Einstellungen ohne Maske

Einstellungen, die es nur in der config.json gibt: Bestandsberechnung über eigene Abfragen, Feldzuordnungen und Sonderfälle der einzelnen Module.

Grundlagen

Die meisten Einstellungen von FINN.ghost haben eine Maske in der Oberfläche. Ein Teil hat keine — sie entstanden für einzelne Installationen und werden ausschließlich in der `config.json` gepflegt. Dieses Kapitel führt sie auf.

> Diese Einstellungen sind bewusst nicht in der Oberfläche. Sie greifen tief in den Ablauf einer Übertragung ein, und ein falscher Wert fällt oft erst auf, wenn im Shop falsche Bestände oder falsche Preise stehen. Wer sie setzt, sollte wissen, was der jeweilige Lauf sonst tun würde.

Wie sie eingetragen werden

Der Schlüssel steht in der `config.json` unter dem Abschnitt des Moduls. Der Teil vor dem Punkt ist der Abschnitt, der Teil dahinter der Name in diesem Abschnitt:

```
{
  "sw6": {
    "customStock": "SELECT ... AS Bestand FROM ...",
    "ignoreReserved": true
  },
  "shopify": {
    "zeroStockField": "_KEINBESTAND"
  }
}
```

Die Abschnitte der Module:

Abschnitt	Modul
`sw6`	Shopware 6
`shopify`	Shopify
`shop`	FINN.webshop
`gmi`	GetMyInvoices
`mail2sl`	FINN.mail2SL
`mcp`	FINN.AI MCP

Abschnitt	Modul
ohne Abschnitt	gilt für die ganze Installation

> Geändert wird nur bei angehaltenem Dienst und nur mit einer Kopie der bisherigen Datei. Bei laufendem Dienst geht die Änderung verloren, weil die Anwendung die Einstellungen im Speicher hält und die Datei beim nächsten Speichern komplett neu schreibt. Das Vorgehen steht unter Anpassungen über die config.json.

Was die Werte bedeuten

Die Tabellen dieses Kapitels nennen je Einstellung den Typ. Vier kommen vor:

Typ	Bedeutung
Schalter	`true` oder `false`. Fehlt der Schlüssel, gilt die Vorgabe.
Feldname	Name einer Spalte oder eines Extrafelds in der SelectLine — zeichengenau, ohne Tabellenpräfix.
Abfrage	vollständige SQL-Abfrage als Text, mit Platzhalter (siehe Bestand und Bestellmenge).
Wert	eine Zahl oder ein Text, etwa eine Preisgruppe oder ein Belegtyp.

> Bei den Einstellungen vom Typ **Feldname** wird ein Flag in der SelectLine ausgewertet. Gemeint ist dabei durchgängig ein Ja/Nein-Extrafeld, das als Text `True` gespeichert wird — so legt die SelectLine Ja/Nein-Extrafelder ab.

> Ein Feldname, den es in der Mandantendatenbank nicht gibt, führt zu einem Fehler in der Abfrage — nicht zu einer Meldung in der Oberfläche. Steht ein Lauf plötzlich mit einem SQL-Fehler im Protokoll, ist ein Tippfehler in einem dieser Felder die erste Vermutung.

Aufbau des Kapitels

Seite	Inhalt
Bestand und Bestellmenge	eigene Abfragen für Bestand und maximale Bestellmenge, für alle drei Shops
Shopware 6	Artikel, Artikelgruppen, Kunden, Belege

Seite	Inhalt
Shopify	Artikel, Belege, Kunden, B2B
Weitere Module	FINN.webshop, Bilder, Oberfläche, SelectLine, GMI, mail2SL, MCP

> Vieles, was diese Einstellungen leisten, geht auch mit einem Makro — und ein Makro ist in der Oberfläche sichtbar und dokumentiert sich damit selbst. Wo beides möglich ist, ist das Makro der bessere Weg. Siehe Makros.

Nächster Schritt

Bestand und Bestellmenge

Bestand und Bestellmenge

Standardmäßig ermittelt FINN.ghost den Bestand eines Artikels aus der SelectLine-Ansicht `SL_vGetFreierArtikelBestand`: **Bestand minus Reserviert**, bei Stücklisten die kleinste Stückzahl, die sich aus den Bestandteilen bauen lässt. Für alle Fälle, in denen das nicht passt, gibt es `customStock` — eine eigene SQL-Abfrage, die den Bestand liefert.

> Typische Gründe: Es sollen nur bestimmte Läger in den Shop gemeldet werden, ein Sicherheitsbestand soll abgezogen werden, oder ein Sperrlager soll unberücksichtigt bleiben.

customStock

Alle drei Shop-Anbindungen kennen die Einstellung. Der Schlüssel ist derselbe, Abschnitt und Platzhalter unterscheiden sich:

Anbindung	Schlüssel	Platzhalter
Shopware 6	`sw6.customStock`	`${article.Number}`
Shopify	`shopify.customStock`	`${nummer}`
FINN.webshop	`shop.customStock`	`${artikelnummer}`

> Der Platzhalter ist je Anbindung ein anderer. Wird der falsche verwendet, wird er nicht ersetzt — die Abfrage läuft dann mit dem Text `${nummer}` als Artikelnummer, findet nichts und liefert Bestand **0** für jeden Artikel.

Was die Abfrage liefern muss

Eine Spalte mit dem Namen **`Bestand`**. Ausgewertet wird die **erste Zeile** des Ergebnisses.

Das Beispiel meldet nur die Läger `01` und `02` und zieht die reservierte Menge ab:

```
{
  "sw6": {
    "customStock": "SELECT SUM(ISNULL(Bestand,0)-ISNULL(Reserviert,0)) AS Bestand FROM
dbo.SL_vGetFreierArtikelBestandMitLager WHERE Artikelnummer = '${article.Number}' AND Lager IN
```

```
('01','02')"
  }
}
```

> Die Ansicht `SL_vGetFreierArtikelBestandMitLager` ist für solche Abfragen der bequemste Einstieg: dieselben Werte wie in der Standardberechnung, nur zusätzlich je Lager. Für Standorte statt Läger gibt es `SL_vGetFreierArtikelBestandMitStandort`.

Spalte heißt nicht `Bestand`	Ergebnis wird als 0 gewertet
Abfrage liefert keine Zeile	Bestand 0
Ergebnis ist negativ	wird auf 0 angehoben
Mehrere Zeilen	nur die erste zählt

> Der Wert wird als ganze Zahl übertragen. Nachkommastellen fallen weg — bei Artikeln, die in Metern oder Kilogramm geführt werden, ist das zu bedenken.

Was die Abfrage ersetzt

Ist `customStock` gesetzt, entscheidet **allein** die Abfrage. Die Standardlogik läuft nicht mehr — und damit auch nichts, was daran hängt:

Läuft nicht mehr	Folge
Abzug von **Reserviert**	muss in der Abfrage selbst stehen
Auflösung von **Stücklisten**	Set- und Behälterartikel liefern den Wert der Abfrage, nicht die baubare Menge

> Die Stücklistenauflösung ist der Punkt, der am häufigsten übersehen wird. Ohne `customStock` meldet FINN.ghost für einen Set-Artikel die Menge, die sich aus den Bestandteilen bauen lässt. Mit `customStock` liefert die Abfrage den Wert — und der eigene Lagerbestand eines Set-Artikels ist in der Regel 0.

Welche Artikel überhaupt übertragen werden, ändert `customStock` **nicht**: Die Auswahl nach *Shopaktiv* und *Inaktiv* passiert davor, beim Zusammenstellen der Artikelliste. Ein nicht shopaktiver Artikel wird also weiterhin gar nicht erst gemeldet.

> Bei **Shopify** ist das anders: Dort gehört die Prüfung auf *Shopaktiv* und *Inaktiv* zur Bestandsermittlung selbst und wird von `shopify.customStock` mit übergangen — ebenso

`shopify.zeroStockField`. Beides gehört dann in die Abfrage.

Rückfall auf die Standardberechnung (nur Shopware 6)

Bei Shopware 6 gibt es einen Ausweg für einzelne Artikel: Liefert die Abfrage den Wert `-9999`, verwirft FINN.ghost das Ergebnis und rechnet für diesen Artikel wie ohne `customStock`.

```
SELECT CASE WHEN ... THEN 0 ELSE -9999 END AS Bestand FROM ART WHERE Artikelnummer =
'${article.Number}'
```

Damit lässt sich die Sonderlogik auf die Artikel beschränken, die sie brauchen. Alle anderen laufen weiter über die geprüfte Standardberechnung — inklusive Stücklistenauflösung.

Shopify und FINN.webshop kennen diesen Rückfall nicht. Dort gilt die Abfrage für jeden Artikel.

Besonderheit bei Shopify

Shopify führt Bestände je Standort. `shopify.customStock` liefert **einen** Wert, und dieser wird auf **alle** eingerichteten Standorte geschrieben — nicht aufgeteilt.

Bei mehreren Standorten summiert Shopify die Bestände. Ein Wert von 10 bei drei Standorten wird im Shop zu 30 verfügbaren Stück. Wer mit mehreren Standorten arbeitet, sollte `customStock` dort nicht einsetzen.

Zusätzlich greift die Abfrage nur, wenn die Bestandsführung für den Artikel aktiv ist. Ist sie abgeschaltet, überträgt Shopify keinen Bestand — dann läuft die Abfrage nicht.

Weitere Einstellungen zum Bestand

Shopware 6

Schlüssel	Typ	Wirkung
`sw6.ignoreReserved`	Schalter	rechnet mit **Bestand** statt Bestand minus Reserviert. Reservierungen aus offenen Aufträgen mindern den Shopbestand dann nicht.
`sw6.customPurchase`	Abfrage	eigene Abfrage für die **maximale Bestellmenge**. Platzhalter `${article}`, Spalte `Bestand`.
`sw6.maxPurchaseField`	Feldname	Artikelfeld mit der maximalen Bestellmenge je Artikel.

> Ist `sw6.maxPurchaseField` gesetzt und im Artikel gefüllt, gilt dieser Wert — `sw6.customPurchase` kommt dann für diesen Artikel nicht mehr zum Zug. Das Feld hat Vorrang.

> `sw6.ignoreReserved` wirkt nur, wenn `sw6.customStock` **nicht** gesetzt ist oder für den Artikel `-9999` liefert. Sonst rechnet ohnehin nur die eigene Abfrage.

Shopify

Schlüssel	Typ	Wirkung
`shopify.zeroStockField`	Feldname	Artikelfeld; ist es `True`, wird Bestand **0** gemeldet, obwohl Bestand vorhanden ist. Wirkt auch über den Variantenartikel und über Stücklisten: hat ein Bestandteil das Feld gesetzt, ist die ganze Stückliste 0.
`shopify.inventoryTrackDisableField`	Feldname	Artikelfeld; ist es `True`, wird die **Bestandsführung** für diesen Artikel in Shopify abgeschaltet. Der Artikel ist dann unbegrenzt bestellbar.

> `shopify.zeroStockField` ist der saubere Weg, einzelne Artikel im Shop auf nicht verfügbar zu setzen, ohne sie in der SelectLine anzufassen — und deutlich einfacher als eine eigene Abfrage.

Wenn der Bestand im Shop nicht stimmt

Beobachtung	Erste Prüfung
alle Artikel stehen auf 0	Platzhalter falsch geschrieben, oder Spalte heißt nicht `Bestand`
Stücklisten stehen zu niedrig	`customStock` löst Stücklisten nicht auf
bei Shopify: gesperrte Artikel bestellbar	`shopify.zeroStockField` wird von der Abfrage übergangen
Shopify zeigt ein Vielfaches	mehrere Standorte, der Wert wird je Standort geschrieben
Bestand ändert sich nicht mehr	Lauf steht mit SQL-Fehler im Protokoll

Die Abfrage lässt sich vor dem Eintragen im SQL Management Studio prüfen — dort den Platzhalter durch eine echte Artikelnummer ersetzen. Meldungen der Läufe stehen im Protokoll, siehe Log Informationen.

Nächster Schritt

Shopware 6

Shopware 6

Alle Schlüssel dieser Seite stehen in der `config.json` im Abschnitt `sw6`. Was der Artikelexport im Regelfall überträgt, steht im Handbuch für Shopware 6.

Artikeltexte und Bezeichnungen

Schlüssel	Typ	Wirkung
`titleField`	Feldname	Extrafeld, dessen Inhalt als **Produktname** übertragen wird — statt der Artikelbezeichnung.
`descriptionField`	Feldname	Extrafeld, dessen Inhalt als **Beschreibung** übertragen wird — statt des Artikeltexts. Zeilenumbrüche werden zu ` `.
`disableDescription`	Schalter	überträgt **keine** Beschreibung. Vorhandene Texte im Shop bleiben unangetastet.
`disableManufacturerDescription`	Schalter	überträgt keine Beschreibung zum Hersteller.
`customSearchKeywordsField`	Feldname	Artikelfeld, aus dessen Inhalt die **Suchworte** des Produkts gebildet werden.
`zusatzInTitleSeparator`	Wert	Trennzeichen zwischen Bezeichnung und Zusatz, Vorgabe `" - "`. Wirkt nur, wenn *Zusatz im Titel* eingeschaltet ist.

> `disableDescription` ist der übliche Weg, wenn die Beschreibungen im Shop redaktionell gepflegt werden. Ohne diesen Schalter überschreibt jeder Artikelexport den Text im Shop mit dem Artikeltext aus der SelectLine — die redaktionelle Arbeit wäre beim nächsten Lauf weg.

> `titleField` und `descriptionField` lesen **Extrafelder** über die SelectLine-API, `customSearchKeywordsField` liest eine **Spalte der Tabelle ART** direkt per SQL. Für `customSearchKeywordsField` ist deshalb der Spaltenname anzugeben, für die anderen beiden der Name des Extrafelds.

Preise, Grundpreise und Datumsfelder

Schlüssel	Typ	Wirkung
`disableSafeguardPriceField`	Feldname	Artikelfeld; ist es `True`, wird ein Artikel **ohne Preis** mit Preis 0 übertragen, statt übersprungen zu werden.
`grundpreisMengumField`	Feldname	Feld in den Mengeneinheiten (`MENGUM`); die so markierte Mengeneinheit liefert Grundpreis-Einheit und Faktor.
`grundpreisReferenceField`	Feldname	Artikelfeld mit der **Bezugsmenge** des Grundpreises. Vorgabe `FreierText2`.
`grundpreisUnitField`	Feldname	Artikelfeld mit der **Einheit** des Grundpreises. Vorgabe `FreieZahl3`.
`releaseDateField`	Feldname	Extrafeld, dessen Datum als **Erscheinungsdatum** übertragen wird. Ohne die Angabe gilt das Anlagedatum des Artikels.
`restockTimeField`	Feldname	Extrafeld mit der **Wiederauffüllzeit** in Tagen.

> Die drei `grundpreis`-Schlüssel wirken nur, wenn der Grundpreis-Export überhaupt eingeschaltet ist. Ist `grundpreisMengumField` gesetzt, hat es Vorrang; erst ohne dieses Feld werden die beiden Artikelfelder gelesen. Die Vorgaben `FreierText2` und `FreieZahl3` greifen also auch dann, wenn nichts eingetragen ist — was zu Grundpreisen aus einem Feld führen kann, das für etwas anderes genutzt wird.

> Ohne `disableSafeguardPriceField` übergeht FINN.ghost Artikel ohne gültigen Preis absichtlich und schreibt eine Meldung ins Protokoll. Der Schalter ist für Artikel gedacht, die im Shop bewusst mit 0 stehen sollen — er sollte nicht dazu dienen, fehlende Preise zu verdecken.

Zusatzfelder umbenennen

Schlüssel	Typ	Wirkung
`extrafeldMapping`	Liste	benennt Zusatzfelder beim Übertragen um.

Schlüssel	Typ	Wirkung
`extrafeldMappingDelete`	Schalter	Vorgabe `true`: das Ursprungsfeld wird nach dem Umbenennen entfernt. Mit `false` bleiben **beide** Felder erhalten.

```
{
  "sw6": {
    "extrafeldMapping": [
      { "old": "custom_sl_artikel_FreierText1", "new": "custom_meine_farbe" }
    ],
    "extrafeldMappingDelete": false
  }
}
```

> Die Umbenennung greift auch in den Übersetzungen, sofern Sprachen eingerichtet sind. Das Zielfeld muss in Shopware als Zusatzfeld existieren — angelegt wird es dabei nicht.

Artikelgruppen und Kategorien

Schlüssel	Typ	Wirkung
`shopaktivArtikelGruppenFeld`	Feldname	Feld an der Artikelgruppe; nur Gruppen mit `True` werden als Kategorie übertragen.
`artikelgruppenNichtAktualisieren`	Schalter	bestehende Kategorien werden **nicht** mehr aktualisiert. Neue werden weiter angelegt.

> `artikelgruppenNichtAktualisieren` ist für Shops gedacht, deren Kategoriebaum in Shopware weitergepflegt wird — mit eigenen Bezeichnungen, eigener Sortierung oder eigener Einordnung. Ohne den Schalter zieht jeder Lauf die Kategorien wieder auf den Stand der SelectLine.

Varianten und Bilder

Schlüssel	Typ	Wirkung

`auffaechernDefault`	Schalter	**alle** Variantenartikel werden in der Storefront aufgefächert, unabhängig vom dafür eingerichteten Artikelfeld.
`useHashAsFilename`	Schalter	Bilddateien werden unter einem Prüfsummennamen abgelegt statt unter *Hersteller-Bezeichnung*.

Ist `auffaechernDefault` gesetzt, greift die Einstellung *Erste Variante in der Storefront anzeigen* nicht mehr — beides zusammen ergibt keinen Sinn.

`useHashAsFilename` hilft, wenn Bezeichnungen Zeichen enthalten, die als Dateiname Probleme machen. Nach dem Umstellen entstehen für bereits übertragene Bilder neue Dateien; die alten bleiben in der Shopware-Medienverwaltung liegen und müssen dort aufgeräumt werden.

Makro vor dem Artikelexport

Schlüssel	Typ	Wirkung
`preExportArticles`	Wert	Dateiname eines SelectLine-Makros, das **vor jedem einzelnen Artikel** ausgeführt wird. Übergeben wird der Parameter `Nummer` mit der Artikelnummer.

Das Makro läuft je Artikel — bei einem Vollexport also zehntausende Male. Seine Laufzeit bestimmt damit die Dauer des ganzen Laufs. Eingetragen wird der **Dateiname** des Makros, nicht seine Bezeichnung; siehe Makros.

Bei Shopify und FINN.webshop gibt es dasselbe Makro in den Einstellungen der Oberfläche. Nur bei Shopware 6 fehlt die Maske dafür.

Kunden

Schlüssel	Typ	Wirkung
`kdfeld`	Feldname	Kundenfeld, in dem die **Shop-Kundennummer** steht. Vorgabe `Shopnummer`.

Schlüssel	Typ	Wirkung
`disableCustomerCreate`	Schalter	im Shop werden **keine** neuen Kunden angelegt. Bestehende werden weiter aktualisiert.
`onlyUpdateZahlungsbedingung`	Schalter	überträgt von bestehenden Kunden **nur** die Zahlungsbedingung, sonst nichts.

> Die Kombination aus beiden Schaltern ist der Weg für Shops, in denen die Kundenpflege im Shop stattfindet und aus der SelectLine nur die Zahlungsbedingung nachgeführt werden soll.

Belege

Schlüssel	Typ	Wirkung
`defaultWarehouseLocationNumber`	Wert	Lagerplatz, der in jeden importierten Beleg geschrieben wird.
`slPriceToleranceField`	Feldname	Belegfeld, in das bei einer **Preisabweichung** zwischen Shop und SelectLine der Hinweistext geschrieben wird.

> Eine Preisabweichung landet in jedem Fall als Fehler im Protokoll. `slPriceToleranceField` macht sie zusätzlich am Beleg sichtbar, damit sie in der SelectLine auffällt und nicht nur im Protokoll steht.

Technische Sonderfälle

Schlüssel	Typ	Wirkung
`index`	Schalter	Vorgabe `false`: FINN.ghost bittet Shopware, die **Indizierung** während der Übertragung auszusetzen. Mit `true` indiziert Shopware bei jedem Schreibzugriff mit.
`oldSWVersion`	Schalter	schreibt die Zeiträume von Preisregeln im Format älterer Shopware-Versionen.
`appServerUrl`	Wert	Adresse, die Shopware zur Bestätigung der App zurückruft. Vorgabe `https://localhost:8080`.

>

Bei `index` ist die Vorgabe `false` die richtige Einstellung: Bei einem Vollexport mit vielen Artikeln würde die laufende Indizierung den Shop erheblich ausbremsen. Nach einem großen Lauf gehört im Shop einmal neu indiziert, damit Suche und Kategorien vollständig sind.

`appServerUrl` wird nur für die App-Registrierung gebraucht und muss die Adresse sein, unter der Shopware diese FINN.ghost-Installation erreicht.

Nächster Schritt

Shopify

Shopify

Alle Schlüssel dieser Seite stehen in der `config.json` im Abschnitt `shopify`. Was der Artikel- und Bestellabgleich im Regelfall tut, steht im Handbuch für Shopify.

Artikel und Preise

Schlüssel	Typ	Wirkung
`continueSelling`	Feldname	Artikelfeld; liefert es einen Wert, bleibt der Artikel bei Bestand 0 **weiter bestellbar**.
`forceTax`	Schalter	überträgt jeden Artikel als **steuerpflichtig**, auch wenn in der SelectLine kein gültiger Steuersatz gefunden wird.
`reducedTax`	Wert	Steuercode der SelectLine, der als **ermäßigt** gilt. Vorgabe `2`. Artikel mit diesem Code kommen in die Artikelgruppe für ermäßigte Steuer.
`costPricePG`	Wert	Preisgruppe, aus der die **Kosten** je Artikel übertragen werden. Ohne die Angabe gilt der letzte Einkaufspreis aus der Kalkulation.
`firstImgToEnd`	Schalter	verschiebt das **erste Bild an das Ende** der Bilderreihe.
`disableSafeguardPriceField`	Feldname	Artikelfeld; ist es `True`, wird ein Artikel **ohne Preis** mit Preis 0 übertragen statt übersprungen.
`extrafeldMapping`	Liste	benennt Zusatzfelder beim Übertragen um, wie bei Shopware 6. Das Ursprungsfeld wird dabei immer entfernt.

> Ohne `disableSafeguardPriceField` wird ein Artikel ohne gültigen Preis in Shopify **archiviert** und eine Meldung ins Protokoll geschrieben. Das ist Absicht — ein Artikel mit Preis 0 im Shop ist teurer als ein fehlender.

>

`firstImgToEnd` wirkt nur bei Artikeln mit mehr als einem Bild. Der Hintergrund: Shopify zeigt das erste Bild als Vorschaubild, die SelectLine sortiert Bilder nach der Ordnung — passt beides nicht zusammen, ist das die schnellste Abhilfe.

Bei `continueSelling` wird — anders als bei den übrigen Feldern dieses Kapitels — **nicht** auf den Wert `True` geprüft. Ausgewertet wird nur, ob das Feld überhaupt einen Wert liefert. Ein Ja/Nein-Extrafeld ist hier deshalb die falsche Wahl: Es enthält auch bei *Nein* einen Wert. Passend ist ein Feld, das für die betroffenen Artikel gefüllt und für alle anderen leer ist.

`reducedTax` vergleicht den *Steuercode* des Artikels, nicht den Steuersatz. Bei abweichender Nummerierung der Steuercodes im Mandanten muss der Wert angepasst werden — sonst landen entweder alle oder keine Artikel in der Gruppe für ermäßigte Steuer.

Artikelgruppen und Aufräumen

Schlüssel	Typ	Wirkung
`shopaktivArtikelGruppenFeld`	Feldname	Feld an der Artikelgruppe; nur Gruppen mit `True` werden übertragen.
`deleteUnknownProducts`	Schalter	**archiviert** Produkte im Shop, die in der SelectLine nicht existieren oder nicht mehr shopaktiv sind.

`deleteUnknownProducts` greift auf den ganzen Shop zu. Produkte, die im Shop von Hand angelegt wurden und in der SelectLine nicht existieren, werden dabei ebenfalls archiviert. Vor dem Einschalten gehört geprüft, ob es solche Produkte gibt.

Bestände

Bestand und Bestandsführung stehen auf einer eigenen Seite, weil dort mehrere Einstellungen zusammenwirken: Bestand und Bestellmenge.

Belege

Schlüssel	Typ	Wirkung

`idField`	Feldname	Belegfeld, in dem die **Shopify-Bestellkennung** abgelegt wird. Vorgabe `IhrAuftrag`.
`orderTag`	Wert	Kürzel, mit dem die Kennung beginnt. Vorgabe `sfy_`.
`mailField`	Feldname	Belegfeld für die **E-Mail-Adresse** des Bestellers.
`phoneField`	Feldname	Belegfeld für die **Telefonnummer**.
`noteToFooter`	Schalter	schreibt die Kundenanmerkung in den **Schlusstext** statt in den Kopftext.
`useSfyPostext`	Schalter	übernimmt den **Positionstext aus Shopify** statt der Artikelbezeichnung aus der SelectLine. Auf 80 Zeichen gekürzt.
`documentImportCheckField`	Feldname	Belegfeld, das nach erfolgreichem Import auf `True` gesetzt wird.
`tooLongAdrField`	Feldname	Belegfeld, in dem Adressbestandteile über 80 Zeichen vermerkt werden.
`simplePrintStatus`	Schalter	erkennt den Versand daran, dass der Lieferschein als **gedruckt** markiert und der Beleg seit dem letzten Lauf bearbeitet wurde — statt am protokollierten Druckvorgang.
`mapMarketOrderIdField`	Feldname	Belegfeld für die **Amazon-Bestellnummer** aus den Bestellattributen. Vorgabe `_SFYMARKETORDERID`.
`businessPartnerContractDateWithTime`	Schalter	schreibt in das Vertragsdatum zusätzlich die **Uhrzeit**.
`noPosCheck`	Schalter	schaltet die Kontrolle ab, ob alle Positionen im Beleg angekommen sind.
`useCurrentQuantity`	Schalter	übernimmt die in Shopify **verbleibende Menge** statt der ursprünglich bestellten. In Shopify entfernte Positionen landen dadurch nicht im Beleg.

> `idField` und `orderTag` sind die Wiedererkennung einer Bestellung. Werden sie im laufenden Betrieb geändert, findet FINN.ghost die bereits importierten Bestellungen nicht mehr und importiert sie erneut. Beides gehört vor der Inbetriebnahme festgelegt und danach nicht mehr angefasst.

`noPosCheck` schaltet eine Sicherung ab: Normalerweise vergleicht FINN.ghost nach dem Anlegen die Anzahl der Positionen und bricht mit einem Fehler ab, wenn die SelectLine Positionen verworfen hat. Ohne diese Prüfung entstehen unvollständige Belege, ohne dass es auffällt.

`useCurrentQuantity` ist die Antwort auf entfernte Positionen: Wird eine Bestellung in Shopify nach dem Eingang bearbeitet — eine Position entfernt oder die Menge verringert — bleibt die ursprüngliche Menge (`quantity`) an der Position stehen, die verbleibende steht in `current_quantity`. Ohne den Schalter übernimmt FINN.ghost die ursprüngliche Menge und legt entfernte Positionen mit an. Mit dem Schalter zählt die verbleibende Menge; Positionen mit Menge 0 werden übersprungen und ein anteiliger Rabatt wird mitgekürzt. Greift nur bei Bestellungen, die zum Zeitpunkt des Imports bereits bearbeitet waren — ein bestehender Beleg wird nicht nachträglich angepasst.

`useCurrentQuantity` und die Gutschriftenprüfung (`disableRefundCheck` auf `false`) schließen sich aus. Die Gutschriftenprüfung braucht die volle Menge im Auftrag, um daraus die Gutschrift zu bilden — sind die Positionen bereits gekürzt, findet sie die Position nicht mehr und der Import läuft auf einen Fehler.

`tooLongAdrField` ist für Shops mit langen Firmenbezeichnungen gedacht. Die SelectLine nimmt nur 80 Zeichen je Adressfeld — was abgeschnitten wird, ist dann wenigstens am Beleg vermerkt.

`simplePrintStatus` ist die Abhilfe, wenn Versandmeldungen ausbleiben, obwohl Lieferscheine gedruckt werden. Normalerweise liest FINN.ghost den Druckvorgang aus der Belegausgabe — wird in der SelectLine nicht über den regulären Druck ausgegeben, entsteht dort kein Eintrag. Der Schalter wertet dann stattdessen das Kennzeichen *Gedruckt* am Beleg aus.

Gutschriften

Schlüssel	Typ	Wirkung
`belegtyp_gutschrift`	Wert	Belegtyp für Gutschriften. Vorgabe `G`.
`belegtyp_gutschriftNoRestock`	Wert	abweichender Belegtyp für Gutschriften **ohne Rücknahme in den Bestand**.
`disableRefundCheck`	Schalter	Vorgabe `true`. Mit `false` prüft FINN.ghost auch bei einem Komplettstorno auf Gutschriften.

> `belegtyp_gutschriftNoRestock` ist die Antwort auf eine häufige Anforderung: Wird eine Rücksendung in Shopify ohne Wiedereinlagerung erstattet, soll in der SelectLine ein Belegtyp entstehen, der den Bestand nicht erhöht.

Zuordnung über Tags und Metafelder

Schlüssel	Typ	Wirkung
`shopkundeTagMapping`	Liste	ordnet einem **Bestell-Tag** einen anderen Shopkunden zu.
`mapMetafields`	Liste	überträgt **Metafelder** der Bestellung in Belegfelder.
`presentment_currencies`	Liste	Währungen, in denen Bestellungen zusätzlich importiert werden.

```
{
  "shopify": {
    "shopkundeTagMapping": { "b2b": "10001", "haendler": "10002" },
    "mapMetafields": { "custom": { "wunschtermin": "FreiesDatum1" } }
  }
}
```

> Bei `mapMetafields` ist die Struktur zweistufig: erst der Namensraum des Metafelds, darin der Schlüssel, dahinter das Ziel-Belegfeld.

Kunden

Schlüssel	Typ	Wirkung
`kdfeld1` bis `kdfeld9`	Feldname	**weitere** Kundenfelder, in denen bei der Suche nach einem vorhandenen Kunden ebenfalls nach der Shopify-Kundennummer gesucht wird.
`customerNumberRange`	Wert	Nummernkreis für neu angelegte Kunden. Die neue Nummer ist die höchste vorhandene Nummer mit diesem Beginn, um eins erhöht.
`updateCustomer`	Schalter	aktualisiert vorhandene Kunden beim Bestellimport, auch wenn das Aktualisieren sonst abgeschaltet ist.

Schlüssel	Typ	Wirkung
`sendEmailInviteToNewCustomers`	Schalter	lässt Shopify neu angelegten Kunden eine **Einladung** senden.

> Die Felder `kdfeld1` bis `kdfeld9` ergänzen das in der Oberfläche eingestellte Kundenfeld, sie ersetzen es nicht. Ist `kdfeld1` gesetzt, schreibt FINN.ghost die Kundennummer bei neuen Kunden allerdings **nicht** mehr selbst in das Standardfeld — dann sind diese Felder die einzige Zuordnung und müssen gepflegt sein.

> Vor `sendEmailInviteToNewCustomers` ist zu bedenken, dass Shopify die Mail an jeden neu übertragenen Kunden schickt. Beim ersten Vollabgleich der Kunden aus der SelectLine sind das unter Umständen sehr viele Mails auf einmal.

B2B mit Shopify Plus

Schlüssel	Typ	Wirkung
`plus.status`	Schalter	schaltet den B2B-Betrieb ein: Kunden werden als **Firmen** übertragen statt als Privatkunden.
`plus.checkoutToDraft`	Schalter	Bestellungen der Firma werden als **Entwurf** angelegt und müssen freigegeben werden.
`plus.editableShippingAddress`	Schalter	die Lieferadresse darf im Shop geändert werden.
`plus.paymentTermsTemplateId`	Wert	Kennung der Zahlungsbedingung, die den Firmen zugewiesen wird.

> `plus.status` setzt Shopify Plus voraus und ändert die Übertragung grundlegend. Zusätzlich entfällt damit die Prüfung, ob eine Bestellung bezahlt ist — im B2B ist der Rechnungskauf der Regelfall, unbezahlte Bestellungen werden also importiert.

> Die Kennung für `plus.paymentTermsTemplateId` ist die reine Nummer der Vorlage aus Shopify; die vollständige Kennung setzt FINN.ghost selbst zusammen.

Nächster Schritt

Weitere Module

Weitere Module

Was in den übrigen Modulen keine Maske hat — und die Einstellungen, die für die ganze Installation gelten und deshalb ohne Abschnitt in der `config.json` stehen.

FINN.webshop

Abschnitt `shop`.

Schlüssel	Typ	Wirkung
`shopaktivArtikelGruppenFeld`	Feldname	Feld an der Artikelgruppe; nur Gruppen mit `True` werden übertragen.

Die eigene Bestandsabfrage `shop.customStock` steht unter Bestand und Bestellmenge.

Bilder aus dem Dateisystem

Ohne Abschnitt — gilt für alle Module, die Artikelbilder übertragen.

Schlüssel	Typ	Wirkung
`imagePath`	Wert	Ordner, aus dem die Artikelbilder gelesen werden — statt aus der SelectLine.

Die Dateien werden dem Artikel über den Dateinamen zugeordnet: **Artikelnummer, Unterstrich, laufende Nummer.**

```
100815_1.jpg
100815_2.jpg
100815_10.jpg
```

Sortiert wird nach der Zahl hinter dem Unterstrich, und zwar numerisch — `_10` kommt also nach `_2`, nicht dazwischen.

Ordner enthält Bilder zum Artikel	diese werden übertragen

Ordner enthält keine passende Datei	Rückfall auf die Bilder in der SelectLine
Ordner nicht erreichbar	Fehler; der Lauf bricht ab

> Ist der Ordner nicht erreichbar, wird nicht auf die SelectLine zurückgefallen, sondern der Lauf bricht mit einem Fehler ab. Bei einem Netzwerkpfad braucht das Dienstkonto Leserechte darauf — läuft der Dienst als lokales Systemkonto, hat es auf einem Netzwerkpfad in der Regel keine.

> Der Rückfall auf die SelectLine gilt je Artikel: Für Artikel mit Dateien im Ordner gelten die Dateien, für alle anderen die Bilder aus der SelectLine. Beides lässt sich also mischen.

> Bildformat und Bildqualität aus den Einstellungen gelten auch für diese Dateien — sie werden vor dem Übertragen umgewandelt. Siehe Server und Oberfläche.

Oberfläche

Ohne Abschnitt.

Schlüssel	Typ	Wirkung
`theme`	Wert	Farbe der Oberfläche, als Objekt mit dem Feld `primary`. Vorgabe `#AA1948`.
`disableBeep`	Schalter	Vorgabe `true`. Mit `false` gibt die Oberfläche bei einem Fehler einen **Signalton** aus.
`onlyCustom`	Schalter	zeigt **ausschließlich** die Seite der individuellen Anpassungen — ohne Menü, ohne Navigation.

```
{
  "theme": { "primary": "#1976D2" },
  "disableBeep": false
}
```

> `onlyCustom` verbirgt alle Module samt Einstellungen und Zeitsteuerung. Es ist für Installationen gedacht, die nur eine kundenspezifische Maske bereitstellen, und setzt eine Lizenz für individuelle Anpassungen voraus. Ohne Zugriff auf die `config.json` kommt man danach nicht mehr an die übrigen Masken.

>

> Der Signalton ist nützlich an Arbeitsplätzen, an denen die Oberfläche dauerhaft offen steht und Fehler auffallen sollen — etwa im Lager oder im Versand.

SelectLine

Schlüssel	Typ	Wirkung
`skipVersionscheck`	Schalter	übergeht die Prüfung, ob die SelectLine-Version mindestens der geforderten entspricht.
`slsettings.warehouseCheckColum`	Feldname	Spalte in `LAGERPLATZ`, über die ein Lagerplatz zusätzlich anhand einer Nummer gefunden wird.

> `skipVersionscheck` ist ausdrücklich ein Notbehelf. Die Mindestversion steht nicht ohne Grund: Fehlen der SelectLine-API Endpunkte, die FINN.ghost erwartet, laufen Übertragungen an unvorhersehbaren Stellen auf Fehler. Der Schalter hilft, eine Installation überhaupt zu starten — die richtige Abhilfe ist das Update der SelectLine.

> `slsettings.warehouseCheckColum` wird gebraucht, wenn Lagerplätze aus einem Drittsystem als reine Nummer kommen und nicht als die von der SelectLine erwartete Kombination aus Lager und Dimensionen. Zuerst wird immer regulär gesucht; erst wenn das nichts findet, greift diese Spalte.

GetMyInvoices

Abschnitt `gmi`.

Schlüssel	Typ	Wirkung
`statusField`	Feldname	Belegfeld, in das bei einer **Betragsabweichung** der Hinweis geschrieben wird. Vorgabe `FreierText1`.

> Weicht der Bruttobetrag ab, wird der Beleg nicht als geprüft markiert und der Hinweis samt Abweichung in Prozent in dieses Feld geschrieben. Ist `FreierText1` im Mandanten anderweitig belegt, gehört hier ein anderes Feld hinterlegt — sonst überschreibt der Abgleich vorhandene Inhalte.

FINN.mail2SL

Abschnitt `mail2sl`.

Schlüssel	Typ	Wirkung
`limit`	Wert	Anzahl der Nachrichten, die je Lauf aus dem Postfach geholt werden. Vorgabe `10`.

> Der Wert begrenzt die Nachrichten **je Lauf**, nicht je Tag. Bei einem Postfach mit hohem Aufkommen kann der Rückstand wachsen, wenn je Lauf weniger Nachrichten verarbeitet werden als neu eintreffen. Dann entweder den Wert erhöhen oder den Lauf häufiger einplanen.

> Ein höherer Wert verlängert den einzelnen Lauf entsprechend. Kommt es dabei zu Zeitüberschreitungen am Postfach, ist der häufigere Lauf der bessere Weg.

FINN.AI MCP

Abschnitt `mcp`. Ergänzt die Einstellungen, die die Oberfläche für den KI-Zugang anbietet.

Schlüssel	Typ	Wirkung
`maxSessions`	Wert	Höchstzahl gleichzeitiger Sitzungen. Vorgabe `50`.
`sessionTimeoutSeconds`	Wert	Zeit in Sekunden, nach der eine untätige Sitzung verworfen wird. Vorgabe `1800` (30 Minuten).
`bodyLimit`	Wert	Höchstgröße einer Anfrage. Vorgabe `"10mb"`.

> Diese drei Werte sind Schutzgrenzen und müssen im Normalbetrieb nicht angefasst werden. Sie sind gedacht für Installationen mit auffällig vielen gleichzeitigen Zugriffen oder mit knappem Speicher auf dem Server.

> Bei `bodyLimit` gehört die Einheit mit in den Wert und der Wert in Anführungszeichen — `"10mb"`, nicht `10`.

Individuelle Anpassungen

Abschnitt `custom`. Diese Schlüssel gehören zu kundenspezifischen Erweiterungen und sind nur dort gesetzt, wo die passende Anpassung ausgeliefert ist.

Schlüssel	Typ	Wirkung
`apiport`	Wert	Port einer zusätzlichen Schnittstelle für Fremdsysteme. Ohne die Angabe wird sie nicht gestartet.
`apiRechnungDruckvorlage`	Wert	Druckvorlage, mit der über diese Schnittstelle angeforderte Rechnungen erzeugt werden.

> Der Port muss frei sein und darf nicht mit dem Port der Oberfläche zusammenfallen. Fehlt `apiRechnungDruckvorlage`, beantwortet die Schnittstelle Anfragen nach einer Rechnung mit einem Fehler.

Nächster Schritt

Zurück zu den Häufigen Fragen oder zur Zeitsteuerung der Module.