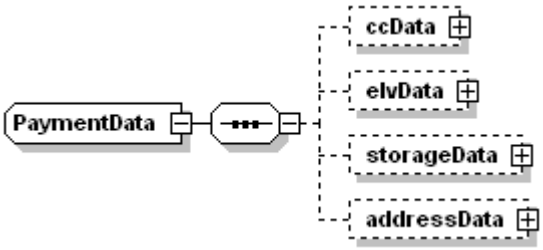


namespace https://ipayment.de/service_v3/extern

used by element **OptionHash/option**

complexType **PaymentData**

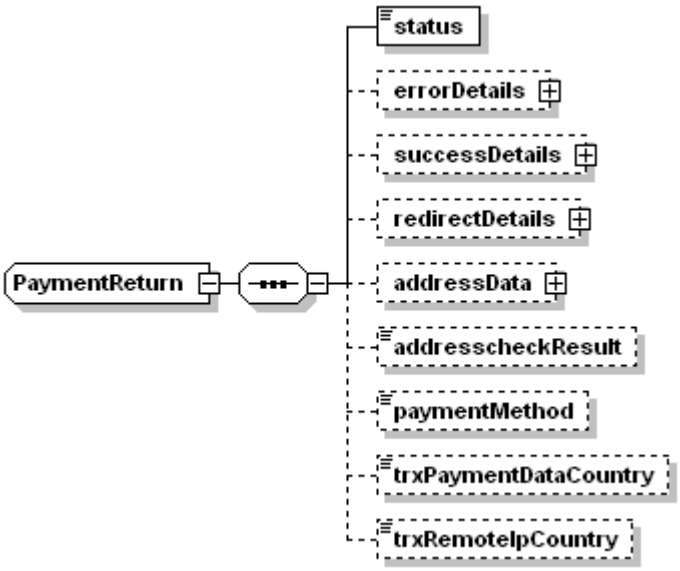
diagram



namespace https://ipayment.de/service_v3/extern

complexType **PaymentReturn**

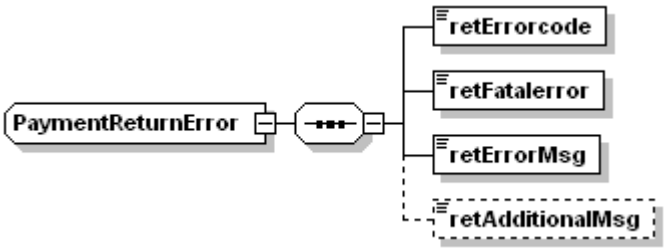
diagram



namespace https://ipayment.de/service_v3/extern

complexType **PaymentReturnError**

diagram

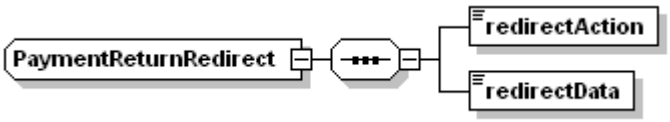


namespace https://ipayment.de/service_v3/extern

used by element **PaymentReturn/errorDetails**

complexType **PaymentReturnRedirect**

diagram

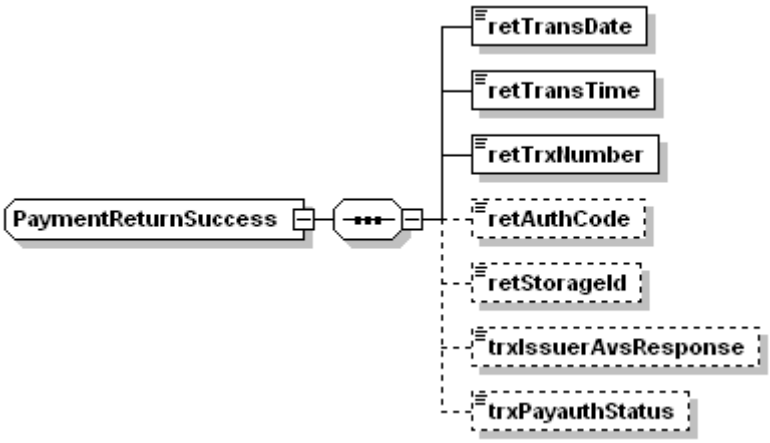


namespace https://ipayment.de/service_v3/extern

used by element **PaymentReturn/redirectDetails**

complexType **PaymentReturnSuccess**

diagram

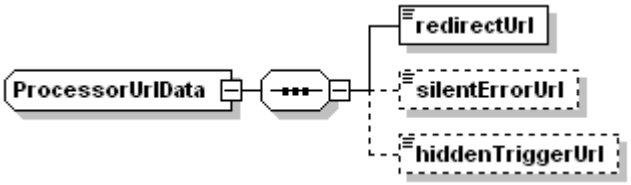


namespace https://ipayment.de/service_v3/extern

used by element **PaymentReturn/successDetails**

complexType **ProcessorUrlData**

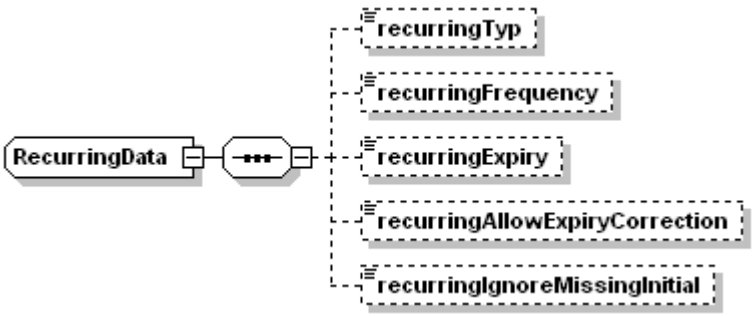
diagram



namespace https://ipayment.de/service_v3/extern

complexType **RecurringData**

diagram



namespace https://ipayment.de/service_v3/extern

used by element **TransactionData/recurringData**

complexType **StorageData**

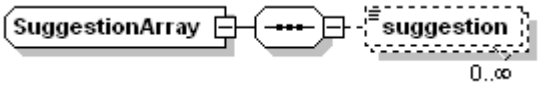
diagram

namespace https://ipayment.de/service_v3/extern

used by element **PaymentData/storageData**

complexType **SuggestionArray**

diagram

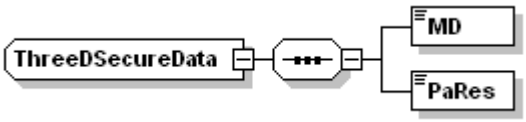


namespace https://ipayment.de/service_v3/extern

used by element **AddresscheckFieldresult/suggestionList**

complexType **ThreeDSecureData**

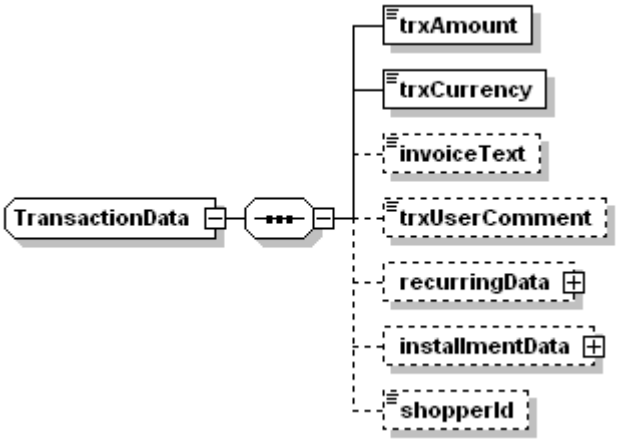
diagram



namespace https://ipayment.de/service_v3/extern

complexType **TransactionData**

diagram



namespace https://ipayment.de/service_v3/extern

simpleType **AddresscheckFieldstatus**

namespace https://ipayment.de/service_v3/extern

type restriction of **xsd:string**

used by elements **AddresscheckFieldresult/status AddresscheckReturn/status**

facets
- enumeration OK
- enumeration CORRECTED
- enumeration SUGGESTIONS
- enumeration ERROR
- enumeration UNCHECKED
- enumeration NORMALIZED

simpleType **EmailcheckStatus**

namespace https://ipayment.de/service_v3/extern

type restriction of **xsd:string**

used by element **EmailcheckReturn/status**

facets
- enumeration OK
- enumeration ERROR

simpleType **InstallmentTyp**

namespace https://ipayment.de/service_v3/extern

type restriction of **xsd:string**

used by element **InstallmentData/installmentTyp**

facets
- enumeration initial
- enumeration sequencial

simpleType **PaymentReturnStatus**

namespace https://ipayment.de/service_v3/extern

type restriction of **xsd:string**

used by element **PaymentReturn/status**

facets
- enumeration SUCCESS
- enumeration ERROR
- enumeration REDIRECT

simpleType **PaymentType**

namespace https://ipayment.de/service_v3/extern

type restriction of **xsd:string**

facets
- enumeration cc
- enumeration elv
- enumeration pp

simpleType **RecurringTyp**

namespace https://ipayment.de/service_v3/extern

type restriction of **xsd:string**

used by element **RecurringData/recurringTyp**

facets
- enumeration initial
- enumeration sequencial

simpleType **TransactionType**

namespace https://ipayment.de/service_v3/extern

type restriction of **xsd:string**

facets
- enumeration auth
- enumeration base_check
- enumeration check_save
- enumeration grefund_cap
- enumeration preauth