Technische Produktbeschreibung

Maileon-Magento-Modul

v0.11.3

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

Dokumentation für das XQueue System

Diese Dokumentation darf ohne vorherige schriftliche Genehmigung durch die XQueue GmbH weder teilweise noch ganz reproduziert, in Datenbanken gespeichert oder in irgendeiner Form übertragen werden. Der Inhalt dieser Dokumentation dient ausschließlich zu Informationszwecken, kann jederzeit geändert werden und stellt keine Verpflichtung seitens der XQueue GmbH dar. Für Fehler der in dieser Dokumentation enthaltenen Informationen wird keine Haftung übernommen.

XQueue GmbH, Christian-Pleß-Str. 11-13, 63069 Offenbach am Main

Inhalt und Ziel	Dieses Dokument beschreibt die genauen Funktionen der Erweiterung von XQueue für Magento
Typ	Benutzerhandbuch
Version	0.11.3
Autor	Marcus Beckerle
Erstellt	22.09.2014
Letzte Änderung	22.02.2019

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

Einleitung

Das Modul synchronisiert Kundendaten, Bestelldaten und Warenkorbabbruchsinformationen zwischen Magento und Maileon. Magento ist hierbei das Primärsystem, das bedeutet, dass die Kundeninformationen in Magento als die Hauptinformationen angesehen werden und Änderungen an diesen Daten mit Magento synchronisiert werden. Anders herum werden nur bestimmte Informationen synchronisiert: die DOI-Bestätigung und Abmelder. Bei Bestelldaten und Warenkorbabbrüchen werden diese Informationen als Transaktionen (Events) an Maileon gesendet und können hier entweder zur Analyse oder zum Auslösen eines Triggermailings verwendet werden. Alle Funktionen sind über ein Konfigurationspanel in Magento einstellbar.

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

Inbetriebname des Moduls

Installation

Das Modul besteht aus zwei Hauptverzeichnissen: `app` und `lib`. Unter `app` befinden sich alle Konfigurationen und der Quellcode der für das Modul geschrieben wurde. Unter `lib` befindet sich eine Kopie des Maileon-PHP-API-Clients der auf der Entwicklerwebsite[1] heruntergeladen werden kann.
Diese Dateien werden einfach in das Hauptverzeichnis von Maileon eingefügt. Hinweis: es werden keine Dateien überschrieben aber da sich das Modul in das Newslettersystem von Magento integriert, wird die Klasse `Mage_Newsletter_Model_Subscriber` im lokalen Modul erweitert. Sollten Sie ein anderes Modul verwenden welches das Newslettersystem ändert, so deaktivieren Sie dieses bitte.
Hinweis: im Maileon-Account müssen ein Custom-Field mit der Bezeichnung `createdByTransaction` vom Typ `boolean`, sowie `storeViewCode` vom Typ `string` angelegt werden. Seit Version 0.11.2 werden diese Kontaktfelder bei Bedarf automatisch angelegt und der Vorgang wird in der Log-Datei , xqueue.plugin.log ' protokolliert.

Konfiguration

Wenn die Installation abgeschlossen ist finden Sie im Magento-Administrationsbereich unter `System` → `Konfiguration` die neue Einstellung `Xqueue` → `Maileon-Settings`, Abbildung 1 zeigt diese Option. Wenn eine Fehlermeldung 404 angezeigt wird loggen Sie sich bitte aus und melden Sie sich erneut am Backend an.

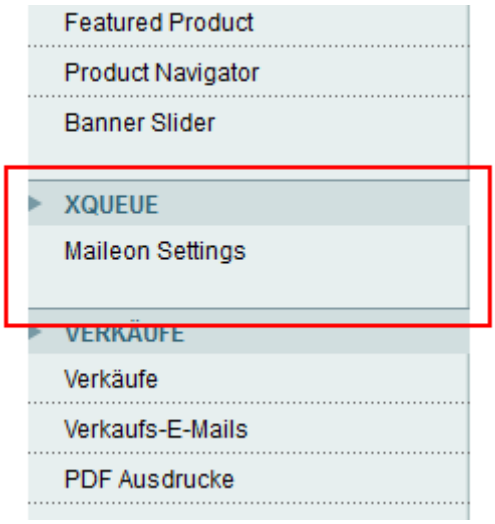

Abbildung 1: Einstellungen für Maileon

[1] http://dev.maileon.com/maileon-php-api-client

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

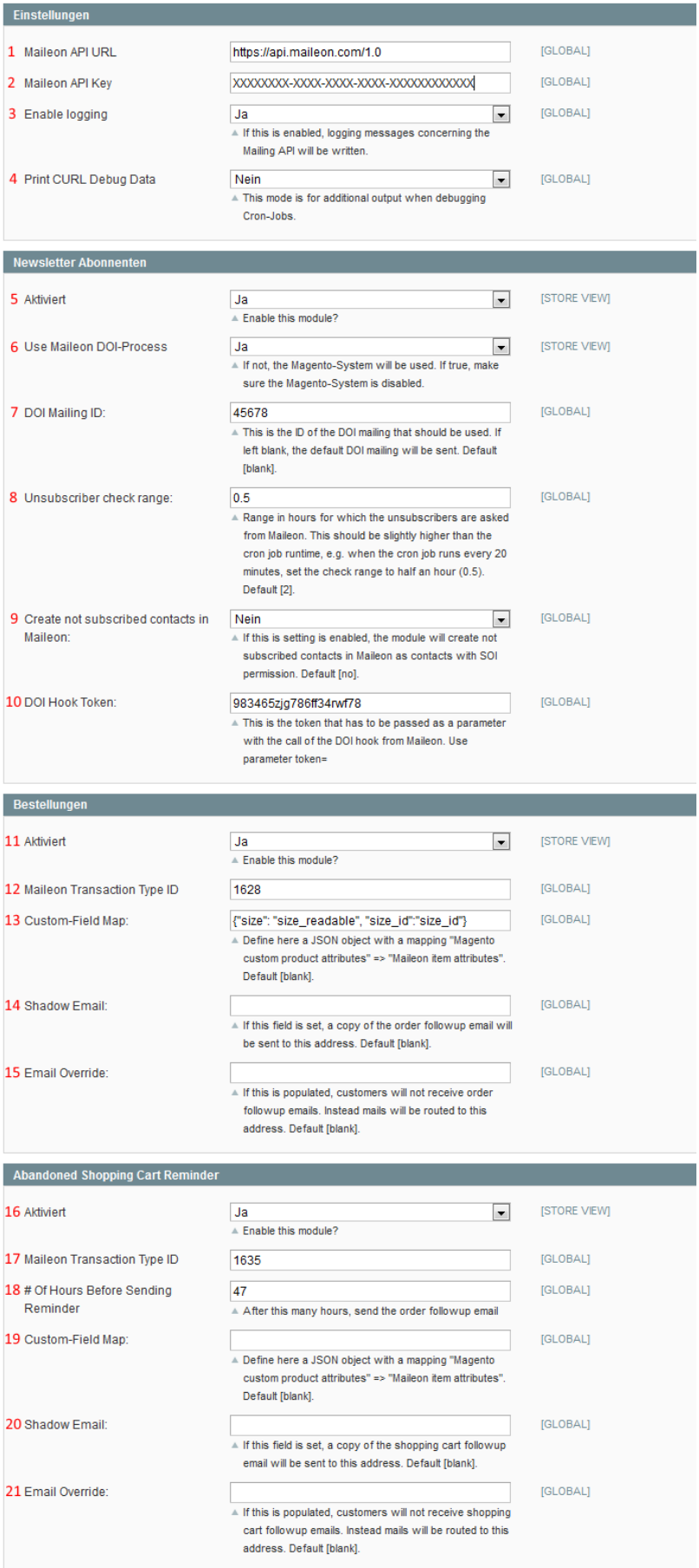

Abbildung 2: Einstellungsmöglichkeiten des Moduls

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

Abbildung 2 zeigt die Optionen die nun gewählt werden können:

1. Die URL zur Maileon REST-API.
2. Der Kundenspezifische API-Key[2].
3. Logging-Informationen aktivieren oder deaktivieren. Hier werden derzeit alle Vorgänge mit Maileon geloggt.
4. Wenn diese Option aktiviert ist werden alle CURL-Anfragen auf der Website ausgegeben. So können Fehler bei der Kommunikation analysiert werden.
5. Aktiviert die Synchronisierung von Newsletter-Abonnenten
6. Wenn diese Option aktiviert ist wird ein DOI-Mailing über Maileon versendet, ansonsten wird das Magento-eigene DOI-System verwendet.
7. Falls im Maileon-Newsletter-Account mehr als ein DOI-Mailing existiert kann hier der Schlüssel des gewünschten Mailings angegeben werden.
8. In der Modulkonfiguration (config.xml) wird ein Cron-Job definiert der die Abmelder von Maileon abfragt. Diese Option legt fest für welchen Zeitraum dies geschieht und sie sollte etwas höher als die Cron-Job-Taktung sein, damit es Überschneidungen gibt. Läuft der Cron-Job z.B. alle 20 oder 25 Minuten sollten die Abmelder der letzten 30 Minuten abgefragt werden. Doppelte Abfragen werden korrekt verarbeitet.
9. Genügt die Anmeldung im Shop als Kriterium, so kann der Kunde mit dieser Option als SOI-Kontakt in Maileon angelegt werden. Er bekommt in diesem Fall nur eine DOI-Bestätigungsmail, wenn er sich noch einmal explizit zum Newsletter anmeldet, kann aber mit der Berechtigung SOI bereits aus Maileon heraus beschickt werden.
10. Klicks von Kontakten auf den DOI-Link in der Bestätigungsmail können aus Maileon heraus an einen Webhook gesendet werden um die Berechtigung zu kommunizieren und zu setzen. In diesem Feld wird der Parameter gesetzt der in Maileon als geheimes Token übergeben wird.
11. Aktiviert die Weiterleitung von Bestellevents an Maileon.
12. Die ID des Transaction-Typs für Bestellungen in Maileon.
13. Für Bestellungen: Ein assoziatives JSON-Array als Schlüssel Magento Custom Attributes von Produkten enthält und als Werte die Bezeichnung im Produkt/Item der Maileon-Transaktion.
    Beispiel: `{"size": "size_readable", "size_id":"size_id"}`
    Diese Anweisung würde das Magento-Produkt-Attribut `size` den Produkten/Items unter der Bezeichnung `size_readable` und das Attribut `size_id` als `size_id` der Transaktion hinzufügen.
14. Wird hier eine Emailadresse gesetzt, so wird bei jeder Bestellung eine zweite Transaktion in Maileon angelegt welche ein Triggermailing an diese Emailadresse auslöst.
    Hinweis: die Option dient zu Testzwecken und wenn in der Bestellbestätigung Kontaktinformationen aus Maileon verwendet werden, dann werden diese aus dem Kontaktprofil übernommen welches zu dieser Email gehört.
15. Wird hier eine Emailadresse gesetzt, so werden die Events mit dieser Emailadresse in Maileon eingetragen. Triggermailings die dann versendet werden, werden ebenfalls (nur) an diese Emailadresse gesendet. Die Option dient zu Testzwecken.
16. Aktiviert die Weiterleitung von Warenkorbabbrechern an Maileon.
17. Die ID des Transaction-Typs für Warenkorbabbrecher in Maileon.
18. Diese Option gibt an nach wie vielen Stunden ein nicht bestellter Warenkorb als verwaist gilt und ein Warenkorbabbrecher-Event gesendet wird.

[2] http://dev.maileon.com/api/rest-api-1-0/securitymechanism

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

19. Für Warenkorbabbrüche: siehe Punkt 13.
20. Wird hier eine Emailadresse gesetzt, so wird bei jeder Warenkorbabbruchs-Transaktion zweite Transaktion in Maileon angelegt welche ein Triggermailing an diese Emailadresse auslöst.
    Hinweis: die Option dient zu Testzwecken und wenn in der Bestellbestätigung Kontaktinformationen aus Maileon verwendet werden, dann werden diese aus dem Kontaktprofil übernommen welches zu dieser Email gehört.
21. Wird hier eine Emailadresse gesetzt, so werden die Events mit dieser Emailadresse in Maileon eingetragen. Triggermailings die dann versendet werden, werden ebenfalls (nur) an diese Emailadresse gesendet. Die Option dient zu Testzwecken.

Wichtig: Das Modul beruht auf den Mechaniken des von Magento selbst bereit gestellten Newslettersystems. Daher greifen auch die Einstellungen des Standardsystemes und es ist notwendig Magento mitzuteilen, dass bei neuen Newsletter-Anmeldern eine Bestätigung (DOI) benötigt wird. Dazu in der Konfiguration den Menüpunkt „`Newsletter`“ auswählen und die Bestätigung auf „`Ja`“ setzen, Abbildung 3 zeigt die Konfiguration.

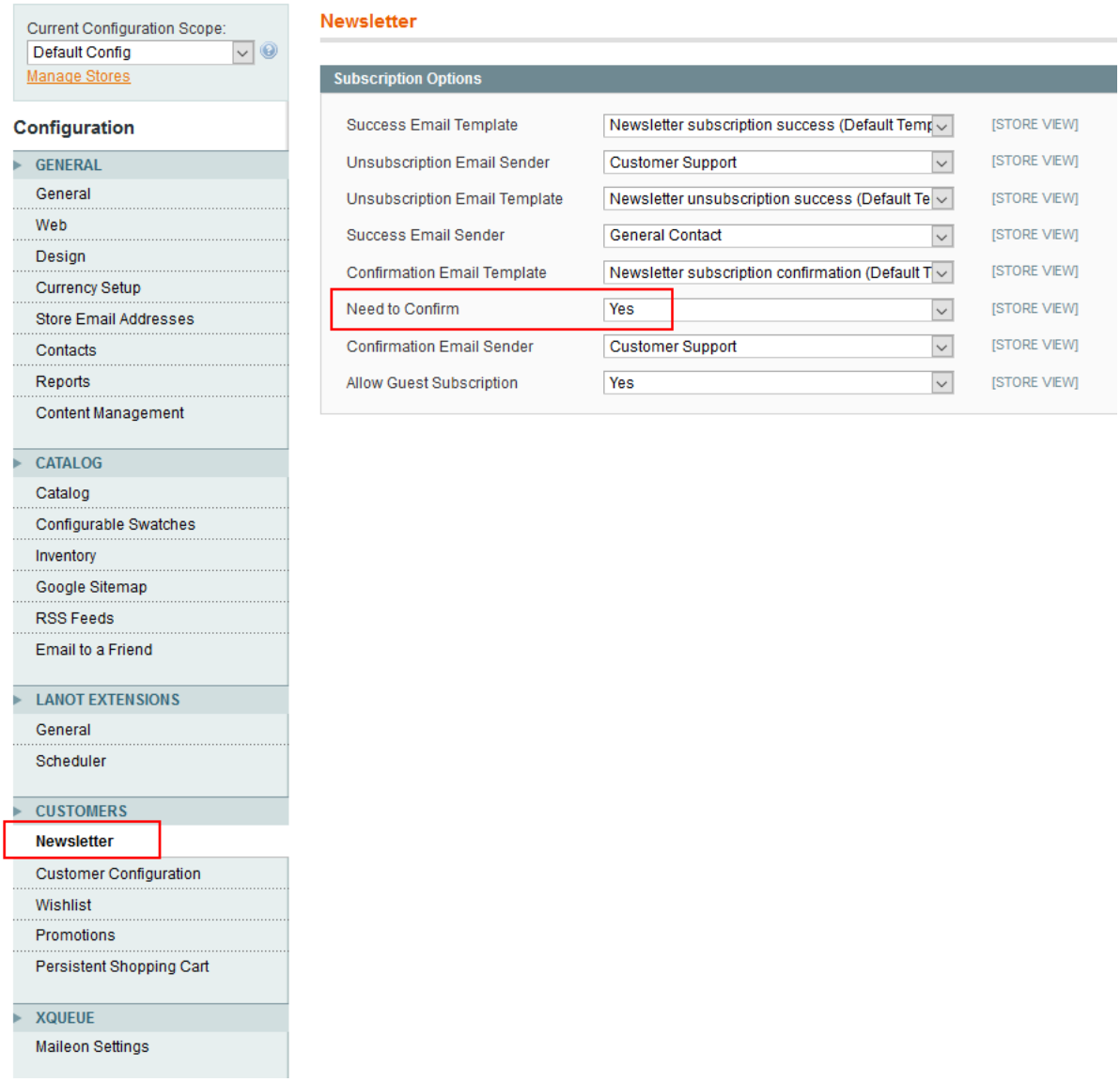

Abbildung 3: Aktivierung der DOI Bestätigung

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

Verschiedene Stores an Maileon anbinden

Es ist möglich verschiedene Storeviews an verschiedene Newsletter-Accounts anzubinden. Abbildung 4 zeigt, dass z.B. der API-Key und somit der entsprechende Maileonaccount auf Storeview-Ebene überschrieben werden kann. Weitere Einstellungen sind auf dem Screenshot ebenfalls markiert, so kann man z.B. auch die Newsletterabbonentenverwaltung über Maileon in bestimmten Stores komplett deaktivieren.

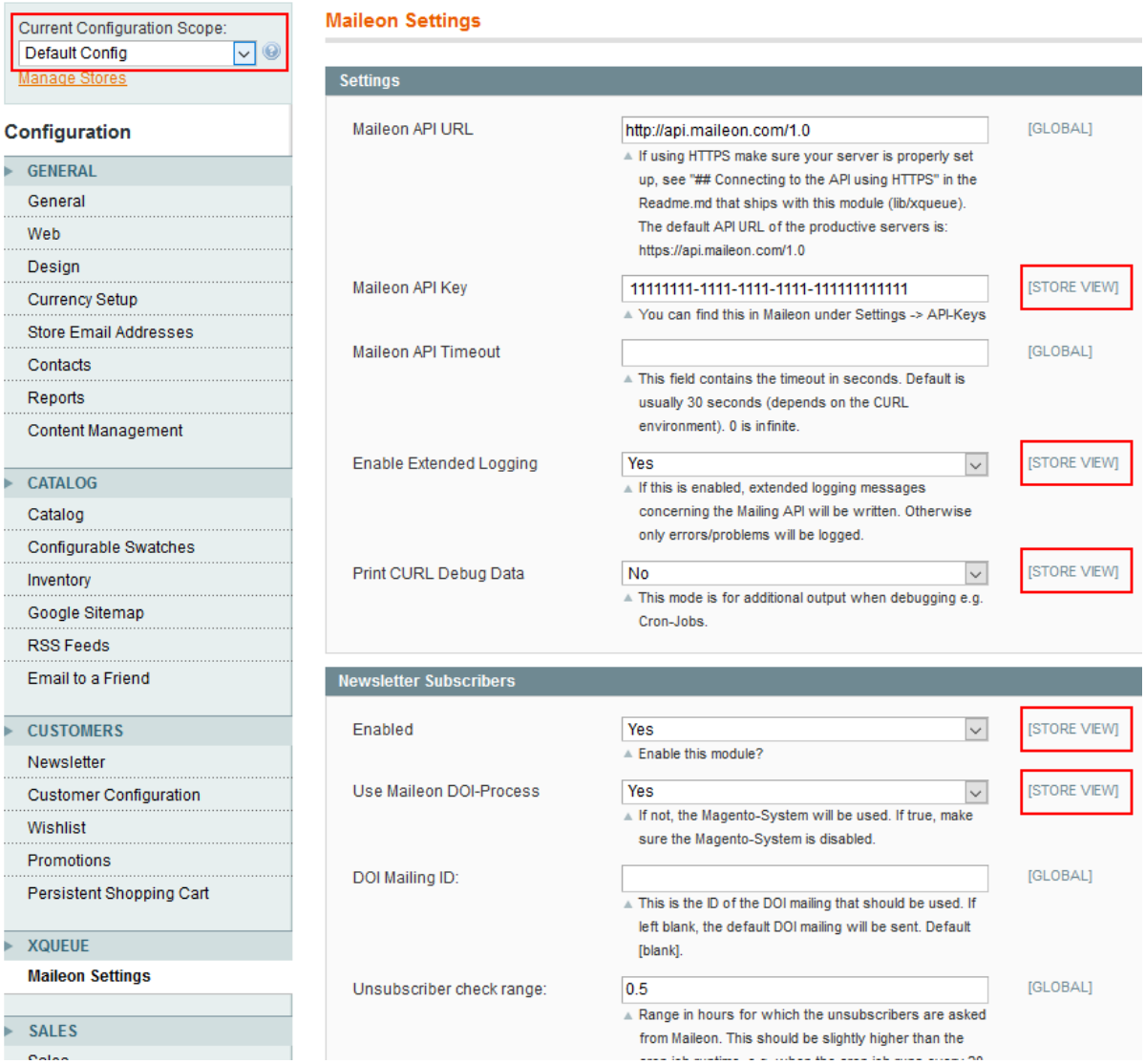

Abbildung 4: Konfigurationseinträge die auf Store-Ebene angepasst werden können

Zur Anpassung der Storeview-Konfiguration muss oben links unter „Current Configuration Scope“ die entsprechende Storeview ausgewählt werden. In Abbildung 5 wurde als Beispiel ein englischer Store als 2. Store mit einer eigenen Storeview eingefügt, die Konfiguration hierfür ausgewählt und der Haken für „Defaultwert“ entfernt. Somit kann für diese Storeview ein anderer API-Key hinterlegt werden und es würden alle Newsletteranmelder in einem eigenen Newsletteraccount verwaltet.

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

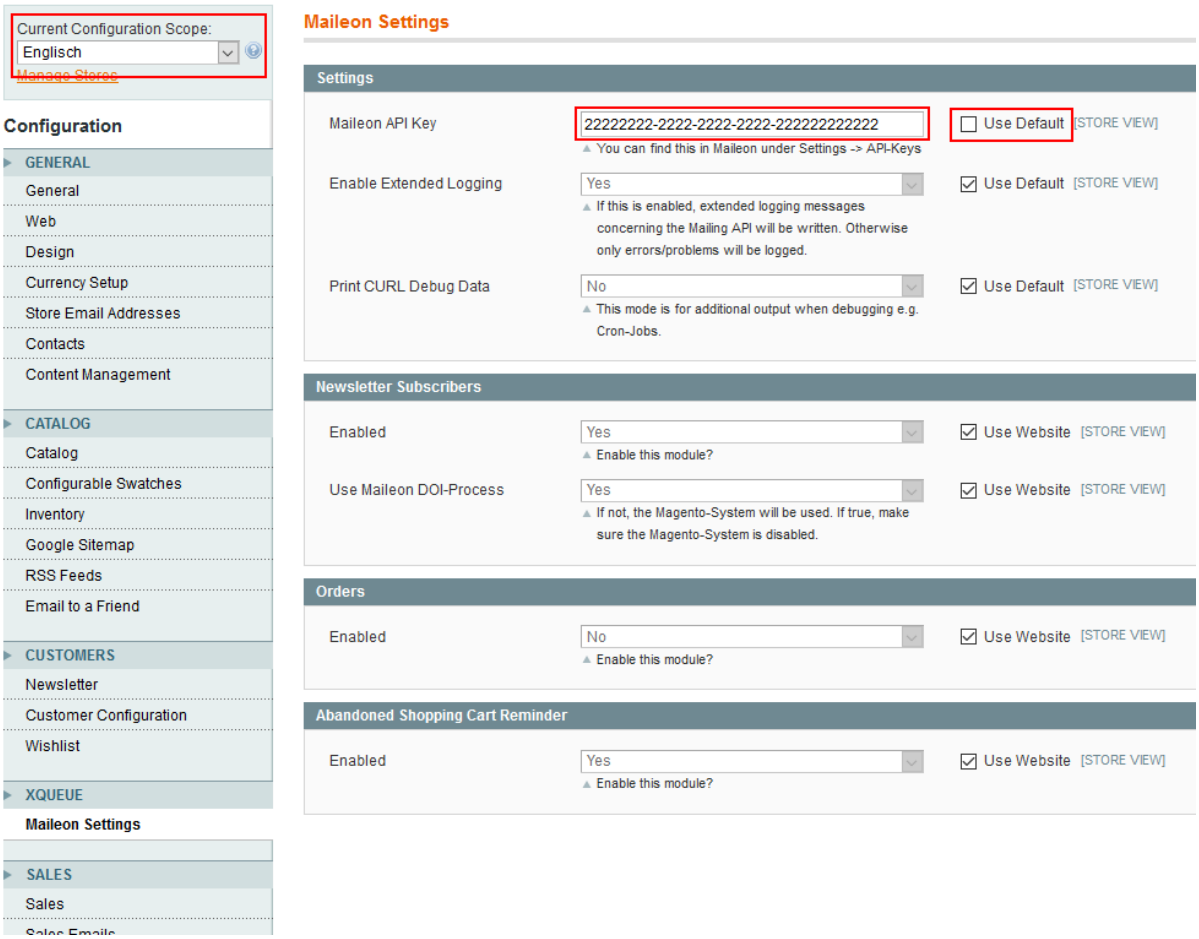

Abbildung 5: Konfiguration einer Storeview

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

Funktion der Komponenten

Kundendaten

Generelle Kundendaten

Kunden können über verschiedene Wege in Magento registriert werden: über den Registrierungsprozess im Allgemeinen, über den Registrierungsprozess während einer Bestellung oder von einem Shopadministrator. Im Standardmodus (Option 9 steht auf „Nein“) werden die Kunden nur angelegt, wenn auf die Option für den Newsletter angeklickt wurde.
Steht Option 9 auf „Ja“, wird der Kontakt mit der Erlaubnis „SOI“ in Maileon angelegt. Ausnahme: hat sich ein Kunde einmal absichtlich in Maileon vom Newsletter abgemeldet wird er nicht wieder automatisch angemeldet.
Für Neuregistrierungen werden in Magento die Hooks `customer_save_before` und `customer_address_save_before` verwendet. Diese werden auch bei Kontaktdatenänderungen aufgerufen. Änderungen vom Kunden oder einem Administrator werden dabei sofort an Maileon weitergeleitet. Bei der Newsletteran-/abmeldung gibt es jedoch eine Besonderheit: entfernt ein Kunde in den Einstellungen den Haken beim Newsletter wird er abgemeldet , fügt er ihn wieder hinzu wird er ohne DOI-Bestätigungsmail mit DOI+ Erlaubnis zu Maileon hinzugefügt, da er sich ja im Magento-Administrationsbereich befindet und somit authentifiziert ist.

Abmeldungen

Abmeldungen aus Magento heraus werden wie beschrieben direkt mit Maileon synchronisiert, Abmelder die sich bei Maileon abgemeldet haben werden durch einen Cron-Job (`TT_checkMaileonForUnsubscribers`) in der Standardeinstellung alle 20 Minuten von Maileon abgefragt. Da sich Kunden zum Beispiel mehrmals innerhalb kurzer Zeit an-/abmelden können wird hierbei die Änderungszeit des Status eines Kunden in die Zeitzone des Maileon-REST-API Servers übersetzt und verglichen welcher Status der aktuellste ist. Ist der aktuellste Status eine Abmeldung, wird der Kontakt auch in Magento als Abmelder registriert. Die Zeitzone des Maileon-Servers ist „`Europe/Berlin`“.

DOI-Bestätigungen

Wird das Magento-DOI-Verfahren verwendet, so wird die Bestätigung direkt über die Maileon-Rest-API an Maileon weitergeleitet. Wird das Maileon-DOI-Verfahren verwendet wird ein Webhook aufgerufen der diese Daten an Magento weiterleitet. In Maileon wird der Webhook unter `Einstellungen` → `Konto` → `Webhooks` erstellt, siehe Abbildung 6. Dazu wird als Kontaktereignis DOI gewählt. Die POST-URL ist die URL die aufgerufen werden muss. Hier ist das Schema:

`http://myshop.com/de/xqueue/contact/acknowledgeDOI`.

Das „/de/“ muss in der URL angegeben werden, wenn die Ländererweiterung aktiviert ist, sonst muss es weggelassen werden. Als Parameter muss „`email`“ mit dem Kontaktfeldwert „`Emailadresse`“ und „`token`“ mit dem Wert, welcher unter Option 10 angegeben wurde, angegeben werden.

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

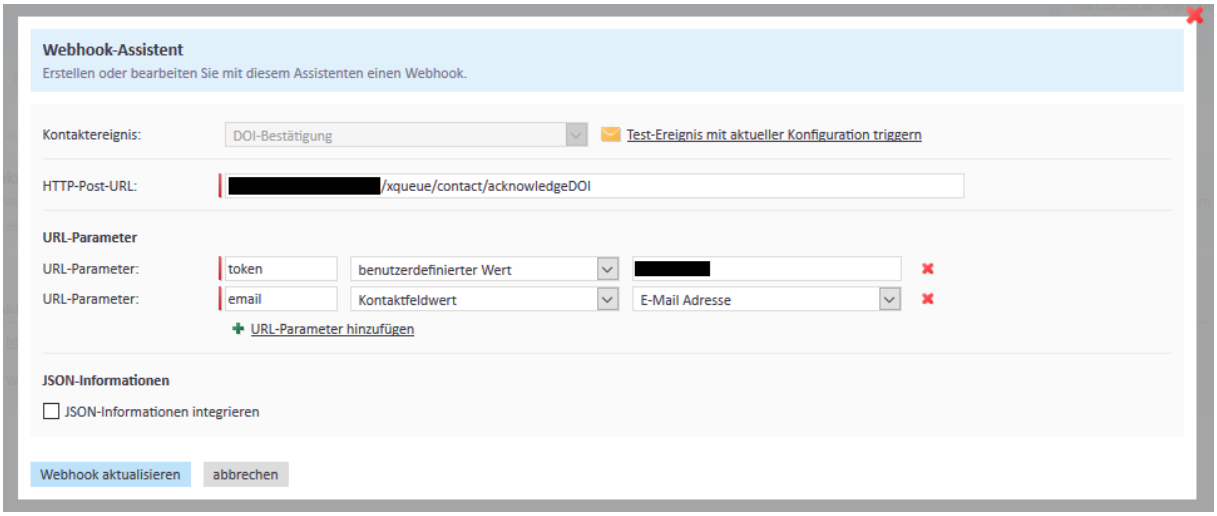

Abbildung 6: Webhook-Einstellungen in Maileon

Gespeicherte Daten:

1. 'SALUTATION'
2. 'GENDER'
3. 'LOCALE'
4. 'GENDER'
5. 'FIRSTNAME'
6. 'LASTNAME'
7. 'FULLNAME'
8. 'CITY'
9. 'COUNTRY'
10. 'ADDRESS'
11. 'ZIP'

Custom-Werte an Maileon übergeben

Aktuell unterstützt das Modul Customwerte für Kontakte nur durch eine Änderung des entsprechenden Codes in der Datei app/code/local/TT/XQueue/Helper/Data.php. Zum Zeitpunkt der Dokumentation betrifft dies die beiden Zeilen 134 und 166

Zeile 134: $newContact->custom_fields = array('createdByTransaction' => $createdByTransaction);
Zeile 166: $newContact->custom_fields = array('createdByTransaction' => false);

In diesen Zeilen wird ein Wert an das Maileon-Customfeld "createdByTransaction" übergeben. Hier könnte nun zum Beispiel die StoreView-ID oder das StoreView-Kürzel übergeben werden.

Die Zeilen würden sich dann wie folgt ändern:
Zeile 134: $newContact->custom_fields = array('createdByTransaction' => $createdByTransaction, 'magentoStoreViewCode' => Mage::app()->getStore()->getCode());
Zeile 166: $newContact->custom_fields = array('createdByTransaction' => false, 'magentoStoreViewCode' => Mage::app()->getStore()->getCode());

Hier wird das Modul angewiesen dem Maileon-Customfeld „magentoStoreViewCode“ den Code der aktuellen StoreView zuzuweisen. Dieser Wert kann dann verwendet werden um etwa alle Kontakte einer Storeview herauszufiltern. **Es muss sichergestellt sein, dass es die hier verwendeten Felder in Maileon gibt und den notwendigen Datentyp (in der Regel einfach „Text“) repräsentieren. Bitte auch die Groß- und Kleinschreibung beachten!**

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

Bestellungen

Abgeschlossene Bestellungen werden über den Magento-Hook `sales_order_place_after` abgefangen und als Transaktion an Maileon über die REST-API übertragen.
Dabei werden derzeit folgende Werte übergeben:

1. 'order.date'
2. 'order.total'
3. 'order.currency' (derzeit nur €)
4. 'generic_fields.string1' (Kommaseparierte Liste aller IDs der bestellten Produkte)
5. 'generic_fields.string2' (Kommaseparierte Liste aller Kategorien der bestellten Produkten)
6. 'customer.salutation'
7. 'customer.full_name'
8. 'customer.id'
9. 'billing.fullname'
10. 'billing. address.line1' (Straße, Hausnummer)
11. 'billing. address.line2' (PLZ)
12. 'billing. address.line3' (Stadt)
13. 'billing. address.line4' (Bundesland)
14. 'shipping.fullname'
15. 'shipping. address.line1' (Straße, Hausnummer)
16. 'shipping. address.line2' (PLZ)
17. 'shipping. address.line3' (Stadt)
18. 'shipping. address.line4' (Bundesland)
19. <Custom-Implementation-Attribute, siehe Ende dieses Kapitels>

Weiterhin werden für jedes Produkt im Warenkorb folgende Informationen übergeben:

1. 'id'
2. 'sku'
3. 'name'
4. 'single_price'
5. 'thumbnail'
6. 'image'
7. 'quantity'
8. 'combined_price'
9. 'short_description'
10. 'description'
11. <Custom-Attribute, siehe Abbildung 2 [13]>
12. <Custom-Implementation-Attribute, siehe Ende dieses Kapitels>

Maileon bedingt es derzeit, dass zu jeder Transaktion ein Kontakt mit der Emailadresse existiert damit die Transaktion einem „Kontakt“ zugeordnet werden kann. Sollte eine Transaktion für eine Emailadresse ausgelöst werden die in Maileon nicht existiert, dann wird diese von Maileon abgelehnt. Dieses Verhalten soll in naher Zukunft geändert werden aber aus diesem Grund wird in diesem Fall derzeit ein SOI-Kontakt angelegt. Dieser bleibt auch nach Absenden der Transaktion bestehen, damit die History der gesendeten Transaktionen nicht beeinflusst wird.
Hinweis: Die Kontakte werden markiert indem das Custom-Field `createdByTransaction` auf „`true`“ gesetzt wird. Sollen diese Kontakte nicht beschickt werden sollte ein Kontaktfilter verwendet werden, welcher diese Kontakte ausschließt. Der Wert wird bei einer etwaigen Registrierung für einen Newsletter überschrieben.

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

Custom Attributes
Spezielle Attribute (in Magento auch Custom Options oder Custom Attributes genannt) die in das Datenmodell von Magento eingefügt wurden können den Produkten/Items der Transaktion hinzugefügt werden, indem ein Mapping, wie in Abbildung 2 [13] dargestellt, übergeben wird.
Wenn diese Funktionalität jedoch nicht ausreicht können eigene Implementierungen vorgenommen werden, welche z.B. Daten aus einer externen Datenbank sammeln und der Transaktion hinzufügen. Die Implementierung muss in einer speziellen Helper-Klasse erfolgen:
Klasse: `TT_XQueue_Helper_External_Data`
Datei: `app/code/local/TT/XQueue/Helper/external/Data.php`

Die Methode `getCustomProductAttributes(Mage_Catalog_Model_Product $product)` wird für die Produkte von Bestellungen und Warenkorbabbrüchen verwendet. Hier können die Attributnamen frei gewählt werden und etwa zusätzliche Bilder oder Informationen zu verwandten Produkten gespeichert werden.
Zusätzliche Informationen über die Bestellung können über die Methode `getCustomOrderAttributes(Mage_Sales_Model_Order $order)` hinzugefügt werden. Hier ist es wichtig darauf zu achten, dass nur Feldnamen verwendet werden die gemäß der Spezifikation des Transaction Types in Maileon existieren. Hinweise dazu finden sich an der entsprechenden Stelle im Quellcode dieser Methode.

Warenkorbabbrecher

Warenkorbabbrecher sind Kunden die Produkte in ihren Warenkorb legen und diesen dann nicht bestellen. Wenn der Kunde eine Emailadresse hinterlegt hat kann diese genutzt werden um ihn an den Warenkorb zu erinnern.
Dazu werden zwei Cron-Jobs eingesetzt: der erste (`TT_markAbandonedCarts`) analysiert den Datenbestand und legt verwaiste Warenkörbe in einer Tabelle ab (`xqueue_queue`). Ein zweiter Cron-Job (`TT_sendAbandonedCartsEmails`) liest aus dieser Tabelle die Warenkorbabbrecher und erstellt daraus eine Transaktion in Maileon.
Erneutes Senden einer Warenkorbabbrecher-Transaktion an den gleichen Kontakt: es wird nur eine einzelne Transaktion für den Kontakt an Maileon gesendet. Eine erneute Transaktion kann nur ausgelöst werden, wenn der Kontakt entweder einen weiteren Warenkorb nicht bestellt oder den Inhalt des Warenkorbs seit Absenden der Transaktion geändert hat.

Dabei werden derzeit folgende Werte übergeben:

1. `'order.date'`
2. `'order.total'`
3. `'order.currency' (derzeit nur €)`
4. `'customer.salutation'`
5. `'customer.full_name'`
6. `'customer.id'`
7. `<Custom-Implementation-Attribute, siehe Ende dieses Kapitels>`

Weiterhin werden für jedes Produkt im Warenkorb folgende Informationen übergeben:

1. `'id'`
2. `'sku'`
3. `'name'`
4. `'single_price'`
5. `'thumbnail'`
6. `'image'`
7. `'quantity'`

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

```
8.  'combined_price'
9.  'short_description'
10. 'description'
11. <Custom-Attribute, siehe Abbildung 2 [18]>
12. <Custom-Implementation-Attribute, siehe Ende dieses Kapitels>
```

Custom Attributes

Für Warenkorbabbrüche gelten die gleichen Möglichkeiten wie für Bestellungen. Es wird jedoch eine gesonderte Methode für die Warenkörbe (in Magento: Quotes) bereitgestellt:
`getCustomQuoteAttributes(Mage_Sales_Model_Quote $quote)`
Auch hier ist es wichtig darauf zu achten, dass nur Feldnamen verwendet werden die gemäß der Spezifikation des Transaction Types in Maileon existieren. Hinweise dazu finden sich ebenfalls an der entsprechenden Stelle im Quellcode dieser Methode.

Funktionsweise der Warenkorb-Analyse

Die Analyse der abgebrochenen Warenkörbe basiert zunächst auf 3 Basistabellen

1. sales_flat_quote
   Diese Tabelle stammt von Magento und wird lesend verwendet um zu analysieren welche Warenkörbe ein gewisses Alter haben.
2. xqueue_queue
   Der Job der Warenkörbe aus Tabelle 1 analysiert schreibt Aufträge für Erinnerungsmails in diese Tabelle.
3. xqueue_log
   Ein zweiter Job arbeitet die Einträge aus Tabelle 2 ab. Dazu nimmt er die Einträge, generiert und sendet eine Anfrage an Maileon, schreibt die Anfrage in die xqueue_log Tabelle und löscht dann den Eintrag aus Tabelle 2. Der Job der die Warenkörbe analysiert vergleicht die Warenkorb-ID mit dieser Tabelle und findet so heraus ob bereits für diesen Warenkorb eine Erinnerung vorliegt.

© 2019 XQueue GmbH. Alle Rechte vorbehalten.

FAQ

Q: Wenn ich mich nicht als Kunde eingeloggt habe aber mich mit einer E-Mail-Adresse eines Kunden einloggen möchte kommt eine Fehlermeldung „Signup for the newsletter is not working, another user is using the same email address“. Wieso ist das so und wie kann man dies abschalten?

A: Dieses Verhalten kommt von Magento selbst. Der zugehörige Code befindet sich in der Klasse app\code\core\Mage\Newsletter\controllers\SubscriberController.php, ca. um Zeile 60 herum. Zum Ändern des Verhaltens muss die Klasse überschrieben werden, wobei der Fall einfach auskommentiert werden kann.

Q: Ich verwende OneStepCheckout und wnen ich mich im Bestellprozess für den Newsletter anmelde bekommt ein Kunde 2 DOI Mails, wieso?

A: OneStepCheckout hat einen Aufruf für eine DOI während der Abarbeitung der Bestelldaten und einen welcher Kunden nach Abarbeitung registriert. Durch einen Bug in manchen Versionen schließen sich die beiden Fälle jedoch nicht aus, sodass ein Kunde beide DOIs bekommt. Das Problem wurde bisher behoben indem die Klasse app/code/local/Idev/OneStepCheckout/Block/Checkout überschrieben wurde. In dem Code-Abschnitt der für Anmelde-Registrierungen zuständig ist wurde dann der Flag auf false gesetzt, welcher die zweite Registrierung auslöst.

© 2019 XQueue GmbH. Alle Rechte vorbehalten.