Technische Dokumentation des Online-Shops

# Vorwort

Der Online-Shop besteht aus den beiden wesentlichen Sichten des Ladenbesitzers und des Kunden. Es gibt dabei genau einen Ladenbesitzer und beliebig viele Kunden, die jeweils über ein eigenes Kundenkonto verfügen.

Zum Austausch zwischen den Services dienen Hilfsklassen, die im Kapitel „Bestell-/Lieferorganisation“ beschrieben sind.

# Services

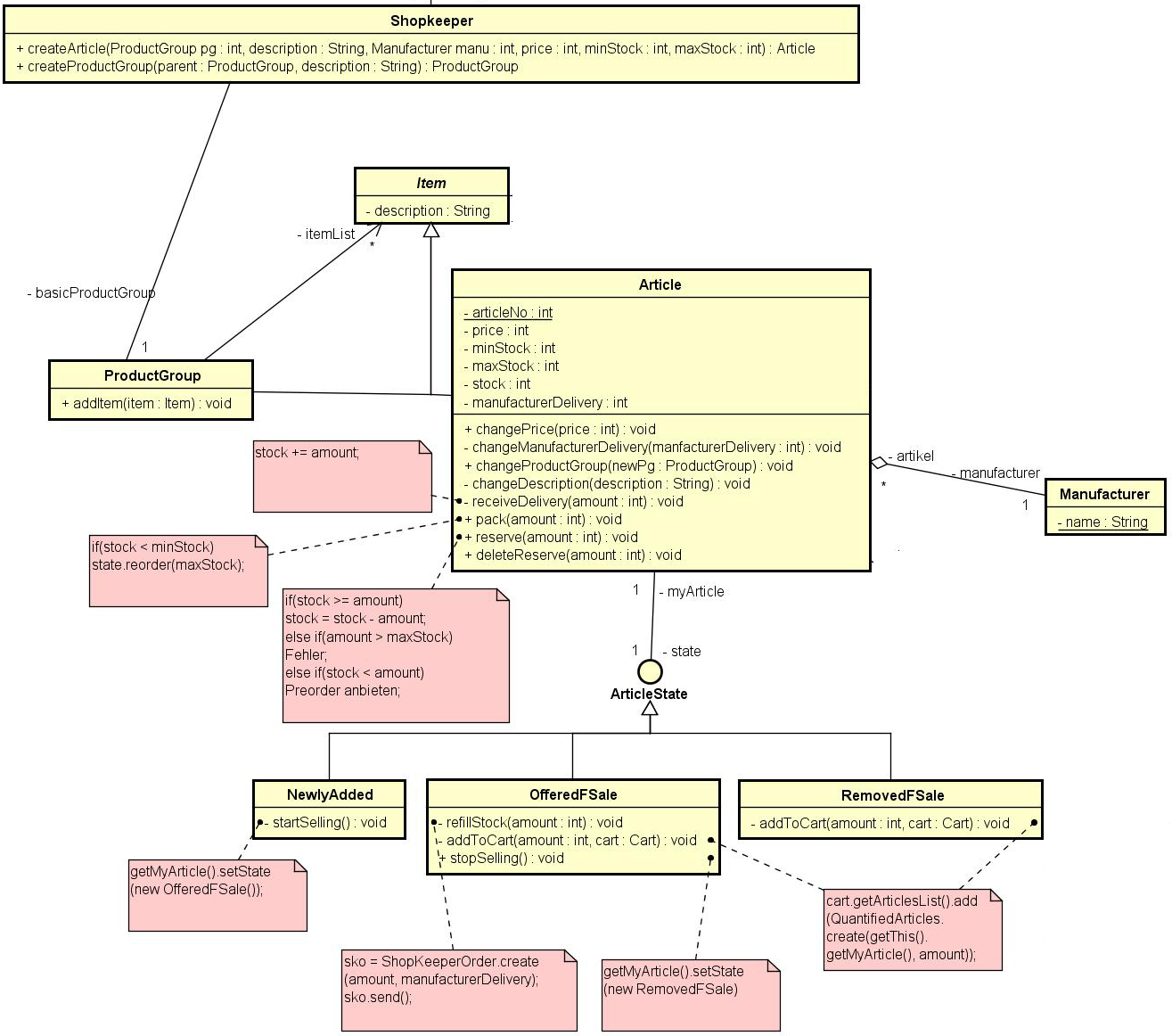
Die Architektur der Anwendung ist servicebasiert. Jede Sicht des Nutzers auf die Anwendung erfolgt über einen eigenen Service. Alle Services erben von der Klasse **Service** (siehe Klassenmodell). ***[|Ausschnitt aus dem Klassenmodell für den Service hier einügen!!|]***

Dem Ladenbesitzer steht der ShopkeeperService zur Verfügung und dem Kunden der ShopService, CustomerManager, AccountService sowie OrderService. Diese werden im Folgenden erläutert.

## Ladenbesitzer

Der Ladenbesitzer kann zentral auf die Lagerverwaltung zugreifen. Standardmäßig ist es der Benutzer „hugo“.

Das Lager bzw. der Ladenbetreiber-Service ist wie im folgenden Klassenmodell aufgebaut.



### Lager

Im Lager liegen alle Verkaufsgegenstände (Items). Es besitzt initial eine Produktgruppe namens „Produkte“, unter der alle weiteren Artikel und Produktgruppen angeordnet sind. (entspricht im Klassenmodell dem Shopkeeper)

### Gegenstände

Gegenstände (Items) sind abstrakte Verkaufseinheiten. Sie besitzen eine eindeutige Beschreibung. Implementierungen müssen ihre jeweilige Beschreibung ändern können sowie ihre Artikelanzahl zurückliefern können.

Gegenstände sind sowohl Artikel als auch Produktgruppen.

### Produktgruppen

Produktgruppen dienen zur Klassifikation und Kategorisierung von Artikeln. Sie besitzen einen eindeutigen Namen sowie eine Liste von Gegenständen.

Es besteht die Möglichkeit, über die Methode addItem(Item) einen Gegenstand zu der Produktgruppe hinzuzufügen.

### Artikel

Artikel sind konkrete Verkaufseinheiten. Sie besitzen jeweils einen Hersteller, eine Beschreibung, einen Mindest- sowie Maximalbestand, eine Herstellerlieferzeit sowie den momentanen Bestand. Ein Artikel ist immer unter einer Produktgruppe angeordnet, die beliebig gewechselt werden kann.

* **Beschreibung ändern –** changeDescription(String)
* **Artikelpreis ändern –** changePrice(Integer)
* **Herstellerlieferzeit ändern –** changeManuDelivery(Integer)
* **Produktgruppe wechseln –** changeProductGroup(ProductGroup)

Der Artikel kann sich in drei verschiedenen Zuständen befinden: der Artikel kann neu, im Verkauf oder aus dem Verkauf genommen sein. Statusspezifische Aktionen müssen von diesen direkt ausgeführt werden und werden nicht über den Article delegiert.

* **Neuer Artikel:** ein neuer Artikel kann nicht von Kunden bestellt werden. Dazu muss er in den Verkauf genommen werden, was durch die Methode startSelling(), die den Status auf OfferedFSale (Im Verkauf) setzt.
* **Im Verkauf:** Artikel, die im Verkauf sind, werden automatisch nachbestellt, wenn ihr momentaner Lagerbestand unter den Mindestlagerbestand fällt. Das Bestellverfahren entspricht dem Bestellpunktverfahren der verbrauchsgesteuerten Disposition. Mindest- sowie Maximalbestand sind also als dessen untere bzw. obere Grenze zu betrachten. Zur Beobachtung dient ein eigener Thread, der die Lagerbestände aller Artikel jede Sekunde kontrolliert ***Welcher?*** Und entsprechende Bestellungen auslöst.  
  Soll der Artikel nicht mehr nachbestellt werden, muss die Funktion stopSelling() ausgeführt werden, die den Artikel in einen RemovedFSale versetzt.  
  OfferedFSale bietet weiterhin die Funktion addToCart(...), mit Hilfe derer ein Artikel zum Einkaufswagen hinzugefügt wird.
* **Aus dem Verkauf genommen:** Artikel, die aus dem Verkauf genommen sind, werden zwar noch verkauft aber nicht mehr nachbestellt. Es können also vom Kunden auch keine Vorbestellungen ausgelöst werden. Die Methode StartSellingAgain() dagegen wechselt den Status wieder zurück auf OfferedFSale.

### Kundenkonten/Lieferbedingungen

Dem Ladenbesitzer stehen Möglichkeiten zur Verfügung, Kundenkonten zu prekonfigurieren.

In der Klasse serverConstants.ConfigConstants existieren statische Variablen, in denen Informationen zur Retourengebühr, des voreingestellten Kontostands sowie des unteren Limits neuer Kundenkonten zur Laufzeit abgelegt sind. Die Variablen können auf diese Weise klassenübergreifend ausgelesen und verändert werden.

Die dem Ladenbesitzer zur Verfügung stehenden Operationen in diesem Sinne editieren jene Attributwerte. Die Kundenkonten lesen die Werte PresetAccountBalance und PresetAccountLowerLimit einmalig bei Anlage aus und die RetourePercentage bei jedem Retourenvorgang.

## Kunde

# Bestell-/Lieferorganisation