# EasyCouponExtension

Entwicklerdokumentation

André Fischer

30.03.2026

# Inhaltsverzeichnis

# 1 Einordnung

- **Admin-Label:** „ActiveCampaign EasyCoupon Erweiterung“
- **Plugin-Klasse:** `ItkComputerGmbh\\EasyCouponExtension\\EasyCouponExtension`

---

# 2 Dateien und Verantwortlichkeiten

| Komponente | Pfad |
|---|---|
| Controller | src/Controller/CouponWebhookController.php |
| Gutschein-Logik + Mail | src/Service/CouponService.php |
| Kundengruppe | src/Service/CustomerGroupService.php |
| System-Config | src/Resources/config/config.xml |
| DI | src/Resources/config/services.xml |
| Routen-Import | `routes.xml` → Controller-Verzeichnis |
| Twig-Mail-Fallback | src/Resources/views/send-coupon-mail.html.twig (Namespace @EasyCouponExtension/…) |

---

# 3 Routen (CouponWebhookController)

Alle mit **defaults: [‘auth_required’ => false, ’_routeScope’ => [‘api’]]**.

| Methode (Symfony-Route) | Pfad | Name |
|---|---|---|
| **handleWebhookWithRule** | /api/v{version}/activecampaign/webhook/{couponId} | api.action.activecampaign.webhook.rule |
| **testSalesChannelId** | /api/v{version}/test | api.action.test.saleschannelid |
| **handleWebhookWithCustomerGroup** | /api/v{version}/activecampaign/webhook/customergroup/{customerGroupId} | api.action.activecampaign.webhook.customergroup |

**Security:** Query-Parameter **x-sec-key** muss mit `EasyCouponExtension.config.securityKey` übereinstimmen, **wenn** der Key in der Config nicht leer ist. Gutschein-Webhook nutzt **`CouponService::getSecurityKey()`**, Kundengruppen-Webhook **`CustomerGroupService::getSecurityKey()`** – beide lesen dieselbe Config.

---

## 4 CouponService::createCouponCode

Signatur: **createCouponCode(string $copyCouponId, string $email, Context $context): array**

**Ablauf (vereinfacht):**

1. Vorlage laden (**EqualsFilter** auf **id** = **copyCouponId**) mit Associations **conditions**, **discount**, `discount.rules`, `discount.discountProducts`.
2. Neue UUID für den Gutschein; Bedingungen per **`copyConditions()`** duplizieren (Parent/Child-Sortierung).
3. Code: **strtoupper(substr(md5($email . time() . rand()), 0, 8))**.
4. **salesChannels**: Es werden **alle** Verkaufskanäle aus `sales_channel.repository` an den neuen Gutschein gehängt – nicht nur ein konfigurierter Kanal.
5. **Dedup: uniqueHash = md5($couponId . validUntilString . email . copyCouponTitle)**; Suche nach **ContainsFilter** auf **comment** mit Prefix **Generated by EasyCouponExtension -** + Hash und **deleted = false**. Treffer → leeres Array **[]** zurück (kein Exception, damit AC nicht retried).

6. **couponData**-Array: Kopie vieler Felder von der Vorlage, neuer **discount**-Block (kopiert oder Default), **validUntil** von Vorlage **oder** aus Config **couponDuration** (Tage ab jetzt), wenn Vorlage kein Datum hat.
7. Wenn Vorlage **validUntil** in der **Vergangenheit** → **Exception('Invalid date')**.
8. **easyCouponRepository->create([$couponData], $context)**; Rückgabe das Datenarray inkl. **code**, **value**, **translations**, …

**Neti-Typen:** Import `NetInventors\\NetiNextEasyCoupon\\Core\\Content\\Condition\\ConditionEntity` für die Bedingungslogik.

---

## 5 CouponWebhookController nach createCouponCode

- Wenn **createCouponCode []** liefert: Response `500` mit JSON **success: true** und Message „user already got a coupon…" – semantisch eher `409`/`200`, aber so im Code.
- Sonst `getSalesChannelIdFromRequest(request)` und **sendCouponCodeViaEmail($coupon, $email, $context, $salesChannelId, $copyCoupon->getComment())**.

**`getComment()`** der Vorlage: Wird als **optionaler Mail-Template-Hinweis** an **sendCouponCodeViaEmail** übergeben: Wenn es eine **UUID** eines `mail_template` ist, wird das Template aus der DB geladen und mit Twig gerendert; sonst Fallback (HTML-String als Twig-Template) oder Bundle-Twig `@EasyCouponExtension/send-coupon-mail.html.twig`.

---

## 6 CustomerGroupService

**assignCustomerGroupByEmail(string $customerGroupId, string $email, Context $context): bool**

1. Kundengruppe per ID muss existieren, sonst **Exception**.
2. Kunde per **EqualsFilter** auf **email** – kein Treffer → **false** (Controller antwortet 404).
3. **`customer.repository->update`** mit **groupId** (kein **requestedGroupId**-Flow).

---

## 7 Mailversand

`CouponService::sendCouponCodeViaEmail` baut `MailService::send`-Payload mit **recipients**, **senderName**/**senderEmail** aus Config, gerendertem HTML/Plain, **salesChannelId** aus Parameter.

---

## 8 CLI-Befehle

| Command | Klasse | Anmerkung |
|---|---|---|
| itk:coupon:create | **CreateCouponCommand** | Nach **createCouponCode** wird bei Mail optional **sendCouponCodeViaEmail( $couponData, $email, $context, $copyCoupon->getComment())** aufgerufen – **Signatur** ist aber **(…, string $salesChannelId, ?string $template = null)**. Der **vierte Parameter ist fälschlich der Kommentar-String**, nicht die Sales-Channel-ID. Das ist ein **Bug** im Command; E-Mail aus CLI kann so fehlschlagen oder falschen Kanal nutzen. |
| itk:coupon:list | **ListCouponsCommand** | |
| itk:coupon:count | **CountCouponsCommand** | |
| itk:coupon:create-from-file | **CreateCouponCodesFrom FileCommand** | |

---

## 9 Logging

`monolog.logger.custom` → `var/log/itk_newsletter_coupons.log`, Level DEBUG (100) laut Handler-Argument.

---

# 10 Architektur-Skizze