



Sitecore E-Commerce Services 2.1 on CMS 7.0

# Order Manager API Reference Guide

*A developer's guide to the Order Manager API.*

## Table of Contents

Chapter 1	Introduction.....	4
Chapter 2	Order Manager Modules .....	5
2.1	Sitecore.Ecommerce.Core.Common .....	6
2.1.1	Address .....	6
2.1.2	Amount .....	7
2.1.3	Communication .....	8
2.1.4	Contact .....	8
2.1.5	IConfigurable .....	9
2.1.6	IEntity .....	9
2.1.7	Location .....	9
2.1.8	Measure .....	10
2.1.9	Party .....	10
2.1.10	PartyLegalEntity .....	11
2.1.11	PartyTaxScheme.....	12
2.1.12	Period .....	12
2.1.13	Person .....	13
2.1.14	ShopContext.....	13
2.2	Sitecore.Ecommerce.Core.OrderManagement.....	14
2.2.1	CoreOrderManager .....	14
2.2.2	StateManager .....	14
2.2.3	StateProvider.....	14
2.2.4	SubstateCombinationSet .....	15
2.2.5	ChangeOrderLineQuantityCommand.....	15
2.2.6	Command.....	15
2.2.7	ConfirmationMessageBuilder .....	16
2.2.8	CoreOrderMangerImpl .....	16
2.2.9	ILoggingEnabled .....	16
2.2.10	OrderConfirmation.....	17
2.2.11	OrderLineFactory .....	17
2.2.12	OrderProcessingStrategy .....	18
2.2.13	OrderProcessor .....	18
2.2.14	PlainTextConfirmationBuilder .....	19
2.2.15	ProcessingStrategy .....	19
2.2.16	CoreOrderStateConfiguration .....	19
2.2.17	MerchantOrderProcessor .....	20
2.3	Sitecore.Ecommerce.Core.OrderManagement.Orders.....	21
2.3.1	AllowanceCharge .....	21
2.3.2	CustomerParty .....	22
2.3.3	Delivery .....	22
2.3.4	Despatch .....	23
2.3.5	Item .....	24
2.3.6	LineItem.....	24
2.3.7	MonetaryTotal .....	26
2.3.8	Order .....	26
2.3.9	OrderedShipment.....	28
2.3.10	OrderLine .....	29
2.3.11	PaymentMeans .....	29
2.3.12	Price .....	30
2.3.13	ReservationTicket.....	30
2.3.14	Shipment .....	31
2.3.15	State .....	33
2.3.16	Substate .....	34
2.3.17	SupplierParty.....	34
2.3.18	TaxCategory.....	35

2.3.19	TaxScheme .....	35
2.3.20	TaxSubTotal .....	36
2.3.21	TaxTotal .....	36
2.4	Sitecore.Ecommerce.Core.OrderManagement.Extensions .....	37
2.4.1	CoreExtensions .....	37
2.5	Sitecore.Ecommerce.Merchant.Pipelines.HttpRequest .....	38
2.5.1	MerchantShopResolver .....	38
2.6	Sitecore.Ecommerce.Merchant.OrderManagement .....	39
2.6.1	AddNewOrderLineProcessingStrategy .....	39
2.6.2	CancelReservationProcessingStrategy .....	40
2.6.3	CaptureOrderProcessingStrategy .....	40
2.6.4	ChangeAllowanceChargeAmountProcessingStrategy .....	41
2.6.5	ChangeOrderLineQuantityProcessingStrategy .....	42
2.6.6	EditOrderLineProcessingStrategy .....	42
2.6.7	MerchantOrderCancellationStrategy .....	43
2.6.8	MerchantOrderProcessor .....	43
2.6.9	MerchantOrderManager .....	44
2.6.10	MerchantOrderStateConfiguration .....	44
2.6.11	OrderFieldProcessingStrategy .....	45
2.6.12	OrderLineFactoryImpl .....	46
2.6.13	OrderLineStateTransitionStrategy .....	46
2.6.14	OrderLineProcessingStrategy .....	47
2.6.15	OrderProcessingStrategyResolver .....	47
2.6.16	OrderProcessingStrategyResolverImpl .....	48
2.6.17	OrderStateProcessingStrategy .....	48
2.6.18	PackOrderProcessingStrategy .....	49
2.6.19	RemoveOrderLineProcessingStrategy .....	49
2.6.20	SendConfirmationProcessingStrategy .....	50
2.6.21	ShipOrderProcessingStrategy .....	50
2.6.22	StateValidator .....	51
2.7	Sitecore.Ecommerce.Merchant.OrderManagement.Extensions .....	52
2.7.1	MerchantExtensions .....	52
2.8	Sitecore.Ecommerce.Visitor.OrderManagement .....	53
2.8.1	VisitorOrderCancellationStrategy .....	53
2.8.2	VisitorOrderProcessor .....	53
2.8.3	VisitorOrderRepository .....	53
2.8.4	TransientOrderConverter .....	54
2.8.5	TransientOrderManager .....	55
2.9	Sitecore.Ecommerce.Visitor.Pipelines.HttpRequest .....	56
2.9.1	VisitorShopResolvingProcessor .....	56
2.10	Sitecore.Ecommerce.Visitor.Pipelines.OrderedCreated .....	57
2.10.1	NotifyCustomer .....	57
2.11	Sitecore.Ecommerce.Visitor.Sites .....	58
2.11.1	VisitorShopResolver .....	58
2.12	Sitecore.Ecommerce.Visitor.Unity .....	59
2.12.1	ShopContextContainerExtension .....	59

# Chapter 1

## Introduction

This guide describes the Sitecore Order Manager (OM) API and some useful extensions to its functionality.

It is useful for developers who are looking for information about the OM API. It describes the class functionality, parent classes, implementation, methods, and properties.

This manual contains the following chapters:

- **Chapter 1 — Introduction**  
This chapter is an introduction to the guide.
- **Chapter 2 — Order Manager Modules**  
This chapter contains the API reference.

## Chapter 2

# Order Manager Modules

This chapter describes the most frequently used classes in the Sitecore Order Manager API. Each section in this chapter presents an important module in the API. Each section contains a description of the classes in this module and tables that describe the properties and methods of the class.

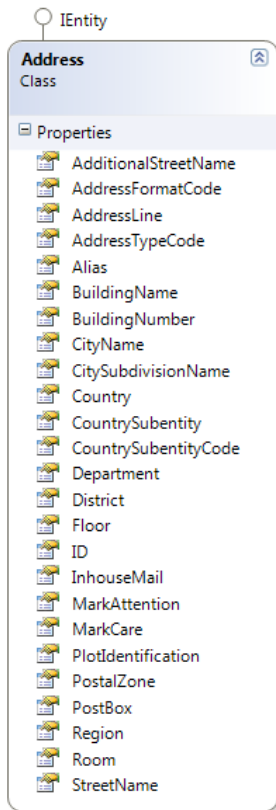
This chapter contains the following sections:

- Sitecore.Ecommerce.Core.Common
- Sitecore.Ecommerce.Core.OrderManagement
- Sitecore.Ecommerce.Core.OrderManagement.Orders
- Sitecore.Ecommerce.Core.OrderManagement.Extensions
- Sitecore.Ecommerce.Merchant.Pipelines.HttpRequest
- Sitecore.Ecommerce.Merchant.OrderManagement
- Sitecore.Ecommerce.Merchant.OrderManagement.Extensions
- Sitecore.Ecommerce.Visitor.OrderManagement
- Sitecore.Ecommerce.Visitor.Pipelines.HttpRequest
- Sitecore.Ecommerce.Visitor.Pipelines.OrderedCreated
- Sitecore.Ecommerce.Visitor.Sites
- Sitecore.Ecommerce.Visitor.Unity

## 2.1 Sitecore.Ecommerce.Core.Common

### 2.1.1 Address

This class implements the `IEntity` interface and contains information about the address.



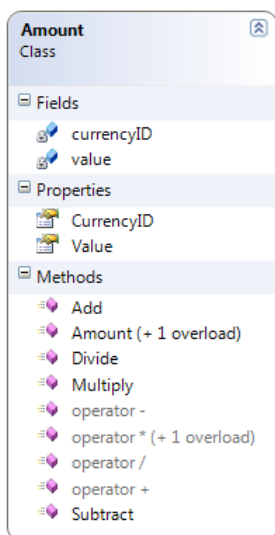
The properties of the `Address` class are:

Property	Description
<code>string</code> <code>AdditionalStreetName</code>	Gets or sets the additional text to further specify the street name.
<code>string</code> <code>AddressFormatCode</code>	Gets or sets the code of the address format.
<code>string</code> <code>AddressLine</code>	Gets or sets the address line.
<code>string</code> <code>AddressTypeCode</code>	Gets or sets the address type code to specify whether it is a home or a business address.
<code>long</code> <code>Alias</code>	Gets the address ID that is used in the database.
<code>string</code> <code>BuildingName</code>	Gets or sets the building name.
<code>string</code> <code>CityName</code>	Gets or sets the name of the city, town or village.
<code>string</code> <code>CitySubdivisionName</code>	Get or sets the city subdivision name. The subdivision can be the district or the borough.
<code>string</code> <code>Country</code>	Gets or sets the country.
<code>string</code> <code>CountrySubentity</code>	Gets or sets the country sub-entity. The sub-entity can be a county or a state.
<code>string</code> <code>CountrySubentityCode</code>	Gets or sets the country sub-entity code.
<code>string</code> <code>Department</code>	Gets or sets the department in the organization.
<code>string</code> <code>District</code>	Gets or sets the district.
<code>string</code> <code>Floor</code>	Gets or sets the floor of the building that the address is on.
<code>string</code> <code>ID</code>	Gets or sets the identifier of a specific address within a scheme of registered addresses.
<code>string</code> <code>InhouseMail</code>	Gets or sets the location of the in-house mail as part of the address.
<code>string</code> <code>MarkAttention</code>	Gets or sets the name of the person or the department in the organization to whom the incoming mail is marked, for example, <i>for the attention of</i> , <i>FAO</i> or <i>ATTN</i> for this address.

Property	Description
<code>string</code> MarkCare	Gets or sets the name of the person or the organization at this address to whom the incoming mail is marked, for example, <i>care of</i> or <i>C/O</i> .
<code>string</code> PlotIdentification	Text for the identifier of the plot where the address is located.
<code>string</code> PostalZone	Gets or sets the postal code.
<code>string</code> PostBox	Gets or sets the post box.
<code>string</code> Region	Gets or sets the region. The region can be a group of countries.
<code>string</code> Room	Gets or sets the room. The room can be a room, suite, or apartment in the building.
<code>string</code> StreetName	Gets or sets the street name.

## 2.1.2 Amount

This class contains information about the amount.



The properties of the `Amount` class:

Property	Description
<code>string</code> CurrencyID	Gets or sets the ID of the currency.
<code>decimal</code> Value	Gets or sets the value of the amount.

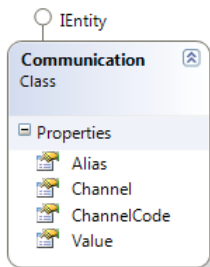
The methods of the `Amount` class are:

Method	Description
<code>Amount</code> Add( <code>Amount</code> amount1, <code>Amount</code> amount2)	Adds two amounts of the same currencyID.
<code>Amount</code> ()	Initializes a new instance of the <code>Amount</code> class.
<code>Amount</code> ( <code>decimal</code> value, <code>string</code> currencyId)	Takes the value and the currency of the amount and initializes a new instance of the <code>Amount</code> class.
<code>Amount</code> Divide( <code>Amount</code> amount, <code>decimal</code> divisor)	Divides the specified amount.
<code>Amount</code> operator -( <code>Amount</code> amount1, <code>Amount</code> amount2)	Implements the <code>-</code> operator by calling the <code>Subtract</code> method.
<code>Amount</code> operator *( <code>Amount</code> amount, <code>decimal</code> multiplier)	Implements the <code>*</code> operator by calling the <code>Multiply</code> method.

Method	Description
<code>Amount operator *(decimal multiplier, Amount amount)</code>	Implements the * operator by calling the Multiply method.
<code>Amount operator /(Amount amount, decimal divisor)</code>	Implements the / operator by calling the Divide method.
<code>Amount operator +(Amount amount1, Amount amount2)</code>	Implements the + operator by calling the Add method.
<code>Amount Subtract(Amount amount1, Amount amount2)</code>	Subtracts the value of amount2 from the value of amount1.

### 2.1.3 Communication

This class implements the `IEntity` interface and contains information about the means of communication.

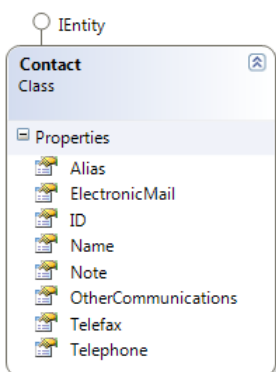


The properties of the `Communication` class:

Property	Description
<code>long Alias</code>	Gets the communication's ID that is used in the database.
<code>string Channel</code>	Gets or sets the communication type as text.
<code>string ChannelCode</code>	Gets or sets the code for the communication type.
<code>string Value</code>	Gets or sets the value of the communication such as the phone number or the email address.

### 2.1.4 Contact

This class implements the `IEntity` interface and contains information about the contact person or department in the organization.



The properties of the `Contact` class are:

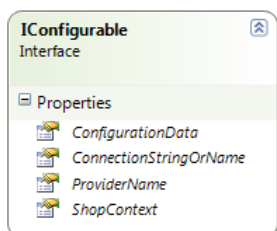
Property	Description
<code>long Alias</code>	Gets the contact's ID that is used in the database.
<code>string ElectronicMail</code>	Gets or sets the email of the contact.
<code>string ID</code>	Gets or sets the identifier of the contact. It is a personal reference number such as "7687687".
<code>string Name</code>	Gets or sets the name of the contact.
<code>string Note</code>	Gets or sets the free text note for the contact. This note can describe the circumstances in which the contact can be used such as emergency.
<code>ICollection&lt;Communication&gt; OtherCommunications</code>	Gets or sets the information about other communication means.



Property	Description
<code>string</code> Telefax	Gets or sets the fax number of the contact.
<code>string</code> Telephone	Gets or sets the telephone number of the contact.

## 2.1.5 IConfigurable

The implementations of this interface contain information about the configuration settings that are needed to refer to the order's source.

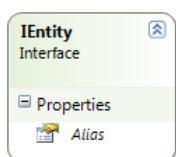


The properties of the `IConfigurable` interface are:

Property	Description
<code>Dictionary&lt;string, object&gt;</code> ConfigurationData	Gets or sets the configuration data.
<code>string</code> ConnectionStringOrName	Gets or sets the connection string or name.
<code>string</code> ProviderName	Gets or sets the name of the data provider. This name is used in the database connections to identify the database system that is used – for example, Microsoft SQL Server, Oracle, and PostgreSQL Direct.
<code>ShopContext</code> ShopContext	Gets or sets of the shop context.

## 2.1.6 IEntity

This interface acts as a contract for the classes that have the `Alias` property.

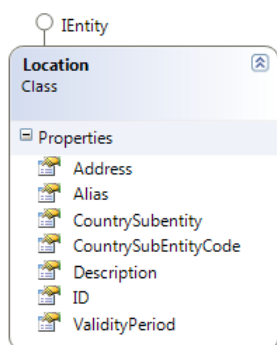


The only property of the `IEntity` interface is:

Property	Description
<code>long</code> Alias	Gets the <code>IEntity</code> 's ID that is used in the database.

## 2.1.7 Location

This class implements the `IEntity` interface and contains information about the location.



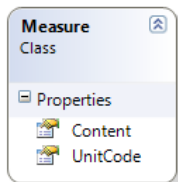
The properties of the `Location` class are:

Property	Description
<code>Address</code> Address	Gets or sets information about the address of the location.
<code>long</code> Alias	Gets the location's ID that is used in the database.
<code>string</code> CountrySubEntity	Gets or sets text about the country sub-entity. A sub-entity can be a county or a state.

Property	Description
<code>string</code> CountrySubEntityCode	Gets or sets the code of the sub-entity.
<code>string</code> Description	Gets or sets text of the description of the location.
<code>string</code> ID	Gets or sets the identifier of the location such as the European Article Number (EAN) and the Global Location Number (GLN).
<code>Period</code> ValidityPeriod	Gets or sets information about the validity period of the location. This is the period in which the location can be used, for example for delivery.

### 2.1.8 Measure

This class contains information about the measure.

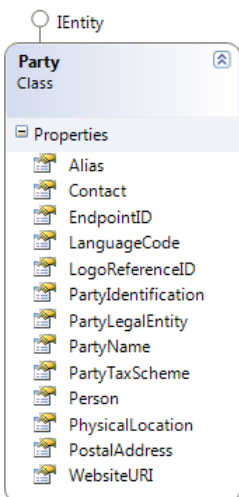


The properties of the `Measure` class are:

Property	Description
<code>decimal</code> Content	Gets or sets the decimal value such as 10000.25  <b>Note</b> You must not use more than four decimals for the value of the <code>Content</code> property.
<code>string</code> UnitCode	Gets or sets the unit code such as "EA".

### 2.1.9 Party

This class implements the `IEntity` interface and contains information about the organization, the sub-organization, or the individual that has a role in the business process.



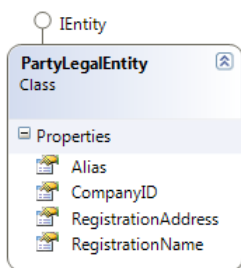
The properties of the `Party` class are:

Property	Description
<code>long</code> Alias	Gets the party's ID that is used in the database.
<code>Contact</code> Contact	Gets or sets the contact of the party.
<code>string</code> EndpointID	Gets or sets the end point for the routing service such as the EAN and the GLN.
<code>string</code> LanguageCode	Gets or sets the code of the party language.
<code>string</code> LogoReferenceID	Gets or sets the identifier of the logo reference.
<code>string</code> PartyIdentification	Gets or sets an identifier of the party.
<code>PartyLegalEntity</code> PartyLegalEntity	Gets or sets the legal entity of the party.

Property	Description
<code>string</code> PartyName	Gets or sets the name of the party.  <b>Note</b> You must use this name when no other party identification is used.
<code>ICollection&lt;PartyTaxScheme&gt;</code> PartyTaxScheme	Gets or sets the information about the tax scheme of the party.
<code>Person</code> Person	Gets or sets the persons of the party.
<code>Location</code> PhysicalLocation	Gets or sets the location of the party.  <b>Note</b> You must use <code>PhysicalLocation</code> if it is different from <code>PostalAddress</code> .
<code>Address</code> PostalAddress	Gets or sets the postal address of the party.
<code>string</code> WebsiteURI	Gets or sets the URL of the party website.

### 2.1.10 PartyLegalEntity

This class implements the `IEntity` interface and contains information about the legal registration that is applicable for the party.

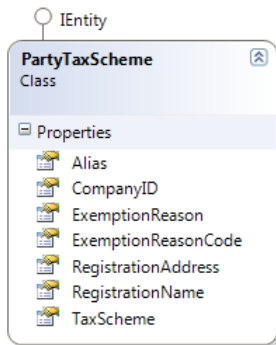


The properties of the `PartyLegalEntity` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the party legal entity that is used in the database.
<code>string</code> CompanyID	Gets or sets the identifier of the legal entity. It identifies the company as registered with the company registration scheme.
<code>Address</code> RegistrationAddress	Gets or sets the registration address of the legal entity. It is the registered address of the party in the company registration scheme.
<code>string</code> RegistrationName	Gets or sets the registration name of the legal entity. It is the name of the party that is registered by the legal authority.

## 2.1.11 PartyTaxScheme

This class implements the `IEntity` interface and contains information about the tax scheme of the party.

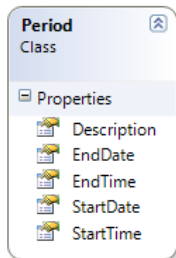


The properties of the `PartyTaxScheme` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the party tax scheme that is used in the database.
<code>string</code> CompanyID	Gets or sets the identifier of the company. This identifier is assigned for tax purposes to the party by the taxation authority.
<code>string</code> ExemptionReason	Gets or sets the reason for the exemption from the tax.
<code>string</code> ExemptionReasonCode	Gets or sets the code of tax exemption.
<code>Address</code> RegistrationAddress	Gets or sets the registered address of the party. It is used for tax purposes.
<code>string</code> RegistrationName	Gets or sets the name of the party. It is the official name of the party as registered by the relevant taxation authority.
<code>TaxScheme</code> TaxScheme	Gets or sets the information about the tax scheme.

## 2.1.12 Period

This class contains information about the period of time.

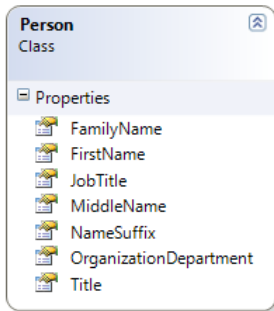


The properties of the `Period` class are:

Property	Description
<code>string</code> Description	Gets or sets the description for the period.
<code>DateTime</code> EndDate	Gets or sets the end date of the period.
<code>TimeSpan</code> EndTime	Gets or sets the end time of the period.
<code>DateTime</code> StartDate	Gets or sets the start date of the period.
<code>TimeSpan</code> StartTime	Gets or sets the start time of the period.

## 2.1.13 Person

This class contains information about the person.

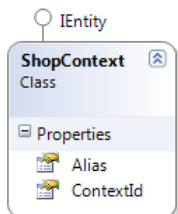


The properties of the `Person` class are:

Property	Description
<code>string</code> FamilyName	Gets or sets the surname or the family name of the person.
<code>string</code> FirstName	Gets or sets the first name of the person.
<code>string</code> JobTitle	Gets or sets the job title of the person.
<code>string</code> MiddleName	Gets or sets the middle name of the person.
<code>string</code> NameSuffix	Gets or sets the name suffix of the person., such as "PhD", and "Jnr".
<code>string</code> OrganizationDepartment	Gets or sets the department that this person works for.
<code>string</code> Title	Gets or sets the title of the person such as "Mr.", "Ms.", "Dr.", and "Sir".

## 2.1.14 ShopContext

This class implements the `IEntity` interface and contains information about the context of the webshop.



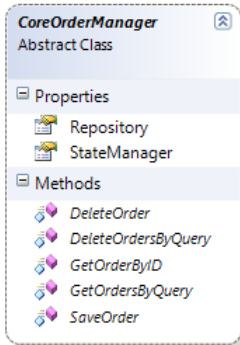
The properties the `ShopContext` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the shop context that is used in the database.
<code>string</code> ContextId	Gets or sets the ID of the shop context. It is a unique name that refers to the webshop. This is typically the hostname of the webshop or the ID of the website definition that contains the webshop.

## 2.2 Sitecore.Ecommerce.Core.OrderManagement

### 2.2.1 CoreOrderManager

This class contains information about the core order manager. It is an abstract class in which all the methods are protected and internal because they are just defining an interface for the implementations.

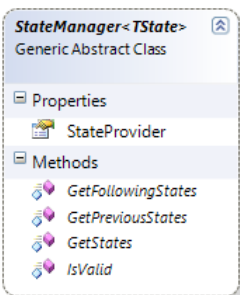


The properties of the `CoreOrderManager` class are:

Property	Description
<code>Repository&lt;Order&gt;</code> Repository	Gets or sets the repository.
<code>StateManager&lt;State&gt;</code> StateManager	Gets or sets the state manager.

### 2.2.2 StateManager

This is a generic abstract class and contains information about the state manager. It is an abstract class in which all the methods are protected and internal because they are defining an interface for the implementations.

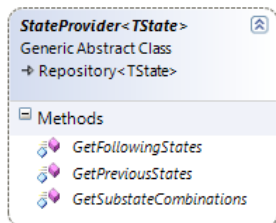


The only property of the `StateManager` class is:

Property	Description
<code>StateProvider&lt;TState&gt;</code> StateProvider	Gets or sets the state provider.

### 2.2.3 StateProvider

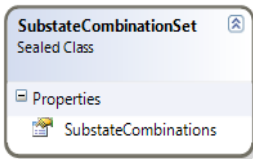
This is a generic abstract class that implements the `Repository` class. All the methods in this class are protected and internal because they are defining an interface for the implementations.



These methods are used to get and navigate between the different states and substates of the order.

## 2.2.4 SubstateCombinationSet

This is a sealed class that contains information about the set of substate combinations.

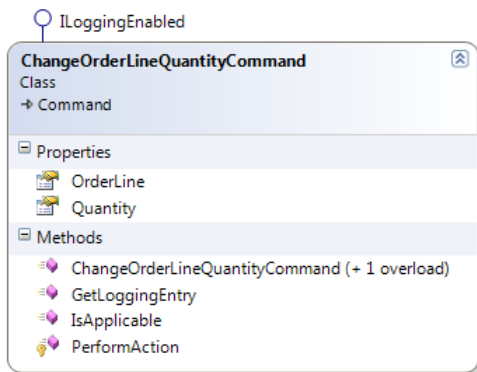


The only property of the `SubstateCombinationSet` class is:

Property	Description
<code>IEnumerable&lt;IDictionary&lt;string, bool&gt;&gt; SubstateCombinations</code>	Gets or sets the sub-state combinations.

## 2.2.5 ChangeOrderLineQuantityCommand

This class implements the `ILoggingEnabled` interface, the `Command` class and contains information about the quantity of the order line and allows you to change it.



The properties of the `ChangeOrderLineQuantityCommand` class are:

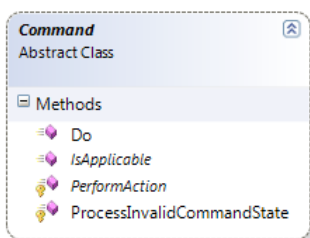
Property	Description
<code>OrderLine</code> <code>OrderLine</code>	Gets or sets the order line.
<code>decimal</code> <code>Quantity</code>	Gets or sets the quantity. It is the value that is changed when changing the quantity of products on that particular order line in the user interface.

The methods of the `ChangeOrderLineQuantityCommand` class are:

Method	Description
<code>ChangeOrderLineQuantityCommand(OrderLine orderLine)</code>	Takes the order line and initializes a new instance of the <code>ChangeOrderLineQuantityCommand</code> class.
<code>ChangeOrderLineQuantityCommand()</code>	Initializes a new instance of the <code>ChangeOrderLineQuantityCommand</code> class.
<code>ILoggingEntry</code> <code>GetLoggingEntry()</code>	Gets the logging entry.
<code>bool</code> <code>IsApplicable()</code>	Specifies whether the command is applicable to the specified order line. Its value is <code>true</code> if the command is applicable.

## 2.2.6 Command

This is an abstract class that the implementations use to process a command.

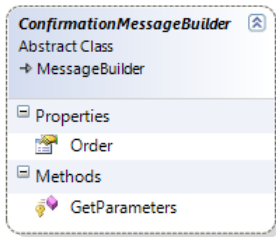


The methods of the `Command` class are:

Method	Description
<code>void</code> <code>Do()</code>	Processes the order line.
<code>bool</code> <code>IsApplicable()</code>	Specifies whether the command is applicable to the specified order line. It is <code>true</code> if the command is applicable.

## 2.2.7 ConfirmationMessageBuilder

This is an abstract class that implements the `MessageBuilder` class and defines the confirmation message builder.

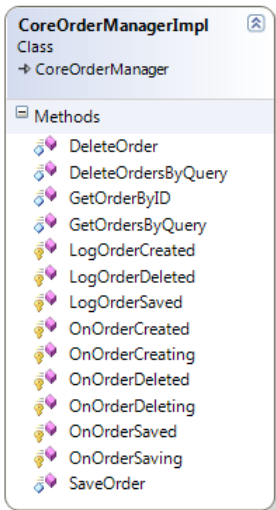


The only property of the `ConfirmationMessageBuilder` class is:

Property	Description
<code>virtual Order</code> Order	Gets or sets the order.

## 2.2.8 CoreOrderMangerImpl

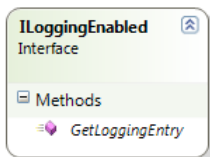
The class implements the `CoreOrderManager` class. It is used to manipulate the order.



All the methods of this class are protected to hide the core API and expose Visitor and Merchant only.

## 2.2.9 ILoggingEnabled

This interface specifies whether the type supports logging or not.



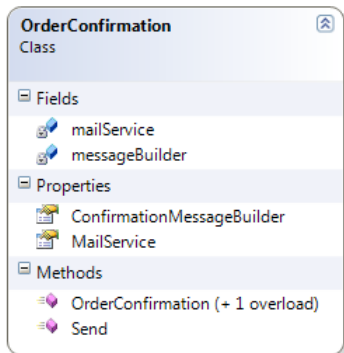
The only method in the `ILoggingEnabled` interface is:

Method	Description
<code>LoggingEntry</code> <code>GetLoggingEntry()</code>	Gets the logging entry.



## 2.2.10 OrderConfirmation

This class contains information about the confirmation of an order.



The properties of the `OrderConfirmation` class are:

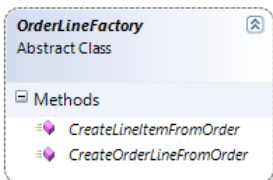
Property	Description
<code>ConfirmationMessageBuilder</code> <code>ConfirmationMessageBuilder</code>	Gets or sets the confirmation message builder.
<code>MailService</code> <code>MailService</code>	Gets or sets the mail service.

The methods of the `OrderConfirmation` class are:

Method	Description
<code>OrderConfirmation()</code>	Initializes a new instance of the <code>OrderConfirmation</code> class.
<code>OrderConfirmation(MailService mailService, ConfirmationMessageBuilder messageBuilder)</code>	Takes the mail service, the message builder, and initializes a new instance of the <code>OrderConfirmation</code> class.
<code>void Send()</code>	Sends the confirmation message.

## 2.2.11 OrderLineFactory

This is an abstract class. You can implement it to create an order line or a line item from the order.

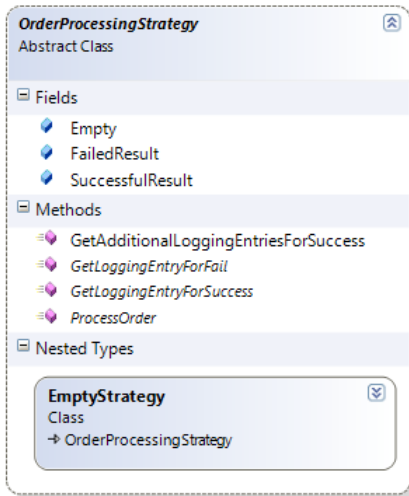


The methods of the `OrderLineFactory` class are:

Method	Description
<code>LineItem CreateLineItemFromOrder(Order order, string productCode, long quantity)</code>	Takes the order, the product code, the quantity, and creates the line item.
<code>OrderLine CreateOrderLineFromOrder(Order order, string productCode, long quantity)</code>	Takes the order, the product code, the quantity, and creates the order line.

## 2.2.12 OrderProcessingStrategy

This is an abstract class. You can implement it to get the different types of log entries.

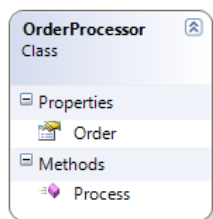


The methods of the `OrderProcessingStrategy` class are:

Method	Description
<pre>IList&lt;LogEntry&gt; GetAdditionalLogging EntriesForSuccess (     Order order)</pre>	Takes the order and provides the additional information for the logging engine about the performed action. For example, if you capture an order, the main action is <i>Capture Order</i> . At the same time the substate of the order is also changed.
<pre>LogEntry GetLoggingEntryForFail (     Order order,     params object[] parameters)</pre>	Takes the order, the parameters, and performs additional failure processing.
<pre>LogEntry GetLoggingEntryForSuccess (     Order order)</pre>	Takes the order and gets the logged message of the successful operation.
<pre>string ProcessOrder (     Order order,     IDictionary&lt;string,     object&gt; parameters)</pre>	Takes the order, the parameters, and processes the order.

## 2.2.13 OrderProcessor

This class is used to process an order.



The only property in the `OrderProcessor` class is:

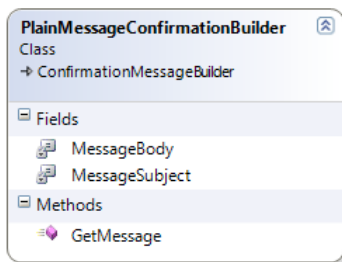
Property	Description
<code>Order</code> Order	Gets or sets the order.

The only method in the `OrderProcessor` class is:

Method	Description
<pre>void Process (     ProcessingStrategy processingStrategy)</pre>	Takes the processing strategy and processes the order.

## 2.2.14 PlainMessageConfirmationBuilder

This class implements the `ConfirmationMessageBuilder` class. It is used to get the message.

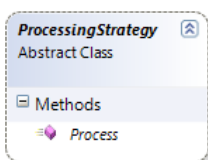


The only method in the `PlainMessageConfirmationBuilder` class is:

Method	Description
<code>MailMessage</code> <code>SendMessage (string templateName)</code>	Takes the template name and gets the mail message.

## 2.2.15 ProcessingStrategy

This is an abstract class that you can implement to process an order.

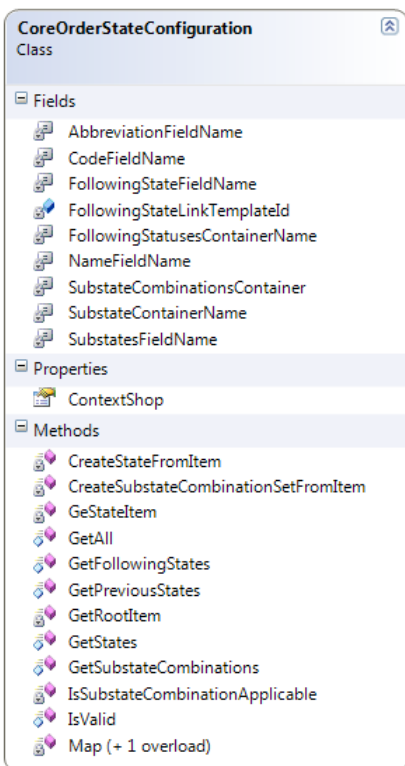


The only method in the `ProcessingStrategy` class is:

Method	Description
<code>void</code> <code>Process (Order order)</code>	Takes the order and runs the processor.

## 2.2.16 CoreOrderStateConfiguration

This class reads the order state configuration from the webshop business catalog.



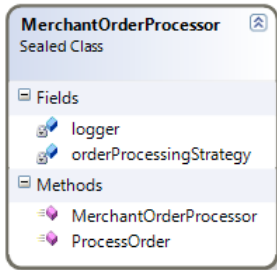
The only property of the `CoreOrderStateConfiguration` class is:

Property	Description
<code>ShopContext</code> <code>ContextShop</code>	Gets or sets the shop context.

The methods of this class are either protected or private because they are only used internally. However, you can use the methods in the corresponding class: `MerchantOrderStateConfiguration` instead of this class.

## 2.2.17 MerchantOrderProcessor

This is a sealed class that defines the merchant order processor.



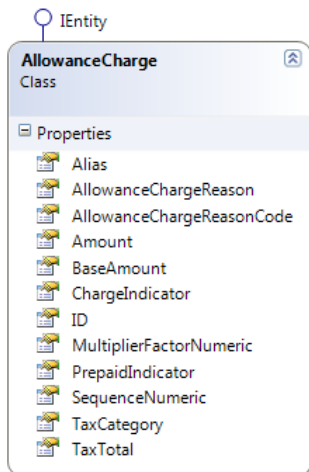
The methods of the `MerchantOrderProcessor` class are:

Method	Description
<pre>MerchantOrderProcessor (     OrderProcessingStrategy     orderProcessingStrategy)</pre>	<p>Takes the order processing strategy and initializes a new instance of the <code>MerchantOrderProcessor</code> class.</p>
<pre>string ProcessOrder (     Order order,     IDictionary&lt;string,     object&gt;     parameters)</pre>	<p>Takes the order, the parameters, and processes the order.</p>

## 2.3 Sitecore.Ecommerce.Core.OrderManagement.Orders

### 2.3.1 AllowanceCharge

This class implements the `IEntity` interface and contains the information about the charge or the discount.

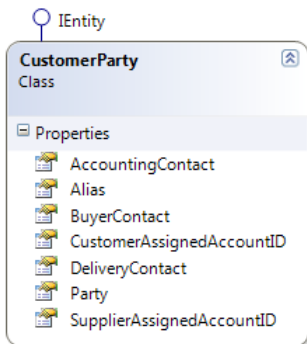


The properties of the `AllowanceCharge` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the allowance charge that is used in the database.
<code>string</code> AllowanceChargeReason	Gets or sets the reason for the allowance charge.
<code>string</code> AllowanceChargeReasonCode	Gets or set sets the code that represents the reason for the allowance charge.
<code>Amount</code> Amount	Gets or sets the amount of the allowance charge.
<code>Amount</code> BaseAmount	Gets or sets the amount that is used for the <code>MultiplierFactorNumeric</code> when calculating the discount.
<code>bool</code> ChargeIndicator	Gets or sets the indicator that classifies the charge as either fees or a discount.
<code>string</code> ID	Gets or sets the allowance charge identifier that is used in a third party database.
<code>decimal</code> MultiplierFactorNumeric	Gets or sets the factor that is applied to the base amount to calculate the allowance charge.
<code>bool</code> PrepaidIndicator	Gets or sets the indicator that classifies the allowance charge as prepaid or not. It is set to <code>true</code> if it is prepaid.
<code>decimal</code> SequenceNumeric	Gets or sets the numerical order sequence in which the allowance charge is calculated when multiple allowance charges apply. If all allowance charges apply to the same base amount, <code>SequenceNumeric</code> is <code>'1'</code> for all allowance charges.
<code>TaxCategory</code> TaxCategory	Gets or sets the information about the tax category
<code>TaxTotal</code> TaxTotal	Gets or sets the total tax value of the allowance charge.

## 2.3.2 CustomerParty

This class implements the `IEntity` interface and contains information about the customer party. The customer party is either the buyer or the entity that is paying.

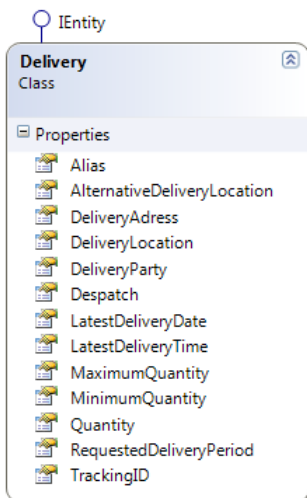


The properties of the `CustomerParty` class are:

Property	Description
<code>Contact</code> AccountingContact	Gets or sets the information about the accounting contact of the buyer.
<code>long</code> Alias	Gets the ID of the customer party that is used in the database.
<code>Contact</code> BuyerContact	Gets or sets the information about the buyer.
<code>string</code> CustomerAssignedAccountID	Gets or sets the ID of the account given by the customer.
<code>Contact</code> DeliveryContact	Gets or sets the information about the delivery contact.
<code>Party</code> Party	Gets or sets the information about the party.
<code>string</code> SupplierAssignedAccountID	Gets or sets the ID of the customer account given by the supplier.

## 2.3.3 Delivery

This class implements the `IEntity` interface and contains information about the delivery of the order.



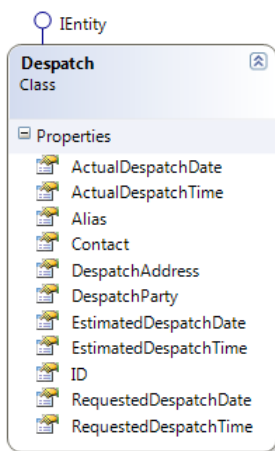
The properties of the `Delivery` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the delivery that is used in the database.
<code>Location</code> AlternativeDeliveryLocation	Gets or sets the alternative delivery location.
<code>Address</code> DeliveryAddress	Gets or sets the information about the delivery address.
<code>Location</code> DeliveryLocation	Gets or sets the information about the delivery location.
<code>Party</code> DeliveryParty	Gets or set the information about the delivery receiver.
<code>Despatch</code> Despatch	Gets or sets the information about the sender of the delivery.
<code>DateTime</code> LatestDeliveryDate	Gets or sets the latest date for the delivery that the buyer determines.
<code>TimeSpan</code> LatestDeliveryTime	Gets or sets the latest times for the delivery that the buyer determines.
<code>decimal</code> MaximumQuantity	Gets or sets the maximum quantity for the delivery.

Property	Description
<code>decimal</code> MinimumQuantity	Gets or sets the minimum quantity for the delivery.
<code>decimal</code> Quantity	Gets or sets the quantity for the delivery.
<code>Period</code> RequestedDeliveryPeriod	Gets or sets the delivery period that the buyer requests.
<code>string</code> TrackingID	Gets or sets the tracking ID of the delivery.

## 2.3.4 Despatch

This class implements the `IEntity` interface and contains information about the delivery dispatch.

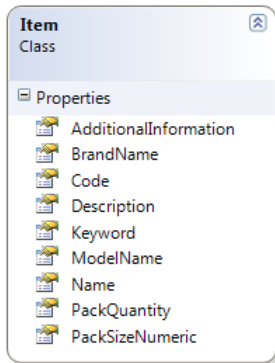


The properties of the `Despatch` class are:

Property	Description
<code>DateTime</code> ActualDespatchDate	Gets or sets the actual dispatch date.
<code>TimeSpan</code> ActualDespatchTime	Gets or sets the actual dispatch time.
<code>long</code> Alias	Gets the ID of the dispatch that is used in the database.
<code>Contact</code> Contact	Gets or sets the information about the contact.
<code>Address</code> DespatchAddress	Gets or sets the address of the dispatch.
<code>Party</code> DespatchParty	Gets or sets the party that dispatches the delivery.
<code>DateTime</code> EstimatedDespatchDate	Gets or sets the dispatch date that is estimated by the seller.
<code>DateTime</code> EstimatedDespatchTime	Gets or sets the dispatch time that is estimated by the seller.
<code>string</code> ID	Gets or sets the identifier of the delivery.
<code>DateTime</code> RequestedDespatchDate	Gets or sets the latest dispatch date that the buyer requests.
<code>TimeSpan</code> RequestedDespatchTime	Gets or sets the latest dispatch time that the buyer requests.

### 2.3.5 Item

This class contains information about the item.

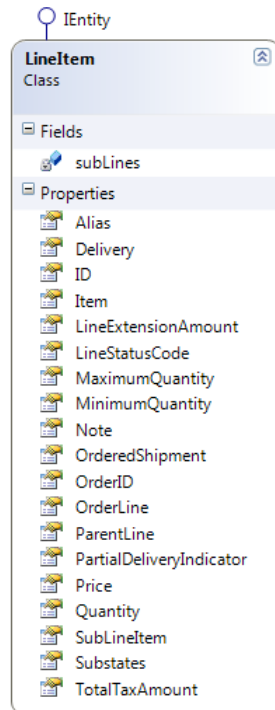


The properties of the `Item` class are:

Property	Description
<code>string</code> AdditionalInformation	Gets or sets the additional information about the item, such as the URL of the relevant web page.
<code>string</code> BrandName	Gets or sets the brand name of the item.
<code>string</code> Code	Gets or sets the code of the item.
<code>string</code> Description	Gets or sets the description of the item.
<code>string</code> Keyword	Gets or sets the keyword for the product search.
<code>string</code> ModelName	Gets or sets the model name of the item.
<code>string</code> Name	Gets or sets the name of the item.
<code>decimal</code> PackQuantity	Gets or sets the number of the packages.
<code>decimal</code> PackSizeNumeric	Gets or sets the number of items in the package.

### 2.3.6 LineItem

This class implements the `IEntity` interface and contains information about the order line item.



The properties of the `LineItem` class are:

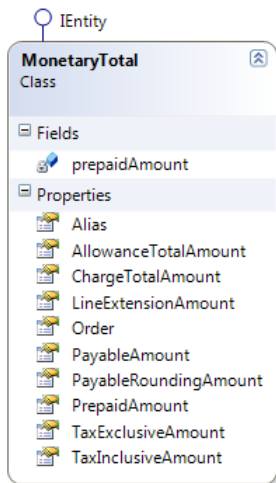
Property	Description
<code>long</code> Alias	Gets the ID of the order line item that is used in the database.
<code>ICollection&lt;Delivery&gt;</code> Delivery	Gets or sets the information about the request for delivery.
<code>string</code> ID	Gets or sets the address identifier that is used in a third party database.
<code>Item</code> Item	Gets or sets the information about the item on the order line.
<code>Amount</code> LineExtensionAmount	Gets or sets the total amount for line item including the sum of all order lines, the allowance, the charges, and excluding the taxes.



Property	Description
<code>string</code> LineStatusCode	Gets or sets the status of the line whether it is: <ul style="list-style-type: none"> <li>• A new line</li> <li>• A change in the line</li> <li>• A deletion of the line.</li> </ul>
<code>decimal</code> MaximumQuantity	Gets or sets the maximum quantity of products that you can order.
<code>decimal</code> MinimumQuantity	Gets or sets the minimum quantity for ordering.
<code>string</code> Note	Gets or sets the free text field. <p><b>Note</b> This element may contain notes or any information that does not exist in other structures or classes.</p>
<code>OrderedShipment</code> OrderedShipment	Gets or sets the information about the order shipment.
<code>string</code> OrderID	Gets or sets the merchant order ID.
<code>OrderLine</code> OrderLine	Gets or sets the order line.
<code>LineItem</code> ParentLine	Gets or sets the parent line.
<code>bool</code> PartialDeliveryIndicator	Gets or sets the value whether the partial delivery allowed.
<code>Price</code> Price	Gets or sets the price of the item.
<code>decimal</code> Quantity	Gets or sets the quantity of the item.
<code>ICollection&lt;LineItem&gt;</code> SubLineItem	Gets the sub-line item.
<code>ICollection&lt;Substate&gt;</code> Substates	Gets or sets the sub-states.
<code>Amount</code> TotalTaxAmount	Gets or sets the tax total amount for the order line.

### 2.3.7 MonetaryTotal

This class implements the `IEntity` interface and contains information about the different monetary amounts.



The properties of the `MonetaryTotal` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the monetary total that is used in the database.
<code>Amount</code> AllowanceTotalAmount	Gets the total amount of all the allowances.
<code>Amount</code> ChargeTotalAmount	Gets the total amount of all the charges.
<code>Amount</code> LineExtensionAmount	Gets the line extension amount without the tax and settlement discounts, but including any applicable rounding amount.
<code>Order</code> Order	Gets or sets the order.
<code>Amount</code> PayableAmount	Gets the amount to be paid.
<code>Amount</code> PayableRoundingAmount	Gets the payable rounding amount, whether positive or negative, that is added to the calculated line extension amount to get the rounded line extension amount
<code>Amount</code> PrepaidAmount	Gets or sets the total prepaid amount.
<code>Amount</code> TaxExclusiveAmount	Gets the tax without amount.
<code>Amount</code> TaxInclusiveAmount	Gets the tax including amount.

### 2.3.8 Order

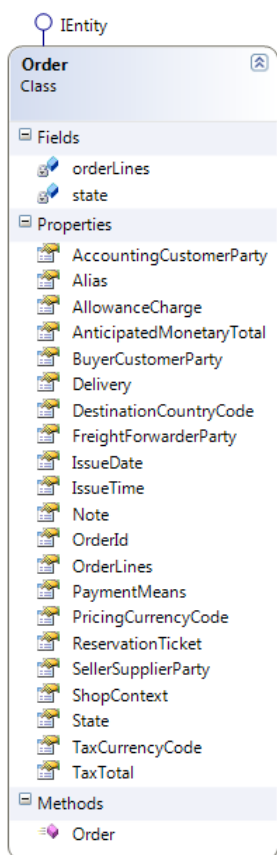
The `Order` class is used to exchange the electronic orders and to process the coherent order-to-invoice flow.

The order is sent from the `BuyerCustomerParty` object to the `SellerSupplierParty` object. It is structured to fulfill the formal requirements for a valid order change and for use in Denmark, such as tax number, time of delivery, and specifications.

Apart from the formal demands on the sender/receiver and the order content, the electronic version must contain the *End Point ID*, a *Person reference* and usually an order number or request number. It must also fulfill the requirements in this document guideline.

- The *End Point ID* is used to identify the *electronic mailbox* to which the electronic document is delivered and also provides the basis for routing the message.
- The *Person Reference* is used to identify the person who is responsible for placing the order.
- The *order number* or *requisition number* is used to identify the order.

These three types of information are used subsequently when the electronic invoice is issued with reference to the previous order or requisition. Depending on the profile that is used, the order can take part of an advanced order flow to allow the exchange of order changes, order deletions and order responses.



The properties of the Order class are:

Property	Description
<code>CustomerParty</code> <code>AccountingCustomerParty</code>	Gets or sets the information about the accounting customer party. This is the party to which the invoice is sent.
<code>long</code> <code>Alias</code>	Gets the order's ID that is used in the database.
<code>ICollection&lt;AllowanceCharge&gt;</code> <code>AllowanceCharge</code>	Gets or sets the information about the allowances and charges that apply to the order as a whole.
<code>MonetaryTotal</code> <code>AnticipatedMonetaryTotal</code>	Gets or sets the net monetary total, excluding allowances and taxes, as expected by the buyer.
<code>CustomerParty</code> <code>BuyerCustomerParty</code>	Gets or sets the information about the buyer.
<code>ICollection&lt;Delivery&gt;</code> <code>Delivery</code>	Gets or sets the information about delivery.
<code>string</code> <code>DestinationCountryCode</code>	Gets or sets the destination country code.  <b>Note</b> This code is used for customs purposes.
<code>ICollection&lt;Party&gt;</code> <code>FreightForwarderParty</code>	Gets or sets the information about the freight carrier.
<code>DateTime</code> <code>IssueDate</code>	Gets or sets the date on which the order is issued.
<code>TimeSpan</code> <code>IssueTime</code>	Gets or sets the time at which the order is issued.
<code>string</code> <code>Note</code>	Gets or sets the note on the order.

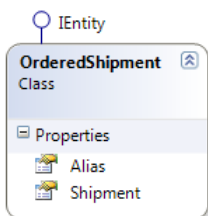
Property	Description
<code>string</code> OrderId	Gets or sets the order identifier that the buyer specifies.
<code>ICollection&lt;OrderLine&gt;</code> OrderLines	Gets or sets one or more order lines.
<code>PaymentMeans</code> PaymentMeans	Gets or sets the information about the payment type.
<code>string</code> PricingCurrencyCode	Gets or sets the currency code for the price.
<code>ReservationTicket</code> ReservationTicket	Gets or sets the reservation ticket.
<code>SupplierParty</code> SellerSupplierParty	Gets or sets the information about the seller.
<code>ShopContext</code> ShopContext	Gets or sets the shop context.
<code>State</code> State	Gets or sets the state of the order.
<code>string</code> TaxCurrencyCode	Gets or sets the code of the currency that is requested for the tax amount in the order invoices.
<code>TaxTotal</code> TaxTotal	Gets or sets the information about the total tax value of the order as calculated by the buyer.

The only method in the `order` class is:

Method	Description
<code>Order()</code>	Initializes a new instance of the <code>Order</code> class.

### 2.3.9 OrderedShipment

This class implements the `IEntity` interface and contains information about the order shipment.

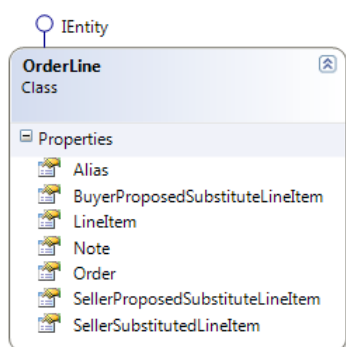


The properties of the `OrderedShipment` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the ordered shipment that is used in the database.
<code>Shipment</code> Shipment	Gets or sets the shipment information.

## 2.3.10 OrderLine

This class implements the `IEntity` interface and contains information about the order line.

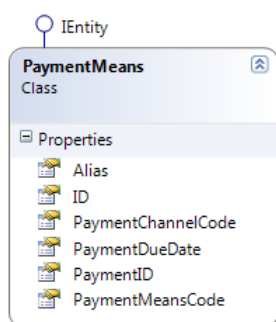


The properties of the `OrderLine` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the order line that is used in the database.
<code>ICollection&lt;LineItem&gt;</code> BuyerProposedSubstituteLineItem	Gets or sets the possible alternative items that the buyer proposes.
<code>LineItem</code> LineItem	Gets or sets the order line item.
<code>string</code> Note	Gets or sets the content of the free text note of the order line.  <b>Note</b> This property can contain notes that are not explicitly stated in any other class.
<code>Order</code> Order	Gets or sets the order.
<code>ICollection&lt;LineItem&gt;</code> SellerProposedSubstituteLineItem	Gets or sets the possible alternative items that the seller proposes.
<code>ICollection&lt;LineItem&gt;</code> SellerSubstitutedLineItem	Gets or sets the actual alternative items that the seller gives.

## 2.3.11 PaymentMeans

This class implements the `IEntity` interface and contains information about the order payment means.



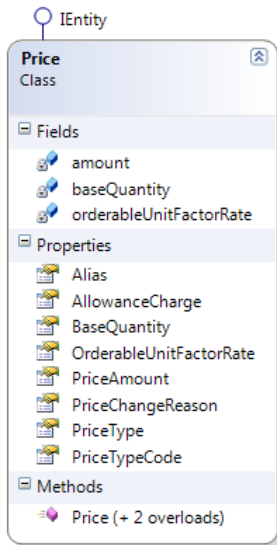
The properties of the `PaymentMeans` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the payment means that is used in the database.
<code>string</code> ID	Gets or sets the payment means identifier that is used in a third party database.
<code>string</code> PaymentChannelCode	Gets or sets the payment channel code.
<code>DateTime</code> PaymentDueDate	Gets or sets the date on which the payment is due.
<code>string</code> PaymentID	Gets or sets the identifier of the payment.

Property	Description
<code>string</code> PaymentMeansCode	Gets or sets the identifier of the payment means.

### 2.3.12 Price

This class implements the `IEntity` interface and contains information about the price of the order.



The properties of the `Price` class are:

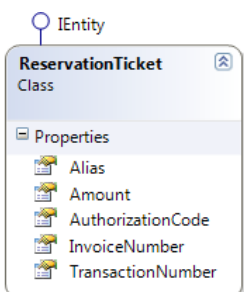
Property	Description
<code>long</code> Alias	Gets the ID of the price that is used in the database.
<code>IEnumerable&lt;AllowanceCharge&gt;</code> AllowanceCharge	Gets or sets the allowance charge.
<code>decimal</code> baseQuantity	Gets or sets the base quantity. It is the actual quantity to which the Price applies
<code>Amount</code> PriceAmount	Gets or sets the price value.
<code>string</code> PriceChangeReason	Gets or sets the price change reason.
<code>string</code> PriceType	Gets or sets the price type.
<code>string</code> PriceTypeCode	Gets or sets the price type code.

The methods in the `Price` class are:

Method	Description
<code>Price()</code>	Initializes a new instance of the <code>Price</code> class.
<code>Price(Amount amount)</code>	Takes the price value and initializes a new instance of the <code>Price</code> class.
<code>Price(Amount amount, decimal baseQuantity)</code>	Takes the price value, the base quantity and initializes a new instance of the <code>Price</code> class.

### 2.3.13 ReservationTicket

This class implements the `IEntity` interface and contains information about the reservation ticket.



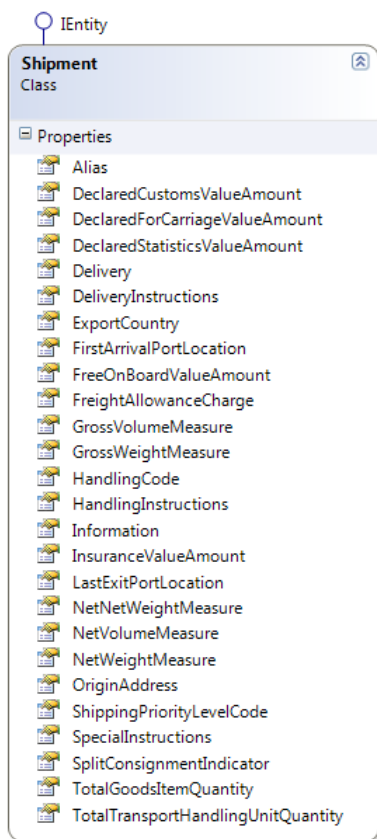
The properties of the `ReservationTicket` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the reservation ticket that is used in the database.
<code>decimal</code> Amount	Gets or sets the total order amount reserved on the credit card for a later withdrawal when the order has been shipped.

Property	Description
<code>string</code> AuthorizationCode	Gets or sets the authorization code. This code is an arbitrary value that is produced and received from the payment provider. It must be used later when receiving the money from the same payment provider. It identifies the reservation of money.
<code>string</code> InvoiceNumber	Gets or sets the invoice number.
<code>string</code> TransactionNumber	Gets or sets the transaction number.

### 2.3.14 Shipment

This class implements the `IEntity` interface and contains information about the shipment of an order.



The properties of the `Shipment` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the shipment that is used in the database.
<code>Amount</code> DeclaredCustomsValueAmount	Gets or sets the amount of the declared customs value. It is used for the goods in a shipment which are subject to the same customs procedure, and have the same tariff/statistical heading, country information and duty regime.
<code>Amount</code> DeclaredForCarriageValueAmount	Gets or sets the amount of the declared for carriage value.
<code>Amount</code> DeclaredStatisticsValueAmount	Gets or sets the amount of the declared statistics value.
<code>Delivery</code> Delivery	Gets or sets the delivery information of the shipment.
<code>string</code> DeliveryInstructions	Gets or sets the delivery instructions.
<code>string</code> ExportCountry	Gets or sets the country from which the goods are originally exported.
<code>Location</code> FirstArrivalPortLocation	Gets or sets the first arrival location.  <b>Note</b> This location can be a port, an airport, or a border crossing.

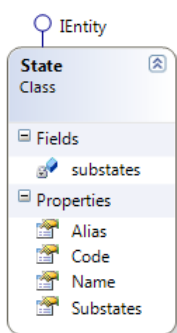
Property	Description
<b>Amount</b> FreeOnBoardValueAmount	Gets or sets the monetary amount that is paid as calculated under the applicable trade delivery.
<b>AllowanceCharge</b> FreightAllowanceCharge	Gets or sets the transportation cost that is added by the shipper under the terms of the carriage contract.  <b>Note</b> In addition to transportation costs, this may include items such as packing, documentation, loading, unloading, and the insurance.
<b>Measure</b> GrossVolumeMeasure	Gets or sets the total volume of the goods including the packaging in the shipment.
<b>Measure</b> GrossWeightMeasure	Gets or sets the weight of the goods without packaging and without the transport equipment.
<b>string</b> HandlingCode	Gets or sets the handling code that is required for the shipment.
<b>string</b> HandlingInstructions	Gets or sets the handling instructions for the shipment
<b>string</b> Information	Gets or sets the information about the shipment.  <b>Note</b> This element may contain notes or any other similar information that is not contained explicitly in another structure.
<b>Amount</b> InsuranceValueAmount	Gets or sets the insurance coverage of the shipment.
<b>Location</b> LastExitPortLocation	Gets or sets the final exporting location.  <b>Note</b> This location can be a port, an airport, or a border crossing.
<b>Measure</b> NetNetWeightMeasure	Gets or sets the weight of the goods without packaging and without the transport equipment
<b>Measure</b> NetVolumeMeasure	Gets or sets the volume of a shipment without packaging and the transport equipment
<b>Measure</b> NetWeightMeasure	Gets or sets total weight of the goods including the packaging.



Property	Description
<code>Address</code> OriginAddress	Gets or sets the address of the place where the goods are produced.  <b>Note</b> This address is used for the customs purposes such as customs tariff, quantitative restrictions, or any other measure related to the trade.
<code>string</code> ShippingPriorityLevelCode	Gets or sets the code of the service priority.
<code>string</code> SpecialInstructions	Gets or sets the special instructions for the shipment.
<code>bool</code> SplitConsignmentIndicator	Gets or sets the value that indicates whether the consignment has been split in transit or not.
<code>decimal</code> TotalGoodsItemQuantity	Gets or sets the number of goods items in the shipment.
<code>decimal</code> TotalTransportHandlingUnitQuantity	Gets or sets the number of pieces of transport handling. These are the equipment that is used in the shipment, such as the pallets, the boxes, and the cases.

### 2.3.15 State

This class implements the `IEntity` interface and contains information about the state.

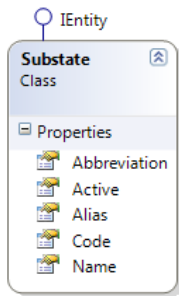


The properties of the `State` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the state that is used in the database.
<code>string</code> Code	Get or sets the code that identifies the state.
<code>string</code> Name	Gets or sets the name of the state.
<code>ICollection&lt;Substate&gt;</code> Substates	Gets the sub-states. A state potentially contains a collection of substates.

### 2.3.16 Substate

This class implements the `IEntity` interface and contains information about the substate.

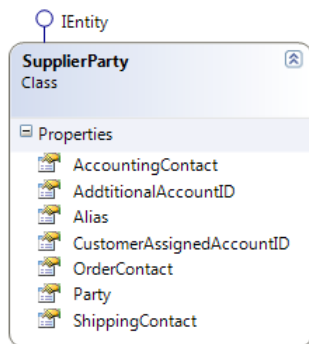


The properties of the `Substate` class are:

Property	Description
<code>string</code> Abbreviation	Gets or sets the abbreviation.
<code>bool</code> Active	Gets or sets a value that indicates whether this sub-state is active or not.
<code>long</code> Alias	Gets the ID of the sub-state that is used in the database.
<code>string</code> Code	Gets or sets the code of the sub-state.
<code>string</code> Name	Get or sets the name of the sub-state.

### 2.3.17 SupplierParty

This class implements the `IEntity` interface and contains information about the supplier.

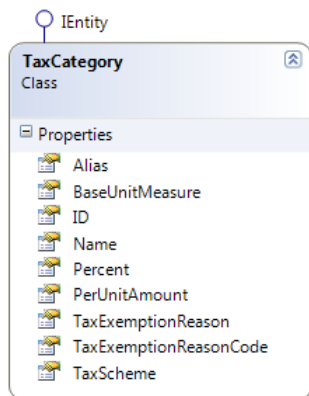


The properties of the `SupplierParty` class are:

Property	Description
<code>Contact</code> AccountingContact	Get or sets the information about the accounting contact of the supplier.
<code>string</code> AdditionalAccountID	Gets or sets the ID of the supplier account that is used in the third party system. It acts as a foreign key if you want to look up the supplier in that other system.
<code>long</code> Alias	Gets the ID of the supplier party that is used in the database.
<code>string</code> CustomerAssignedAccountID	Gets or sets the ID of the account of the customer.
<code>Contact</code> OrderContact	Gets or sets the information about the seller.
<code>Party</code> Party	Gets or sets the information about the party.
<code>Contact</code> ShippingContact	Gets or sets the information about the shipping contact.

## 2.3.18 TaxCategory

This class implements the `IEntity` interface and contains information about the category of the tax.

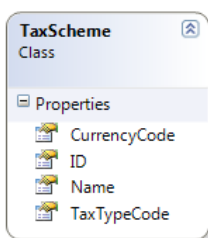


The properties of the `TaxCategory` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the tax category that is used in the database.
<code>Measure</code> BaseUnitMeasure	Gets or sets the measure that is used to calculate the tax, if the tax is applied at a certain rate per unit.
<code>string</code> ID	Gets or sets the identifier of the tax category.
<code>string</code> Name	Gets or sets the name of the tax category.
<code>decimal</code> Percent	Gets or sets the percentage of the tax category.
<code>Amount</code> PerUnitAmount	Gets or sets the tax that is set per unit if it is defined.
<code>string</code> TaxExemptionReason	Gets or sets the reason for not paying the tax.
<code>string</code> TaxExemptionReasonCode	Gets or sets the code that represents the reason for not paying the tax.
<code>TaxScheme</code> TaxScheme	Gets or sets the tax scheme.

## 2.3.19 TaxScheme

This class contains information about the scheme of the tax.

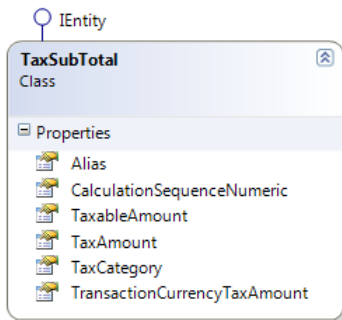


The properties of the `TaxScheme` class are:

Property	Description
<code>string</code> CurrencyCode	Gets or sets the code of the currency in which the tax is collected.
<code>string</code> ID	Gets or sets the identifier of the tax scheme.
<code>string</code> Name	Gets or sets the tax area.
<code>string</code> TaxTypeCode	Gets or sets the identification for the tax type.

### 2.3.20 TaxSubTotal

This class implements the `IEntity` interface. It contains information about the subtotal of a particular tax category within a tax type such as the standard tax rate within the VAT region.

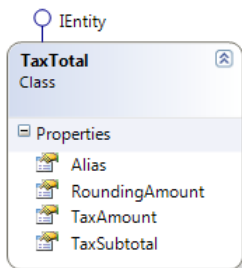


The Properties of `TaxSubTotal` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the tax sub-total that is used in the database.
<code>decimal</code> CalculationSequenceNumeric	Gets or sets the numerical order in which the taxes are applied when multiple taxes are defined. If all taxes apply to the same taxable amount, <code>CalculationSequenceNumeric</code> is set to '1' for all taxes
<code>Amount</code> TaxableAmount	Gets or sets the net amount to which the tax rate is applied.
<code>Amount</code> TaxAmount	Gets the amount of the tax.
<code>TaxCategory</code> TaxCategory	Gets or sets the category of the tax.
<code>Amount</code> TransactionCurrencyTaxAmount	Gets or sets the tax amount presented in the currency of the voice.

### 2.3.21 TaxTotal

This class implements the `IEntity` interface and contains information about the total amount of a particular tax type.



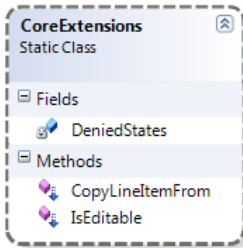
The properties of the `TaxTotal` class are:

Property	Description
<code>long</code> Alias	Gets the ID of the tax total that is used in the database.
<code>Amount</code> RoundingAmount	Gets or sets the rounding amount for taxes.  <b>Note</b> The rounding amount, whether positive or negative, is added to the calculated tax total to produce the rounded total tax value.
<code>Amount</code> TaxAmount	Gets the total for all taxes of a tax type.  <b>Note</b> This value is a calculated as the sum of the tax sub totals for each tax category within the tax type.
<code>ICollection&lt;TaxSubTotal&gt;</code> TaxSubtotal	Gets or sets information about the tax subtotal of the order.

## 2.4 Sitecore.Ecommerce.Core.OrderManagement.Extensions

### 2.4.1 CoreExtensions

This is a static class that defines the extensions of the core.



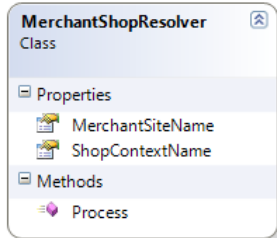
The methods of the `CoreExtension` class are:

Method	
<pre>void CopyLineItemFrom(     LineItem acceptor,     LineItem donor)</pre>	Takes the acceptor line item, the donor line item, and copies the data partially from one line item to another.
<pre>bool IsEditable(     Order order)</pre>	Takes the order and checks if the order is editable.

## 2.5 Sitecore.Ecommerce.Merchant.Pipelines.HttpRequest

### 2.5.1 MerchantShopResolver

This class defines the merchant shop resolver.



The properties of the `MerchantShopResolver` class are:

Property	Description
<code>string MerchantSiteName</code>	Gets or sets the name of the merchant site.
<code>string ShopContextName</code>	Gets or sets the name of the shop context.

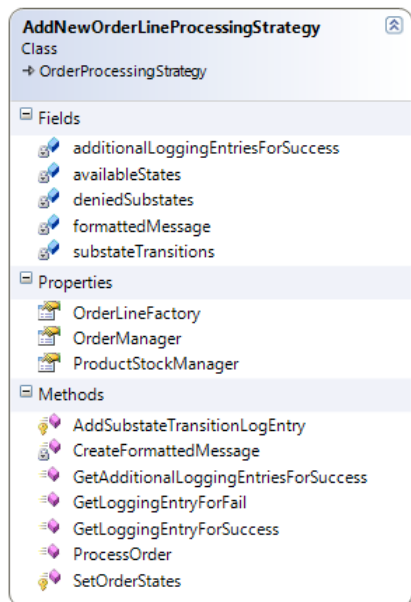
The only method of the `MerchantShopResolver` class is:

Method	Description
<code>void Process ( PipelineArgs args)</code>	Takes the pipeline arguments and runs the processor.

## 2.6 Sitecore.Ecommerce.Merchant.OrderManagement

### 2.6.1 AddNewOrderLineProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the strategy for adding a new order line to the existing order.



The properties of the `AddNewOrderLineProcessorStrategy` class are:

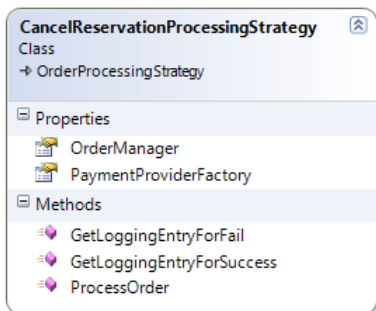
Property	Description
<code>MerchantOrderManager</code> <code>OrderManager</code>	Gets or sets the order manager.
<code>IProductStockManager</code> <code>ProductStockManager</code>	Gets or sets the product stock manager.
<code>OrderLineFactory</code> <code>OrderLineFactory</code>	Gets or sets the order line factory.

The methods of the `AddOrderLineProcessorStrategy` class are:

Method	Description
<code>string</code> <code>ProcessOrder</code> ( <code>Order</code> order, <code>IDictionary&lt;string, object&gt;</code> parameters)	Takes the order, the parameters and processes the order.
<code>LoggingEntry</code> <code>GetLoggingEntryForSuccess</code> ( <code>Order</code> order)	Takes the order and gets the log entry for the operations that resulted in success.
<code>LoggingEntry</code> <code>GetLoggingEntryForFail</code> ( <code>Order</code> order, <code>params object[]</code> parameters)	Takes the order, the parameters, and gets the log entry for the operations that resulted in failure.
<code>IList&lt;LoggingEntry&gt;</code> <code>GetAdditionalLoggingEntriesForSuccess</code> ( <code>Order</code> order)	Takes the order and gets the additional log entries for success.

## 2.6.2 CancelReservationProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the strategy for canceling the reservation of the money.



The properties of the `CancelReservationProcessingStrategy` class are:

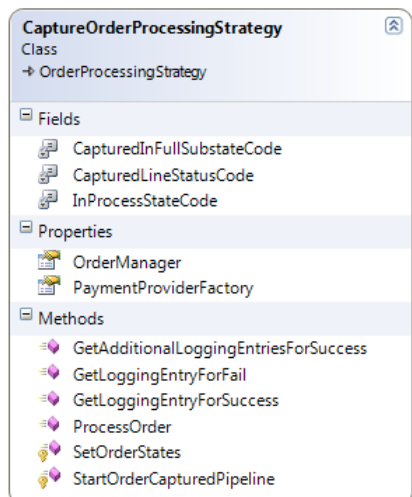
Property	Description
<code>MerchantOrderManager</code> <code>OrderManager</code>	Gets or sets the order manager.
<code>PaymentProviderFactory</code> <code>PaymentProviderFactory</code>	Gets or sets the payment provider factory.

The methods of the `CancelReservationProcessingStrategy` class are:

Method	Description
<code>string</code> <code>ProcessOrder</code> ( <code>Order</code> order, <code>IDictionary&lt;string, object&gt;</code> parameters)	Takes the order, the parameters, and processes the order.
<code>LoggingEntry</code> <code>GetLoggingEntryForSuccess</code> ( <code>Order</code> order)	Takes the order and gets the logged message of the successful operation.
<code>LoggingEntry</code> <code>GetLoggingEntryForFail</code> ( <code>Order</code> order, <code>params object[]</code> parameters)	Takes the order, the parameters, and gets the logged message of the operation that failed.

## 2.6.3 CaptureOrderProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the processing strategy for capturing the order.



The properties of the `CaptureOrderProcessingStrategy` class are:

Property	Description
<code>MerchantOrderManager</code> <code>OrderManager</code>	Gets or sets the order manager.
<code>PaymentProviderFactory</code> <code>PaymentProviderFactory</code>	Gets or sets the payment provider factory.

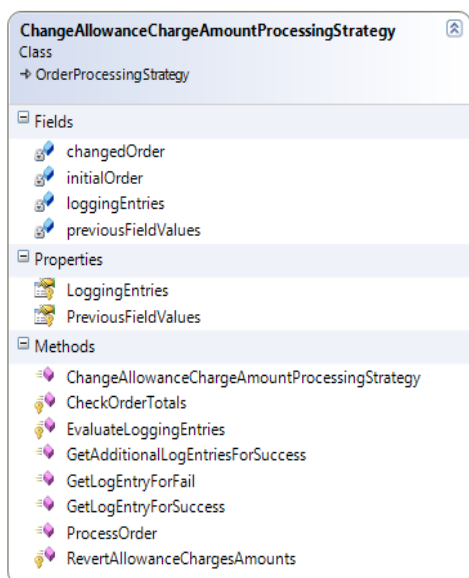


The methods of the `CaptureOrderProcessingStrategy` class are:

Method	Description
<pre>string ProcessOrder (     Order order,     IDictionary&lt;string, object&gt; parameters)</pre>	Takes the order, the parameters, and processes the order.
<pre>LoggingEntry GetLoggingEntryForSuccess (     Order order)</pre>	Takes the order, and gets the logged message of the successful operation.
<pre>IList&lt;LoggingEntry&gt; GetAdditionalLoggingEntriesForSuccess (     Order order)</pre>	Takes the order, and gets the additional logged message of the successful operation.
<pre>LoggingEntry GetLoggingEntryForFail (     Order order, params object[] parameters)</pre>	Takes the order, the parameters and gets the logged message of the operation that failed.

## 2.6.4 ChangeAllowanceChargeAmountProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the processing strategy of the allowance charge amount value.

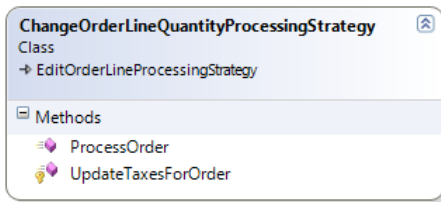


The methods of the `ChangeAllowanceChargeAmountProcessingStrategy` are:

Method	Description
<pre>ChangeAllowanceChargeAmountProcessingStrategy (     Order initialOrder,     IDictionary&lt;string, object&gt;     previousFieldValues)</pre>	Initializes a new instance of the class.
<pre>IList&lt;LogEntry&gt; GetAdditionalLogEntriesForSuccess (Order order)</pre>	Performs additional success processing.
<pre>LogEntry GetLogEntryForFail (     Order order,     params object[]     parameters)</pre>	Performs additional failure processing.
<pre>LogEntry GetLogEntryForSuccess (     Order order)</pre>	Gets logging entries for success.
<pre>string ProcessOrder (     Order order,     IDictionary&lt;string, object&gt; parameters)</pre>	Processes the order.

## 2.6.5 ChangeOrderLineQuantityProcessingStrategy

This class implements the `EditOrderLineProcessingStrategy` class and defines the processing strategy for changing the order line quantity.

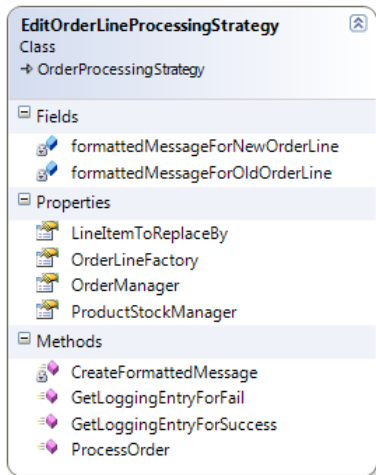


The only public method of the `ChangeOrderLineQuantityProcessingStrategy` class is:

Method	Description
<code>string ProcessOrder (Order order, IDictionary&lt;string, object&gt; parameters)</code>	Processes the order.

## 2.6.6 EditOrderLineProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the processing strategy for editing the order line.



The properties of the `EditOrderLineProcessingStrategy` class are:

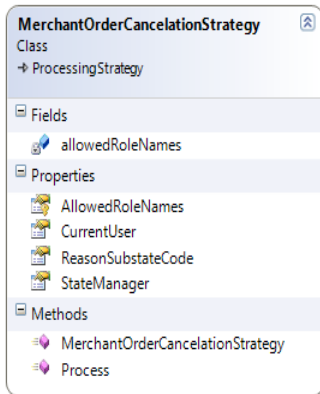
Property	Description
<code>IProductStockManager ProductStockManager</code>	Gets or sets the product stock manager.
<code>MerchantOrderManager OrderManager</code>	Gets or sets the order manager.
<code>OrderLineFactory OrderLineFactory</code>	Gets or sets the order line factory.
<code>LineItem LineItemToReplaceBy</code>	Gets or sets the order line item to be added.

The methods of the `EditOrderLineProcessingStrategy` class are:

Method	Description
<code>string ProcessOrder (Order order, IDictionary&lt;string, object&gt; parameters)</code>	Takes the order, the parameters, and processes the order.
<code>LoggingEntry GetLoggingEntryForSuccess (Order order)</code>	Takes the order, and gets the logged message of the successful operation.
<code>LoggingEntry GetLoggingEntryForFail (Order order, params object[] parameters)</code>	Takes the order, the parameters, and the gets the logged message of the operation that failed.

## 2.6.7 MerchantOrderCancellationStrategy

This class implements the `ProcessingStrategy` class and defines the strategy for cancelling the merchant order.



The properties of the `MerchantOrderCancellationStrategy` class are:

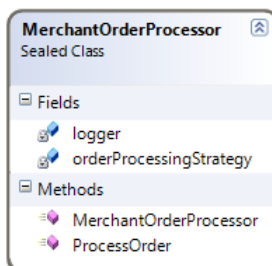
Property	Description
<code>Core.Contracts.OrderManagement.StateManager&lt;State&gt; StateManager</code>	Gets or sets the state manager.
<code>User currentUser</code>	Gets or sets the current user.
<code>string ReasonSubstateCode</code>	Gets or sets the reason sub-state code.
<code>virtual IEnumerable&lt;string&gt; AllowedRoleNames</code>	Gets the allowed role names.

The methods of the `MerchantOrderCancellationStrategy` class are:

Method	Description
<code>void Process (Order order)</code>	Takes the order and processes it.
<code>MerchantOrderCancellationStrategy ()</code>	Initializes a new instance of the <code>MerchantOrderCancellationStrategy</code> class.

## 2.6.8 MerchantOrderProcessor

This class is sealed and defines the merchant order processor.

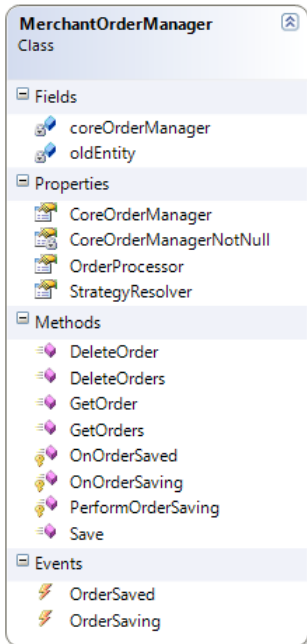


The methods of the `MerchantOrderProcessor` class are:

Method	Description
<code>MerchantOrderProcessor (OrderProcessingStrategy orderProcessingStrategy)</code>	Takes the order processing strategy and initializes a new instance of the <code>MerchantOrderProcessor</code> class.
<code>string ProcessOrder (Order order, IDictionary&lt;string, object&gt; parameters)</code>	Takes the order, the parameters, and processes the order.  <b>Note</b> You can pass any stuff to <code>parameters</code> to handle your own business scenario.

## 2.6.9 MerchantOrderManager

This class defines the merchant order manager.



The properties of the `MerchantOrderManager` class are:

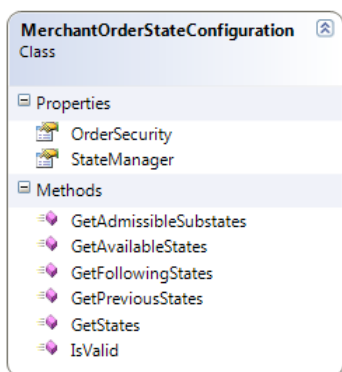
Property	Description
<code>CoreOrderManager</code> CoreOrderManager	Gets or sets the core order manager.
<code>MerchantOrderProcessor</code> OrderProcessor	Gets or sets the order processor.
<code>OrderProcessingStrategyResolver</code> StrategyResolver	Gets or sets the strategy resolver.

The methods of the `MerchantOrderManager` class are:

Method	Description
<code>void DeleteOrder ( Order order)</code>	Deletes the order.
<code>void DeleteOrders ( Expression&lt;Func&lt;Order, bool&gt;&gt; expression)</code>	Takes the Linq expression and deletes the orders.
<code>Order GetOrder ( string orderId)</code>	Takes the order ID and returns the order.
<code>IQueryable&lt;Order&gt; GetOrders ( Expression&lt;Func&lt;Order, bool&gt;&gt; expression)</code>	Takes the Linq expression and returns the orders.
<code>void Save ( Order order)</code>	Saves the order.

## 2.6.10 MerchantOrderStateConfiguration

This class defines the merchant order state configuration.



The properties of the `MerchantOrderStateConfiguration` are:

Property	Description
<code>OrderSecurity</code> OrderSecurity	Gets or sets the order security.
<code>CoreOrderStateConfiguration</code> StateManager	Gets or sets the state manager.

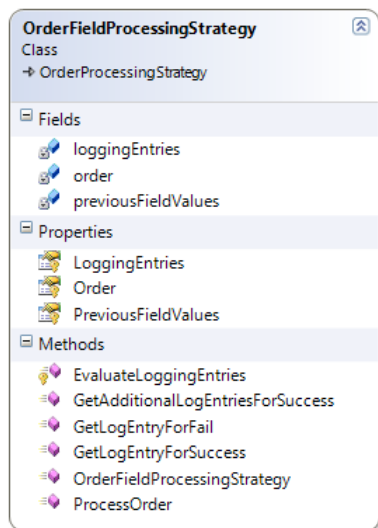
The Methods of the `MerchantOrderStateConfiguration` are:

Method	Description
<code>IQueryable&lt;Substate&gt; GetAdmissibleSubstates ( State state)</code>	Gets the admissible substates of the merchant order.
<code>IQueryable&lt;State&gt; GetAvailableStates ( Order order)</code>	Gets the available states of the merchant order.

Method	Description
<code>IQueryable&lt;State&gt;</code> <code>GetFollowingStates (</code> <code>State state)</code>	Takes the merchant order state and gets the following states.
<code>IQueryable&lt;State&gt;</code> <code>GetPreviousStates (</code> <code>State state)</code>	Takes the merchant order state and gets the previous states.
<code>IQueryable&lt;State&gt;</code> <code>GetStates ()</code>	Gets the merchant order states.
<code>bool</code> <code>IsValid (</code> <code>State state)</code>	Determines whether the specified merchant order state is valid.

### 2.6.11 OrderFieldProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the strategy for processing the order field.

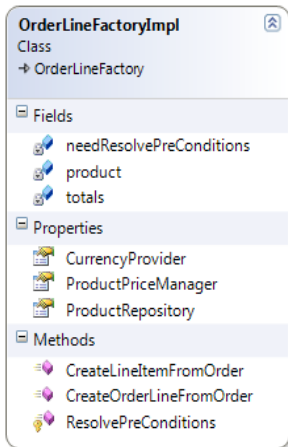


The methods of the `OrderFieldProcessingStrategy` class are:

Method	Description
<code>IList&lt;LogEntry&gt;</code> <code>GetAdditionalLogEntriesForSuccess (</code> <code>Order order)</code>	Performs additional success processing.
<code>LogEntry</code> <code>GetLogEntryForFail (</code> <code>Order order,</code> <code>params object[] parameters)</code>	Gets the logged message of the operation that failed and performs additional
<code>LogEntry</code> <code>GetLogEntryForSuccess (</code> <code>Order order)</code>	Gets the logged message of the successful operation.
<code>OrderFieldProcessingStrategy (</code> <code>IDictionary&lt;string, object&gt;</code> <code>previousFieldValues)</code>	Initializes a new instance of the class.
<code>string</code> <code>ProcessOrder (</code> <code>Order order,</code> <code>IDictionary&lt;string, object&gt;</code> <code>parameters)</code>	Processes the order.

### 2.6.12 OrderLineFactoryImpl

This class implements the `OrderLineFactory` and defines the implementation of the order line factory.



The properties of the `OrderLineFactoryImpl` class are:

Property	Description
<code>IProductRepository</code> ProductRepository	Gets or sets the product repository.
<code>IProductPriceManager</code> ProductPriceManager	Gets or sets the product price manager.
<code>IEntityProvider&lt;DomainModel.Currencies.Currency&gt;</code> CurrencyProvider	Gets or sets the currency provider.

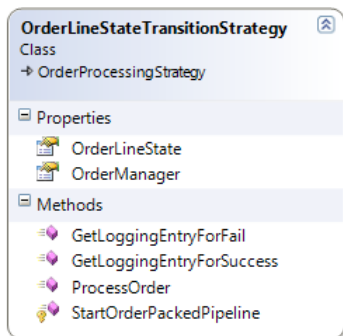
The methods of the `OrderLineFactoryImpl` class are:

Method	Description
<code>OrderLine</code> CreateOrderLineFromOrder ( Order order, string productCode, long quantity)	Takes the order, the product code, the quantity, and creates the order line.
<code>LineItem</code> CreateLineItemFromOrder ( Order order, string productCode, long quantity)	Takes order, the product code, the quantity, and creates the line item.

### 2.6.13 OrderLineStateTransitionStrategy

This class implements the `OrderProcessingStrategy` class and defines the strategy of the order line state transition.

This class defines the order line state transition strategy.



The properties of the `OrderLineStateTransitionStrategy` class are:

Property	Description
<code>string</code> OrderLineState	Gets or sets the order line status.
<code>MerchantOrderManager</code> OrderManager	Gets or sets the order manager.

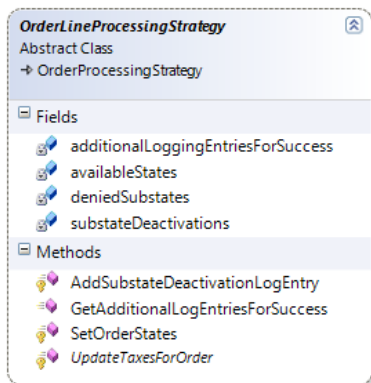
The methods of the `OrderLineStateTransitionStrategy` class are:

Method	Description
<code>string</code> ProcessOrder ( Order order, IDictionary<string, object> parameters)	Takes the order, the parameters, and processes the order.

Method	Description
<pre>LoggingEntry GetLoggingEntryForSuccess (     Order order)</pre>	Takes the order, and gets the logged message of the successful operation.
<pre>LoggingEntry GetLoggingEntryForFail (     Order order,     params object[] parameters)</pre>	Takes the order, the parameters, and gets the logged message of the operation that failed.
<pre>void StartOrderPackedPipeline (     Order order)</pre>	Takes the order, and starts the order packed pipeline.

### 2.6.14 OrderLineProcessingStrategy

This is an abstract class that implements the `OrderProcessingStrategy` and defines the strategy for processing the order line.

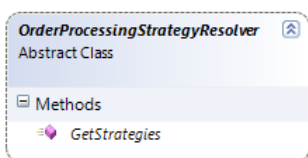


The only public method in the `OrderLineProcessingStrategy` class is:

Method	Description
<pre>IList&lt;LogEntry&gt; GetAdditionalLogEntriesForSuccess (     Order order)</pre>	Gets additional logging entry for success.

### 2.6.15 OrderProcessingStrategyResolver

This is an abstract class that defines the strategy resolver. This resolver is used to process the order.

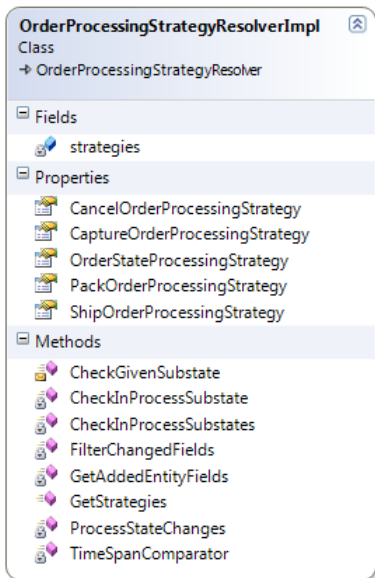


The only method of `OrderProcessingStrategyResolver` class is:

Method	Description
<pre>IEnumerable&lt;OrderProcessingStrategy&gt; GetStrategies (     Order oldOrder,     Order newOrder)</pre>	Gets the order processing strategy. It compares two orders and decides what to do with them. For example, it may realize that the order has been changed, moved to a process state, or captured. The class will resolve the appropriate strategy that can handle it.

## 2.6.16 OrderProcessingStrategyResolverImpl

This class implements the `OrderProcessingStrategyResolver` class and represents the implementation of the order processing strategy resolver.



The properties of the `OrderProcessingStrategyResolverImpl` class are:

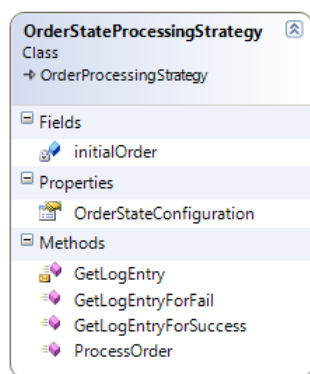
Property	Description
<code>OrderProcessingStrategy</code> <code>CancelOrderProcessingStrategy</code>	Gets or sets the cancel order processing strategy.
<code>OrderProcessingStrategy</code> <code>CaptureOrderProcessingStrategy</code>	Gets or sets the capture order processing strategy.
<code>OrderProcessingStrategy</code> <code>OrderStateProcessingStrategy</code>	Gets or sets the order state processing strategy.
<code>OrderProcessingStrategy</code> <code>PackOrderProcessingStrategy</code>	Gets or sets the pack order processing strategy.
<code>OrderProcessingStrategy</code> <code>ShipOrderProcessingStrategy</code>	Sets or sets the ship order processing strategy.

The methods of the `OrderProcessingStrategyResolverImpl` class are:

Method	Description
<code>IEnumerable&lt;OrderProcessingStrategy&gt;</code> <code>GetStrategies (</code> <code>Order</code> oldOrder, <code>Order</code> newOrder)	Gets the order processing strategies.

## 2.6.17 OrderStateProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the strategy for processing the order state.



The only property of the `OrderStateProcessingStrategy` class is:

Property	Description
<code>MerchantOrderStateConfiguration</code> <code>OrderStateConfiguration</code>	Gets or sets the order state configuration.

The methods of the `OrderStateProcessingStrategy` class are:

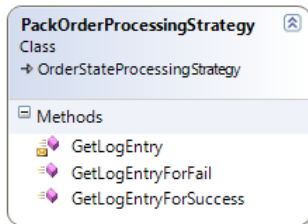
Method	Description
<code>LogEntry</code> <code>GetLogEntryForFail (</code> <code>Order</code> order, <code>params object[]</code> parameters)	Gets the logged message of the operation that failed and performs additional failure processing.
<code>LogEntry</code> <code>GetLogEntryForSuccess (</code> <code>Order</code> order)	Gets the logged message of the successful operation.



Method	Description
<pre>string ProcessOrder (     Order order,     IDictionary&lt;string,     object&gt; parameters)</pre>	Processes the order.

### 2.6.18 PackOrderProcessingStrategy

This class implements the `PackStateProcessingStrategy` class and defines the strategy for processing the pack order.

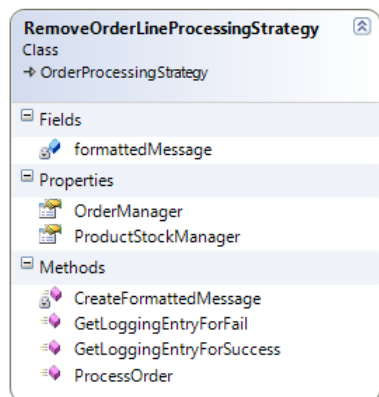


The methods of the `PackOrderProcessingStrategy` class are:

Method	Description
<pre>LogEntry GetLogEntryForFail (     Order order,     params object[]     parameters)</pre>	Gets the logged message of the operation that failed and performs additional failure processing.
<pre>LogEntry GetLogEntryForSuccess (     Order order)</pre>	Gets the logged message of the successful operation.

### 2.6.19 RemoveOrderLineProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the processing strategy for removing the order line.



The properties of the `RemoveOrderLineProcessingStrategy` class are:

Property	Description
<pre>MerchantOrderManager OrderManager</pre>	Gets or sets the order manager.
<pre>IProductStockManager ProductStockManager</pre>	Gets or sets the product stock manager.

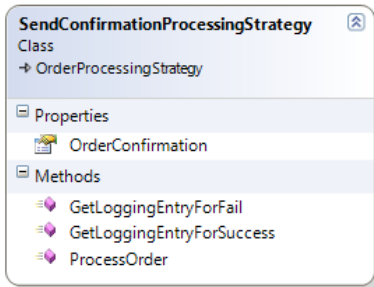
The methods of the `RemoveOrderLineProcessingStrategy` class are:

Method	Description
<pre>string ProcessOrder (     Order order,     IDictionary&lt;string,     object&gt; parameters)</pre>	Takes the order, the parameters, and processes the order.
<pre>LoggingEntry GetLoggingEntryForSuccess (     Order order)</pre>	Takes the order, and gets the logged message of the successful operation.

Method	Description
<pre>LoggingEntry GetLoggingEntryForFail (     Order order,     params object[]     parameters)</pre>	Takes the order, the parameters, and gets the logged message of the operation that failed.

### 2.6.20 SendConfirmationProcessingStrategy

This class implements the `OrderProcessingStrategy` class and defines the processing strategy for sending the order confirmation.



The only property of the `SendConfirmationProcessingStrategy` class is:

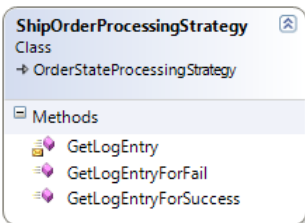
Property	Description
<pre>OrderConfirmation OrderConfirmation</pre>	Gets or sets the order confirmation.

The methods of the `SendConfirmationProcessingStrategy` class are:

Method	Description
<pre>string ProcessOrder (     Order order,     IDictionary&lt;string,     object&gt; parameters)</pre>	Takes the order, the parameters, and processes the order.
<pre>LoggingEntry GetLoggingEntryForSuccess (     Order order)</pre>	Takes the order, and gets the logged message of the successful operation.
<pre>LoggingEntry GetLoggingEntryForFail (     Order order,     params object[]     parameters)</pre>	Takes the order, the parameters, and gets the logged message of the operation that failed.

### 2.6.21 ShipOrderProcessingStrategy

This class implements the `OrderStateProcessingStrategy` class and defines the processing strategy for shipping the order.

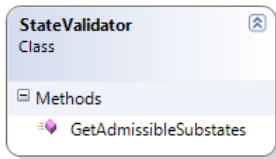


The methods of the `ShipOrderProcessingStrategy` class are:

Method	Description
<pre>LogEntry GetLogEntryForFail (     Order order,     params object[]     parameters)</pre>	Gets the logged message of the operation that failed and performs additional failure processing.
<pre>LogEntry GetLogEntryForSuccess (     Order order)</pre>	Gets the logged message of the successful operation.

## 2.6.22 StateValidator

This class defines the state validator.



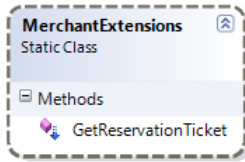
The only method of the `StateValidator` class is:

Method	Description
<code>IEnumerable&lt;Substate&gt;</code> <code>GetAdmissibleSubstates (</code> <code>State state,</code> <code>SubstateCombinationSet</code> <code>substateCombinationSet)</code>	Takes the state, the sub-state combination set, and gets the admissible sub-states.

## 2.7 Sitecore.Ecommerce.Merchant.OrderManagement.Extensions

### 2.7.1 MerchantExtensions

This is a static class that defines the merchant extensions.



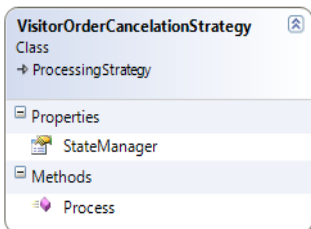
The only method of the `MerchantExtensions` class is:

Method	Description
<code>ReservationTicket</code> <code>GetReservationTicket (</code> <code>    this Order order)</code>	Takes the order, and creates the reservation tickets from the order.

## 2.8 Sitecore.Ecommerce.Visitor.OrderManagement

### 2.8.1 VisitorOrderCancellationStrategy

This class implements the `ProcessingStrategy` class and defines the strategy for cancelling the visitor order.



The only property of the `VisitorOrderProcessingStrategy` class is:

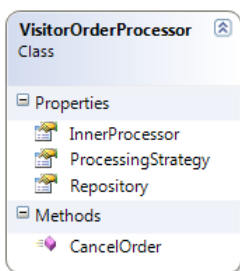
Property	Description
<code>StateManager&lt;State&gt;</code> StateManager	Gets or sets the state manager.

The only method of the `VisitorOrderProcessingStrategy` class is:

Method	Description
<code>void Process (Order order)</code>	Takes the order and processes it.

### 2.8.2 VisitorOrderProcessor

This class defines the processor of the visitor order.



The properties of the `VisitorOrderProcessor` class are:

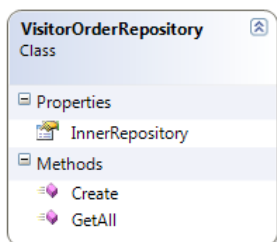
Property	Description
<code>OrderProcessor</code> InnerProcessor	Gets or sets the inner processor.
<code>ProcessingStrategy</code> ProcessingStrategy	Gets or sets the processing strategy.
<code>Repository&lt;Order&gt;</code> Repository	Gets or sets the repository.

The only method of the `VisitorOrderProcessor` class is:

Method	Property
<code>virtual void CancelOrder (Order order)</code>	Takes the order, and cancels it.

### 2.8.3 VisitorOrderRepository

This class defines the visitor order repository.



The only property of the `VisitorOrderRepository` class is:

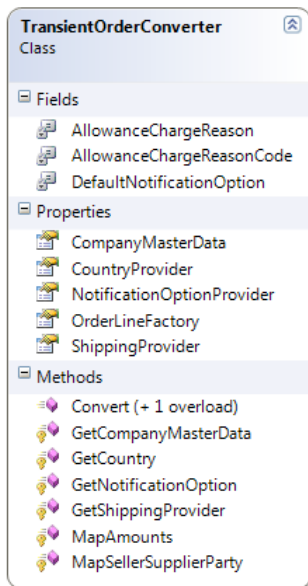
Property	Description
<code>Repository&lt;Order&gt;</code> InnerRepository	Gets or sets the inner order repository.

The methods of the `VisitorOrderRepository` class are:

Method	Description
<code>void Create (Order order)</code>	Takes the order, and creates it.
<code>IQueryable&lt;Order&gt; GetAll (Expression&lt;Func&lt;Order, bool&gt;&gt; expression)</code>	Takes the expression, and gets all the orders.

## 2.8.4 TransientOrderConverter

This class defines the transient order converter.



The properties of the `TransientOrderConverter` class are:

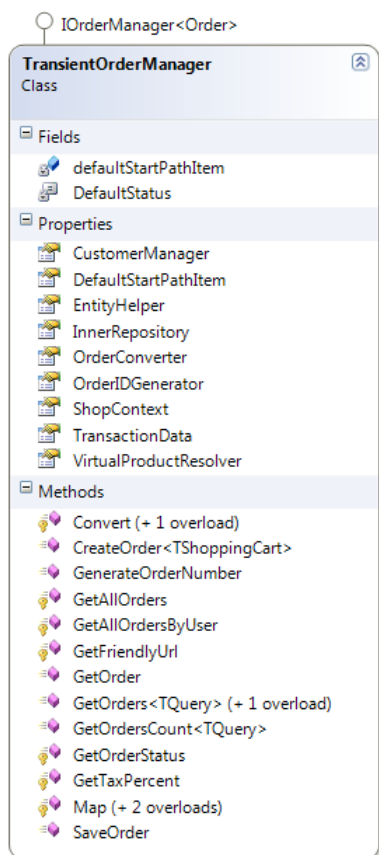
Property	Description
<code>CompanyMasterData CompanyMasterData</code>	Gets or sets the company master data.
<code>IEntityProvider&lt;Country&gt; CountryProvider</code>	Gets or sets the country provider.
<code>IEntityProvider&lt;NotificationOption&gt; NotificationOptionProvider</code>	Gets or sets the notification option provider.
<code>OrderLineFactory OrderLineFactory</code>	Gets or sets the order line factory.
<code>IEntityProvider&lt;ShippingProvider&gt; ShippingProvider</code>	Gets or sets the shipping provider.

The only public method of the `TransientOrderConverter` class is:

Method	Description
<code>Order Convert (DomainModel.Orders.Order source)</code>	Converts the specified source.

## 2.8.5 TransientOrderManager

This class implements the `IOrderManager` interface and defines the transient order manager.



The properties of the `TransientOrderManager` class are:

Property	Description
<code>ICustomerManager&lt;CustomerInfo&gt;</code> CustomerManager	Gets or sets the customer manager.
<code>Sitecore.Data.Items.Item</code> DefaultStartPathItem	Gets or sets the default start path item.
<code>EntityHelper</code> EntityHelper	Gets or sets the entity helper instance.
<code>VisitorOrderRepository</code> InnerRepository	Gets or sets the inner repository.
<code>TransientOrderConverter</code> OrderConverter	Gets or sets the order converter.
<code>OrderIDGenerator</code> OrderIDGenerator	Gets or sets the order number generator.
<code>ShopContext</code> ShopContext	Gets or sets the shop context.
<code>ITransactionData</code> TransactionData	Gets or sets the transaction data.
<code>VirtualProductResolver</code> VirtualProductResolver	Gets or sets the virtual product resolver.

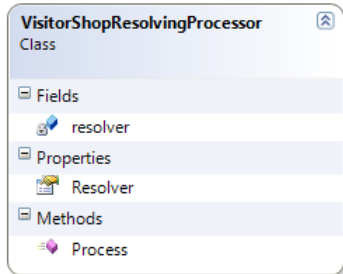
The public methods of the `TransientOrderManager` class are:

Method	Description
<code>OldOrder</code> CreateOrder<TShoppingCart>( TShoppingCart shoppingCart) where TShoppingCart : ShoppingCart	Takes the shopping cart and creates the order.
<code>string</code> GenerateOrderNumber()	Generates the order number.
<code>virtual OldOrder</code> GetOrder( string orderNumber)	Takes the order number gets the order.
<code>virtual IEnumerable&lt;OldOrder&gt;</code> GetOrders<TQuery>( TQuery query)	Takes the query and gets the orders.
<code>virtual IEnumerable&lt;OldOrder&gt;</code> GetOrders<TQuery>( TQuery query, int pageIndex, int pageSize)	Takes the query, the page index and the page size and gets the orders.

## 2.9 Sitecore.Ecommerce.Visitor.Pipelines.HttpRequest

### 2.9.1 VisitorShopResolvingProcessor

This class defines the resolving processor of the visitor shop.



The only property of the `VisitorShopResolvingProcessor` class is:

Property	Description
<code>VisitorShopResolver</code> Resolver	Gets or sets the resolver.

The only method of the `VisitorShopResolvingProcessor` class is:

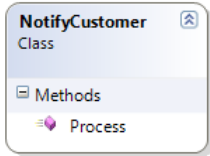
Method	Description
<code>void Process ( PipelineArgs args)</code>	Runs the processor.



## 2.10 Sitecore.Ecommerce.Visitor.Pipelines.OrderedCreated

### 2.10.1 NotifyCustomer

This class defines the process of notifying the customer.



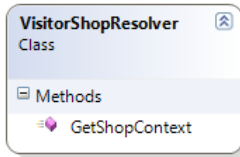
The only method of the `NotifyCustomer` class is:

Method	Description
<code>void Process ( PipelineArgs args)</code>	Runs the processor.

## 2.11 Sitecore.Ecommerce.Visitor.Sites

### 2.11.1 VisitorShopResolver

This class defines the resolver of the visitor shop.



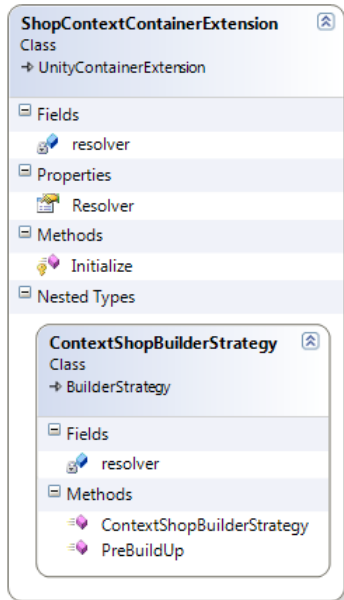
The only method of the `VisitorShopResolver` class is:

Method	Description
<code>ShopContext GetShopContext( SiteContext siteContext)</code>	Gets the shop context.

## 2.12 Sitecore.Ecommerce.Visitor.Unity

### 2.12.1 ShopContextContainerExtension

This class defines the extension of the site context container.



The only property of the `ShopContextContainerExtension` is:

Method	Description
<code>VisitorShopResolver</code> <code>Resolver</code>	Gets or sets the resolver.