# Maison Common Quick Order Plugin v1.3.1

## Technische Dokumentation für Entwickler und IT-Mitarbeiter

## Architektur-Übersicht

### Modularer Aufbau

Das Plugin folgt einer **modularen Architektur** mit klarer Trennung der Verantwortlichkeiten:

maison-common-quick-order/  
├── maison-common-quick-order.php # Basis-Plugin (gemeinsame Funktionen)  
├── includes/  
│ ├── class-mc-order-system.php # Order-System (bestehende Funktionalität)  
│ └── class-mc-sampling-system.php # Sampling-System (EAN-Scanner + Abmusterung)  
├── assets/  
│ ├── js/  
│ │ ├── mc-collections-ajax.js # Order-System JavaScript  
│ │ └── mc-sampling-ajax.js # Sampling-System JavaScript  
│ └── css/  
│ ├── mc-collections-style.css # Order-System Styles  
│ └── mc-sampling-style.css # Sampling-System Styles  
└── CHANGELOG.md

### Klassen-Struktur

MC\_Quick\_Order // Basis-Klasse (gemeinsame Funktionen)  
├── MC\_Order\_System // Order-System (Collections, Cart Totals)  
└── MC\_Sampling\_System // Sampling-System (EAN-Scanner, Abmusterung)

**Design-Prinzipien:** - **Single Responsibility:** Jede Klasse hat eine klar definierte Aufgabe - **Dependency Injection:** Systeme sind lose gekoppelt - **WordPress Standards:** Hooks, Filters, und WordPress Coding Standards - **Backward Compatibility:** Bestehende Funktionen bleiben unverändert

## System-Requirements

### Mindestanforderungen

* **WordPress:** 5.0+
* **WooCommerce:** 4.0+
* **PHP:** 7.4+ (empfohlen: 8.0+)
* **MySQL:** 5.6+ oder MariaDB 10.1+

### Plugin-Dependencies

* **WooCommerce Quick Order Plugin:** Für Order-System Tabellen
* **User Switching Plugin:** Optional für Kundenwechsel-Funktionalität

### Browser-Kompatibilität

* **Desktop:** Chrome 70+, Firefox 65+, Safari 12+, Edge 79+
* **Mobile:** iOS Safari 12+, Chrome Mobile 70+
* **JavaScript:** ES6+ Features werden verwendet

### Hardware (Sampling-System)

* **EAN-Scanner:** HID-kompatible Bluetooth-Scanner
* **Unterstützte Formate:** EAN-13, EAN-8, UPC-A
* **Verbindung:** Bluetooth HID oder USB HID

## Installation & Setup

### 1. Plugin-Installation

# Via WordPress-Backend  
1. Plugin-ZIP hochladen  
2. Aktivieren  
3. Abhängigkeiten prüfen  
  
# Via FTP/SFTP  
1. Dateien nach /wp-content/plugins/maison-common-quick-order/ kopieren  
2. Berechtigungen setzen (755 für Ordner, 644 für Dateien)  
3. Plugin im Backend aktivieren

### 2. Abhängigkeiten installieren

// Erforderlich für Order-System  
- WooCommerce Quick Order Plugin  
  
// Optional für erweiterte Funktionen  
- User Switching Plugin (für Kundenwechsel)

### 3. Konfiguration

#### Collections für Sampling markieren

// Im WordPress-Backend: Produkte → Collections  
1. Collection bearbeiten  
2. Checkbox "Abmusterungs-Collection" aktivieren  
3. Speichern

#### EAN-Felder konfigurieren

Das Plugin sucht automatisch in folgenden Meta-Feldern:

$ean\_fields = array(  
 '\_ean', // Standard  
 '\_ean13', // EAN-13 spezifisch  
 '\_barcode', // Allgemein  
 '\_gtin' // Global Trade Item Number  
);

#### EK-Preise einrichten

// Produktebene: Produkte → Produkt bearbeiten → Preise  
- Feld "EK-Preis (Einkaufspreis)" ausfüllen  
  
// Varianten-Ebene: Bei variablen Produkten  
- Jede Variante kann eigenen EK-Preis haben

## Shortcode-Referenz

### Order-System

#### [mc\_quick\_order]

Zeigt das Order-System mit Collections-Navigation und Quick Order Tabelle.

// Basis-Verwendung  
[mc\_quick\_order]  
  
// Mit spezifischer Collection  
[mc\_quick\_order collection\_id="123"]  
  
// Ohne Collections-Navigation  
[mc\_quick\_order show\_collections="false"]

#### [mc\_cart\_totals]

Zeigt EK/VK-Summen des aktuellen Warenkorbs.

// Standard-Anzeige  
[mc\_cart\_totals]  
  
// Nur EK-Summe  
[mc\_cart\_totals show\_vk="false"]  
  
// Nur VK-Summe  
[mc\_cart\_totals show\_ek="false"]  
  
// Custom CSS-Klassen  
[mc\_cart\_totals class="custom-totals-style"]

### Sampling-System

#### [mc\_sampling]

Zeigt das Sampling-System mit EAN-Scanner und Quick Order Integration.

// Automatische Collection-Erkennung  
[mc\_sampling]  
  
// Spezifische Collection für Vorführungen  
[mc\_sampling collection\_id="123"]  
  
// Nur Scanner ohne Collections-Navigation  
[mc\_sampling show\_collections="false"]  
  
// Nur Scanner ohne Quick Order Tabelle  
[mc\_sampling show\_table="false"]  
  
// Nur Tabelle ohne Scanner  
[mc\_sampling show\_scanner="false"]  
  
// Kombinierte Optionen  
[mc\_sampling collection\_id="123" show\_collections="false" show\_table="true"]

## AJAX-Endpoints

### Order-System

#### mc\_load\_collection

Lädt Produkte einer spezifischen Collection.

// Request  
{  
 action: 'mc\_load\_collection',  
 collection\_id: 123,  
 nonce: mc\_collections\_ajax.nonce  
}  
  
// Response  
{  
 success: true,  
 data: {  
 html: '<div class="quick-order-table">...</div>',  
 collection\_name: 'Frühjahr 2024'  
 }  
}

#### mc\_search\_customers

Sucht Kunden für User Switching Integration.

// Request  
{  
 action: 'mc\_search\_customers',  
 search: 'Mustermann',  
 nonce: mc\_collections\_ajax.nonce  
}  
  
// Response  
{  
 success: true,  
 data: [  
 {  
 id: 123,  
 display\_name: 'Max Mustermann',  
 user\_email: 'max@example.com'  
 }  
 ]  
}

#### mc\_get\_cart\_totals

Holt aktuelle Warenkorb-Summen.

// Request  
{  
 action: 'mc\_get\_cart\_totals',  
 nonce: mc\_collections\_ajax.nonce  
}  
  
// Response  
{  
 success: true,  
 data: {  
 ek\_total: '€ 45,50',  
 vk\_total: '€ 89,90',  
 item\_count: 3  
 }  
}

### Sampling-System

#### mc\_sampling\_add\_item

Fügt Abmusterungs-Item über EAN-Code hinzu.

// Request  
{  
 action: 'mc\_sampling\_add\_item',  
 ean\_code: '1234567890123',  
 collection\_id: 123,  
 nonce: mc\_sampling\_ajax.nonce  
}  
  
// Response  
{  
 success: true,  
 data: {  
 message: 'Produkt hinzugefügt',  
 product\_name: 'Beispiel-Produkt',  
 artikel\_id: 'ART-123'  
 }  
}

## Datenbank-Schema

### Custom Meta-Fields

#### Produkt-Meta

-- EK-Preise  
meta\_key: '\_ek\_price'  
meta\_value: '45.50'  
  
-- EAN-Codes (verschiedene Felder unterstützt)  
meta\_key: '\_ean'  
meta\_value: '1234567890123'

#### Collection-Meta

-- Sampling-Collection Markierung  
meta\_key: 'is\_sampling\_collection'  
meta\_value: '1'

#### Order-Meta (Sampling-Bestellungen)

-- Unterschrift erforderlich  
meta\_key: 'requires\_signature'  
meta\_value: '1'  
  
-- Unterschrift-Timestamp  
meta\_key: 'signature\_timestamp'  
meta\_value: '2024-03-15 14:30:00'

#### Cart-Item-Meta (Sampling-Items)

-- Abmusterungs-Flag  
'is\_sampling' => true  
  
-- Artikel-ID (Parent-SKU)  
'artikel\_id' => 'ART-123'  
  
-- Original EAN (bei Scanner-Eingabe)  
'scanned\_ean' => '1234567890123'  
  
-- Größe-Bezeichnung  
'sampling\_size' => 'Abmusterung'

## Integration Points

### WooCommerce Integration

#### Cart-System

// Custom Cart Items für Sampling  
add\_filter('woocommerce\_add\_cart\_item\_data', 'mc\_add\_sampling\_cart\_item\_data');  
add\_filter('woocommerce\_get\_item\_data', 'mc\_display\_sampling\_cart\_item\_data');  
add\_action('woocommerce\_checkout\_create\_order\_line\_item', 'mc\_save\_sampling\_order\_item\_meta');

#### Pricing-System

// EK-Preis Integration  
add\_action('woocommerce\_product\_options\_pricing', 'mc\_add\_ek\_price\_field');  
add\_action('woocommerce\_process\_product\_meta', 'mc\_save\_ek\_price\_field');  
add\_filter('woocommerce\_quick\_order\_meta\_keys', 'mc\_add\_ek\_price\_to\_meta\_keys');

#### Checkout-System

// Unterschriften-Integration  
add\_action('woocommerce\_checkout\_process', 'mc\_validate\_sampling\_signature');  
add\_action('woocommerce\_checkout\_update\_order\_meta', 'mc\_save\_signature\_meta');

### Quick Order Plugin Integration

#### Meta-Keys Extension

// EK-Preis zu Quick Order Meta-Keys hinzufügen  
add\_filter('woocommerce\_quick\_order\_meta\_keys', function($meta\_keys) {  
 if (!in\_array('\_ek\_price', $meta\_keys)) {  
 $meta\_keys[] = '\_ek\_price';  
 }  
 return $meta\_keys;  
});

#### Table Data Modification

// Formatierung der EK-Preise in Quick Order Tabellen  
add\_filter('woocommerce\_quick\_order\_table\_data', 'mc\_modify\_quick\_order\_table\_data');

### User Switching Integration

#### Customer Search

// AJAX-Handler für Kundensuche  
add\_action('wp\_ajax\_mc\_search\_customers', 'mc\_ajax\_search\_customers');  
add\_action('wp\_ajax\_nopriv\_mc\_search\_customers', 'mc\_ajax\_search\_customers');

## Debugging & Troubleshooting

### Debug-Modus aktivieren

// In wp-config.php  
define('WP\_DEBUG', true);  
define('WP\_DEBUG\_LOG', true);  
define('WP\_DEBUG\_DISPLAY', false);

### Log-Ausgaben

Das Plugin schreibt Debug-Informationen in das WordPress Error-Log:

// Order-System Logs  
error\_log('MC Quick Order: ajax\_load\_collection called');  
error\_log('MC Quick Order: Collection ID: ' . $collection\_id);  
  
// Sampling-System Logs  
error\_log('MC Sampling: EAN scan attempt: ' . $ean\_code);  
error\_log('MC Sampling: Product found: ' . $product\_id);

### Häufige Probleme

#### EAN-Scanner funktioniert nicht

// Browser-Konsole prüfen  
console.log('Scanner input detected:', eanCode);  
  
// Mögliche Ursachen:  
1. Scanner nicht als HID-Device erkannt  
2. JavaScript-Fehler in mc-sampling-ajax.js  
3. AJAX-Endpoint nicht erreichbar

#### Quick Order Tabelle lädt nicht

// WordPress Debug-Log prüfen  
1. AJAX-Nonce ungültig  
2. Collection-ID nicht gefunden  
3. Quick Order Plugin nicht aktiv

#### EK-Preise werden nicht angezeigt

// Mögliche Ursachen:  
1. \_ek\_price Meta-Field nicht gesetzt  
2. Quick Order Plugin Cache nicht geleert  
3. JavaScript-Formatierung fehlgeschlagen

### Performance-Optimierung

#### AJAX-Caching

// Collection-Daten cachen  
$cache\_key = 'mc\_collection\_' . $collection\_id;  
$cached\_data = get\_transient($cache\_key);  
if ($cached\_data === false) {  
 // Daten laden und cachen  
 set\_transient($cache\_key, $data, HOUR\_IN\_SECONDS);  
}

#### Asset-Loading

// Conditional Loading nur auf relevanten Seiten  
if (has\_shortcode($post->post\_content, 'mc\_quick\_order') ||   
 has\_shortcode($post->post\_content, 'mc\_sampling')) {  
 wp\_enqueue\_script('mc-collections-ajax');  
 wp\_enqueue\_style('mc-collections-style');  
}

## Security

### Nonce-Validierung

// Alle AJAX-Requests verwenden WordPress Nonces  
if (!wp\_verify\_nonce($\_POST['nonce'], 'mc\_collections\_nonce')) {  
 wp\_die('Security check failed');  
}

### Input-Sanitization

// Alle Eingaben werden sanitized  
$collection\_id = absint($\_POST['collection\_id']);  
$search\_term = sanitize\_text\_field($\_POST['search']);  
$ean\_code = sanitize\_text\_field($\_POST['ean\_code']);

### Capability-Checks

// Benutzer-Berechtigungen prüfen  
if (!current\_user\_can('edit\_shop\_orders')) {  
 wp\_die('Insufficient permissions');  
}

## Wartung & Updates

### Plugin-Updates

// Version-Check in Basis-Plugin  
if (get\_option('mc\_quick\_order\_version') !== MC\_QUICK\_ORDER\_VERSION) {  
 // Update-Routine ausführen  
 mc\_run\_update\_routine();  
 update\_option('mc\_quick\_order\_version', MC\_QUICK\_ORDER\_VERSION);  
}

### Datenbank-Wartung

-- Verwaiste Meta-Daten bereinigen  
DELETE pm FROM wp\_postmeta pm  
LEFT JOIN wp\_posts p ON pm.post\_id = p.ID  
WHERE p.ID IS NULL AND pm.meta\_key LIKE '\_ek\_price';

### Cache-Management

// Plugin-spezifische Caches leeren  
function mc\_clear\_plugin\_cache() {  
 global $wpdb;  
 $wpdb->query("DELETE FROM {$wpdb->options} WHERE option\_name LIKE '\_transient\_mc\_%'");  
}

## API-Erweiterungen

### Custom Hooks

#### Actions

// Sampling-Item hinzugefügt  
do\_action('mc\_sampling\_item\_added', $cart\_item\_key, $product\_data);  
  
// Collection gewechselt  
do\_action('mc\_collection\_switched', $old\_collection\_id, $new\_collection\_id);  
  
// Cart Totals berechnet  
do\_action('mc\_cart\_totals\_calculated', $ek\_total, $vk\_total);

#### Filters

// EAN-Felder anpassen  
$ean\_fields = apply\_filters('mc\_sampling\_ean\_fields', $default\_ean\_fields);  
  
// Sampling-Collections filtern  
$collections = apply\_filters('mc\_sampling\_collections', $collections);  
  
// Cart Totals Format anpassen  
$formatted\_total = apply\_filters('mc\_cart\_totals\_format', $total, $type);

### REST-API Integration (geplant v1.4.0)

// Geplante Endpoints  
GET /wp-json/mc/v1/collections  
GET /wp-json/mc/v1/sampling/{collection\_id}  
POST /wp-json/mc/v1/sampling/add-item  
GET /wp-json/mc/v1/cart-totals

## Deployment

### Staging-Environment

# Plugin-Dateien kopieren  
rsync -av --exclude='.git' maison-common-quick-order/ staging/wp-content/plugins/maison-common-quick-order/  
  
# Datenbank-Migration  
wp db export production.sql  
wp db import production.sql --url=staging.example.com

### Production-Deployment

# Backup erstellen  
wp db export backup-$(date +%Y%m%d).sql  
  
# Plugin aktualisieren  
wp plugin update maison-common-quick-order  
  
# Cache leeren  
wp cache flush

### Monitoring

// Error-Monitoring  
add\_action('wp\_ajax\_mc\_\*', function() {  
 if (defined('WP\_DEBUG') && WP\_DEBUG) {  
 error\_log('MC Plugin AJAX call: ' . $\_POST['action']);  
 }  
});

## Roadmap

### Version 1.4.0 (Q2 2024)

* **REST-API:** Vollständige API für externe Integration
* **Verkaufsbuch-Export:** PDF-Generierung aus Abmusterungen
* **Erweiterte Scanner-Unterstützung:** Multi-Device-Support

### Version 1.5.0 (Q3 2024)

* **Offline-Modus:** PWA-Features für iPad-Nutzung
* **Advanced Analytics:** Reporting-Dashboard
* **LSE-Connector:** Direkte Warenwirtschafts-Integration

### Version 2.0.0 (Q4 2024)

* **Headless-Architecture:** React-Frontend
* **Multi-Tenant:** Support für mehrere Mandanten
* **AI-Integration:** Intelligente Produktvorschläge

## Support-Kontakte

### Entwicklung

* **Lead Developer:** Michael Deinhardt
* **Repository:** Internal GitLab
* **Documentation:** Confluence

### Deployment

* **Staging:** staging.maisoncommon.com
* **Production:** shop.maisoncommon.com
* **Monitoring:** New Relic / WordPress Health Check

*Diese Dokumentation wird kontinuierlich aktualisiert. Letzte Änderung: Version 1.3.1 - Juni 2024*