Multiple Warehouse Inventory Management

Magento 1

Overview

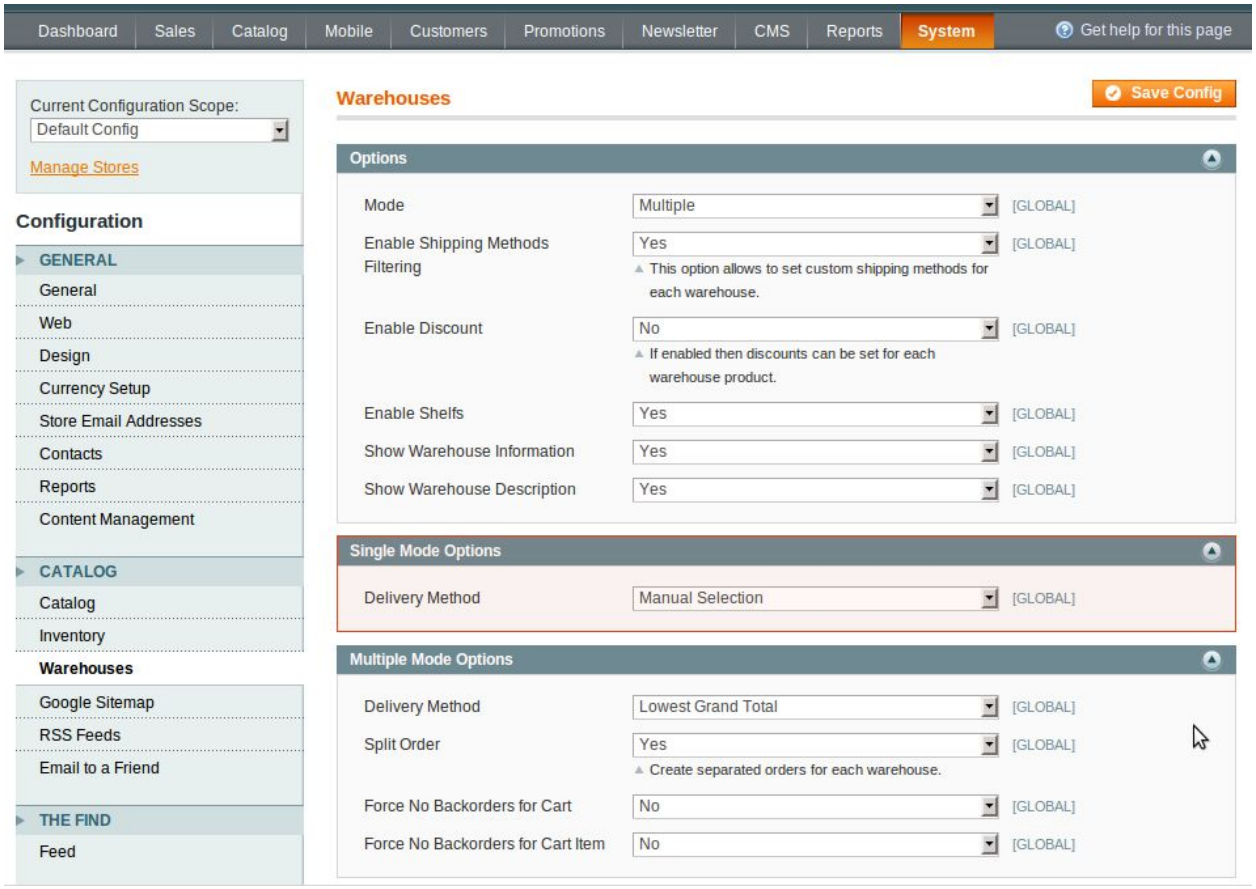
When scaling your business, you will most likely encounter the management of multiple Magento® warehouses. It can also pose a problem when you already have two or more store houses. Magento only let's you ship items from a single storage warehouse. Therefore, you can’t manage your multiple global warehouses.

We offer you our solution. Merchant Protocol implements the Multi-Warehouse extension with improved functionalities, providing you with more options and features. This irreplaceable tool extends the capabilities of your multiple warehouses management.

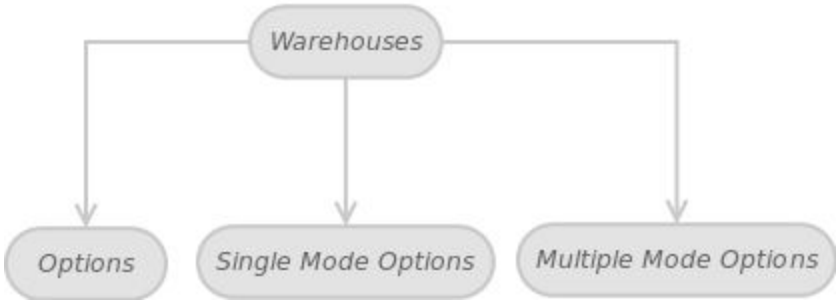
Use the power of this Magento extension to organize your stock and multi-depot. You'll be able to solve a wide range of tasks, including inventory management and improving warehouse performance. Enjoy all the advantages of dealing with our powerful module.

Single Mode

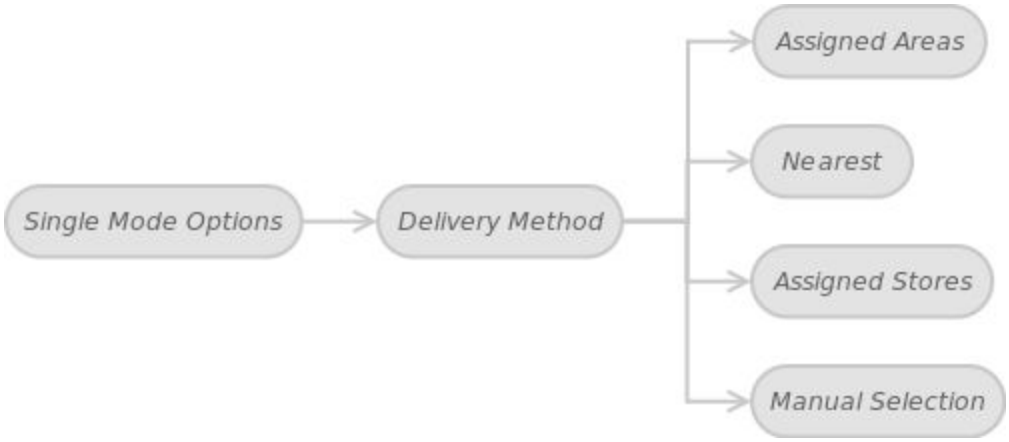
The given article is devoted to the Multi-Warehouse Magento extension's Single Mode Options.



In the previous article, we used the illustration to show the block diagram in Figure 1. Options Overview:



The purpose of this paper is to explain the Single Mode Options. Let's take a look at the following schematic diagram.



In the Configuration article, we mentioned that a customer may be assigned to a particular warehouse. You can configure this option per delivery method: Assigned Areas, Nearest, Assigned Stores, Manual Selection.

Let's represent the initial data in tabular form:

Customer	City	State	Zip / Postal Code	Country
John Doe	Colby	Kansas	67701	United States

Table 1 Customer Data

Warehouse	City	State	Zip / Postal Code	Country
Colorado	Denver	Colorado	80202	United States
Kansas	Kansas City	Kansas	66117	United States
New York	New York	New York	10118	United States
California	Sacramento	California	95866	United States

Table 2 Warehouse Data

Assigned Areas

If this option is enabled, the Areas tab becomes visible to the warehouse editor. You assign a customer to a warehouse by area (region, state, etc). In this case, that will be Kansas, since this warehouse is located in the customer's state. The system tracks the customer's location using Customer Locator feature.

Nearest

Use this option to assign a customer to the closest warehouse. In this case, the system automatically chooses Colorado because it's nearest to John Doe's location.

Assigned Stores

As you know, each website is a collection of store views that share the same customer, order information and shopping cart details. This option may be useful if you have multiple store views in your website and want to map these views to the warehouses. The stores tab is available for a warehouse editor if this method is active. This way a customer will be assigned to a warehouse according to the current store view.

Manual Selection

This method allows a customer to choose the appropriate warehouse manually. If this option is enabled, then Your Warehouse box becomes visible to a customer.

Multiple Mode Delivery Methods

In previous articles we have considered: Main Options and Single Mode Options for the Multi-Warehouse Magento module. This article deals with the Multiple Mode Options.

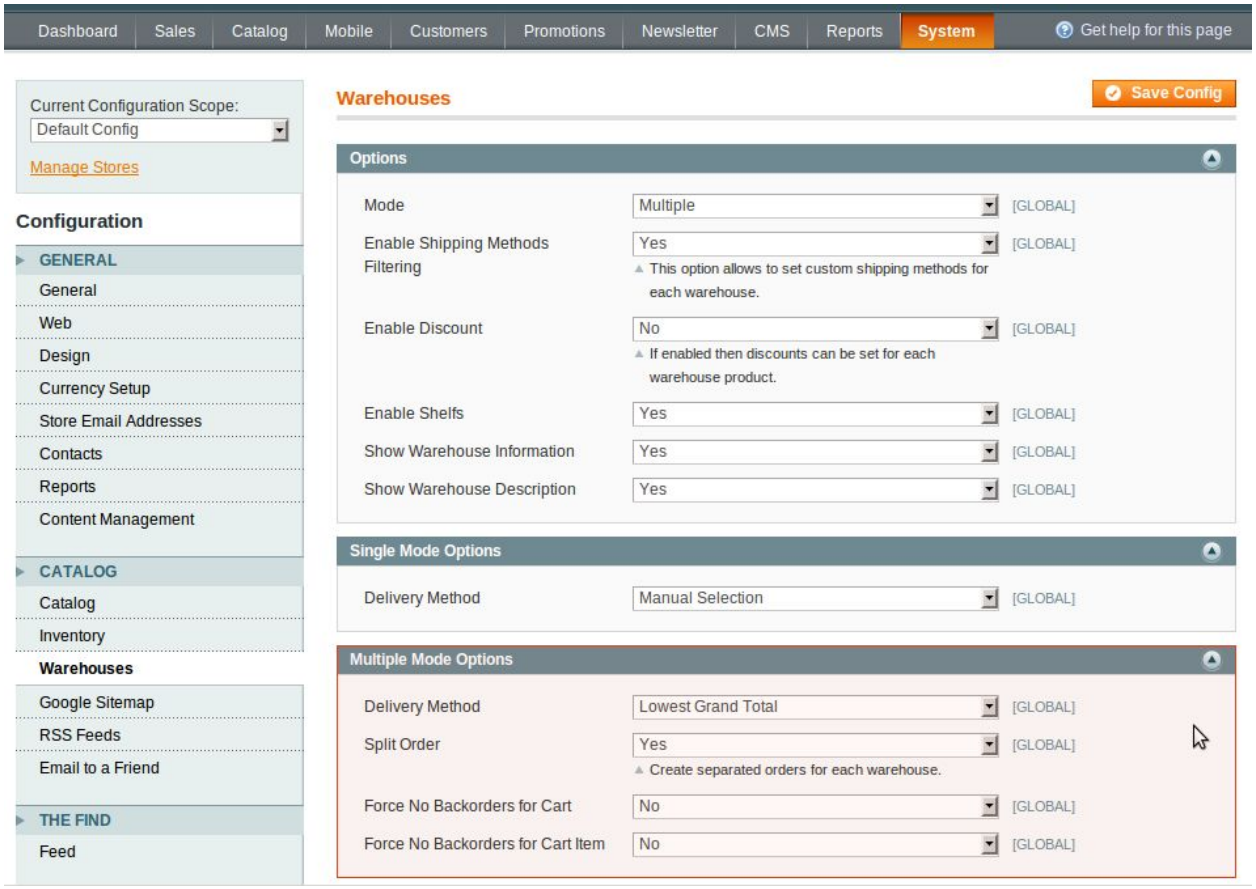
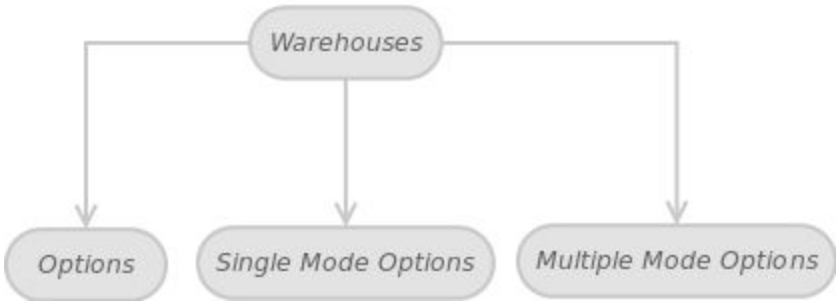
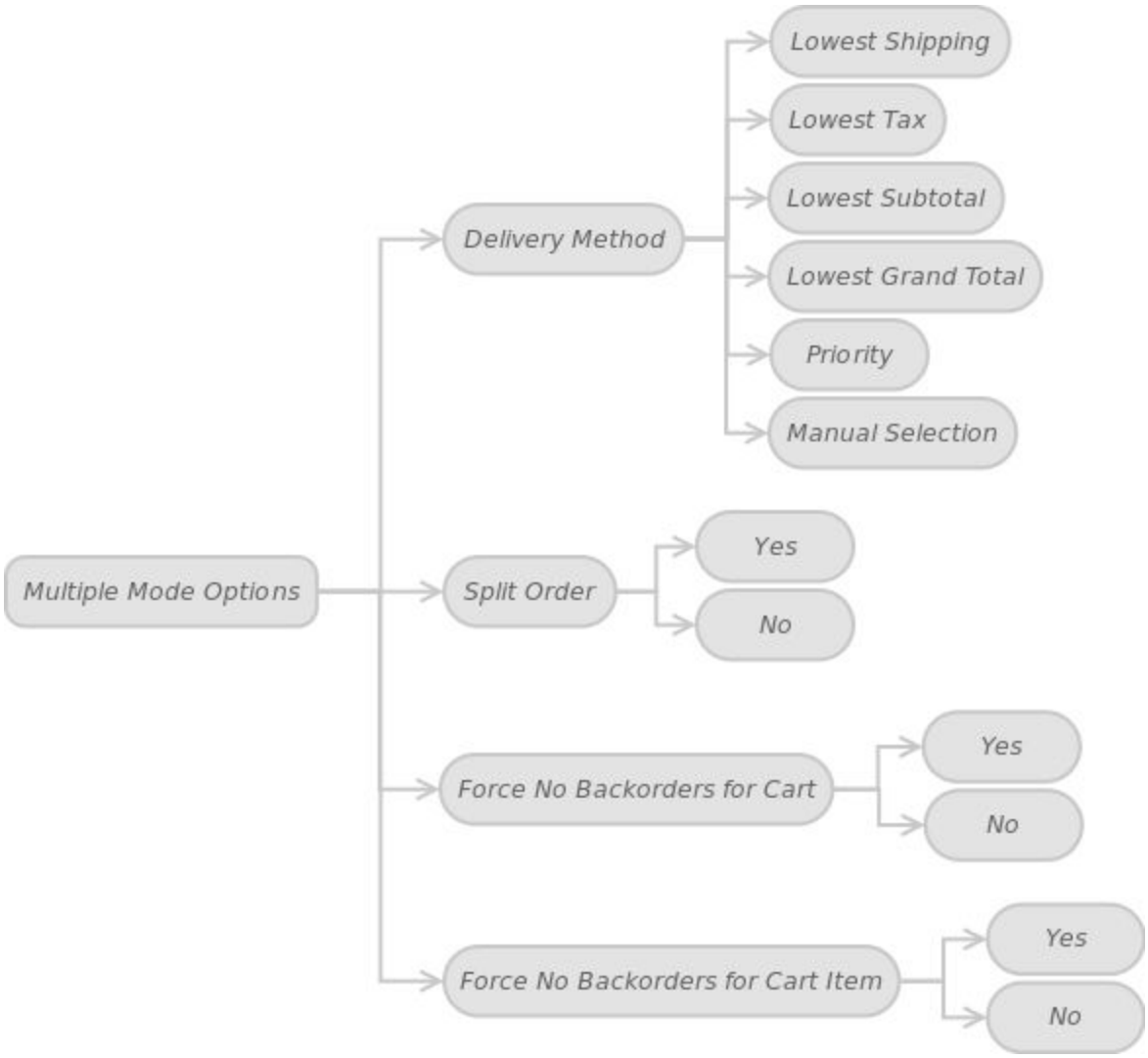


Figure 1. Options Overview shows a schematic illustration of the Multi-Warehouse options



In turn,



For convenience, this article will be split into two parts. The first part is dedicated to the Delivery Method.

Initial data:

The customer John Doe from Colby (Kansas) wants to buy two products: Item A and Item B. You have two warehouses, one is in Colorado and the other in Kansas.

Lowest Shipping

The system checks the item's shipping prices for each warehouse and suggests the combination that gives the lowest shipping cost. Let's assume each product's shipping price is static for each warehouse and doesn't vary.

Warehouse	Item A	Item B
Colorado	$1.00	$1.00
Kansas	$1.50	$1.50

Table 3. Lowest Shipping – Shipping Prices

Items	Item B: $1.00 (Colorado)	Item B: $1.50 (Kansas)
Item A: $1.00 (Colorado)	$1.00 + $1.00 = $2.00	$1.00 + $1.50 = $2.50
Item A: $1.50 (Kansas)	$1.50 + $1.00 = $2.50	$1.50 + $1.50 = $3.00

Table 4. Lowest Shipping – Summary Shipping Prices

In this instance, John Doe will get Item A and Item B at the prices of the Colorado warehouse. If, for example, Item B's shipping price in Kansas were $0.50, the system would suggest the Item A from the Colorado and the Item B from the Kansas.

Lowest Tax

If you select this delivery method option, Multi-Warehouse will set warehouses to their lowest tax basis.

Warehouse	Item A	Item B
Colorado	$2.00	$2.00
Kansas	$1.00	$1.00

Table 5. Lowest Tax – Taxes

Items	Item B: $2.00 (Colorado)	Item B: $1.00 (Kansas)
Item A: $2.00 (Colorado)	$2.00 + $2.00 = $4.00	$2.00 + $1.00 = $3.00
Item A: $1.00 (Kansas)	$1.00 + $2.00 = $3.00	$1.00 + $1.00 = $2.00

Table 6. Lowest Tax – Summary Taxes

In this case, the order will be shipped from Kansas.

Lowest Subtotal

You may sell the same product at different prices depending on the warehouse location. In the example below, you're offering discounts on Item A and Item B for the Colorado and Kansas warehouses.

Warehouse	Item A	Item B
Colorado	Price: $10.00 Discount: $0.00 Final Price: $10.00	Price: $10.00 Discount: $1.00 Final Price: $9.00
Kansas	Price: $10.00 Discount: $1.00 Final Price: $9.00	Price: $10.00 Discount: $0.00 Final Price: $10.00

Table 7. Lowest Subtotal – Prices

Items	Item B: $9.00 (Colorado)	Item B: $10.00 (Kansas)
Item A: $10.00 (Colorado)	$10.00 + $9.00 = $19.00	$10.00 + $10.00 = $20.00
Item A: $9.00 (Kansas)	$9.00 + $9.00 = $18.00	$9.00 + $10.00 = $19.00

Table 8. Lowest Subtotal – Summary Prices

The Multi-Warehouse Magento extension will specify the lowest price on every item and suggest it to the consumer (John Doe).

In this case, the system will offer:

- Shipping of the Item A from the Kansas (Final Price: $9.00)
- Shipping of the Item B from the Colorado (Final Price: $9.00)

Lowest Grand Total

The system selects products from warehouses by the lowest cart price. There are a number of factors that impact the total sale price of a product. Amongst them: tax, shipping price and discount. Each product has an individual final price according to the warehouse in which it is stored.

Warehouse	Item A	Item B
Colorado	Price $10.00 Tax $2.00 Discount $0.00 Shipping $1.00 Final Price $13.00	Price $10.00 Tax $2.00 Discount $1.00 Shipping $1.00 Final Price $12.00
Kansas	Price $10.00 Tax $1.00 Discount $1.00 Shipping $1.50 Final Price $11.50	Price $10.00 Tax $1.00 Discount $0.00 Shipping $1.50 Final Price $12.50

Table 9. Lowest Grand Total – Prices

This Multi-Warehouse extension calculates all the factors and suggests the optimal solution. In this example, that will be the lowest cart price (sum price of items A and B).

Items	Item B: $12.00 (Colorado)	Item B: $12.50 (Kansas)
Item A: $13.00 (Colorado)	$13.00 + $12.00 = $25.00	$13.00 + $12.50 = $25.50

Item A: $11.50 (Kansas)	$11.50 + $12.00 = $23.50	$11.50 + $12.50 = $24.00

Table 10. Lowest Grand Total – Summary Prices

As you can see in Table 10, the lowest cart price is $23.50. Thus, item A will be shipped from the Kansas warehouse and item B from Colorado.

Priority

You have multiple warehouses, the main warehouse could be in Colorado and the rest in states such as Kansas, New York and California.

In the administrator panel follow: Catalog -> Manage Warehouses.

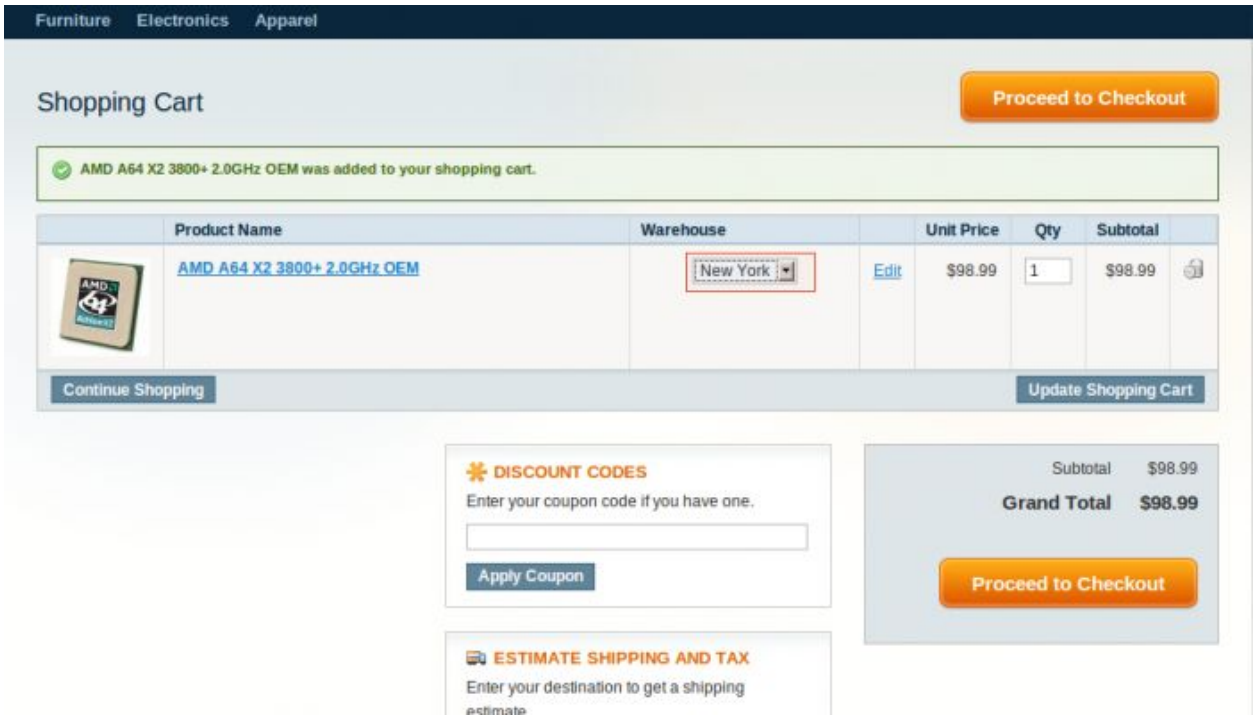
The smallest number means the highest priority. In the screenshot above, we can see the following warehouse priority:

1. Colorado
2. Kansas
3. New York
4. California

The Priority option lets you offer the customer (John Doe) the products (Item A and Item B) from the warehouses in order of priority, starting with the highest priority warehouse first. If one of the products is out of stock (Colorado), the system will offer this item from another warehouse (Kansas) according to priority and so on.

Manual Selection

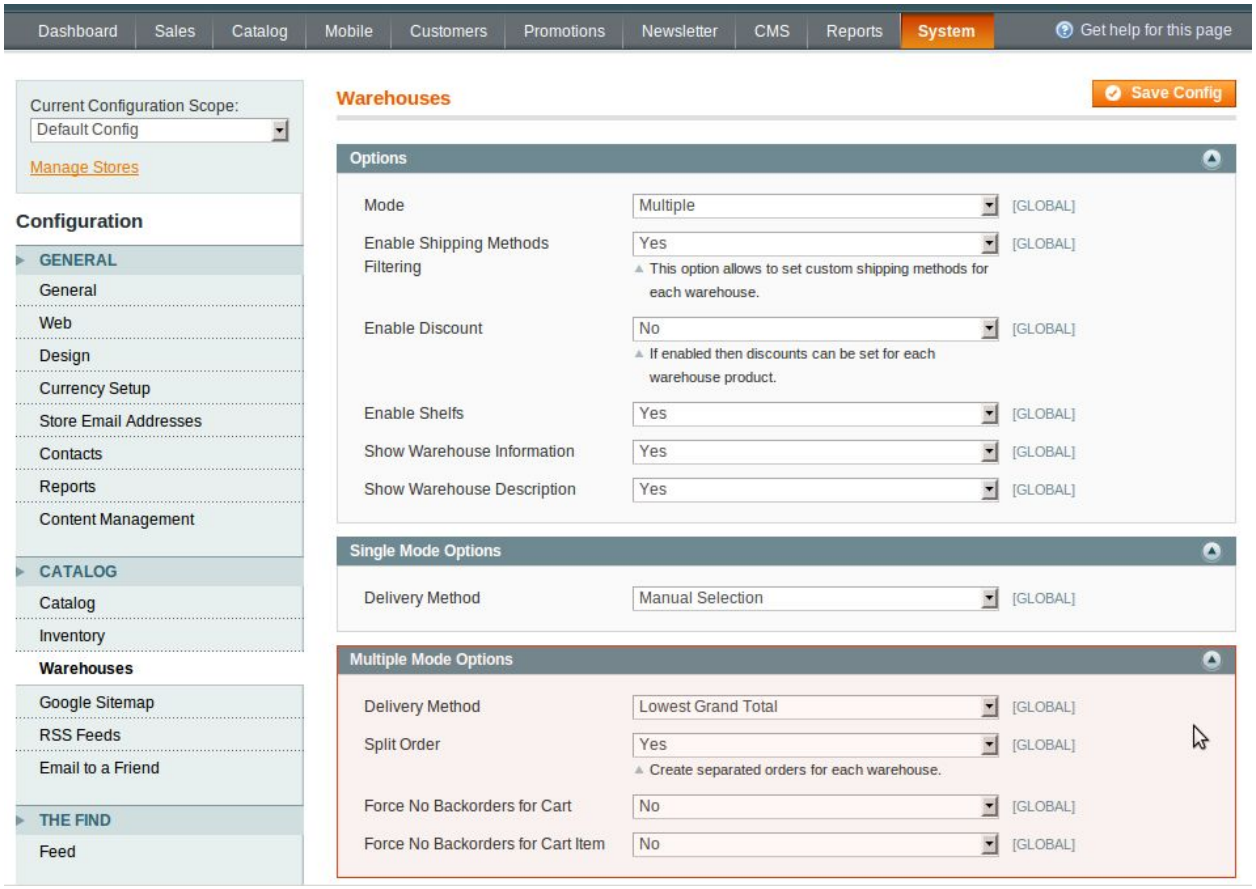
You can allow visitors to select a warehouse for each product individually.



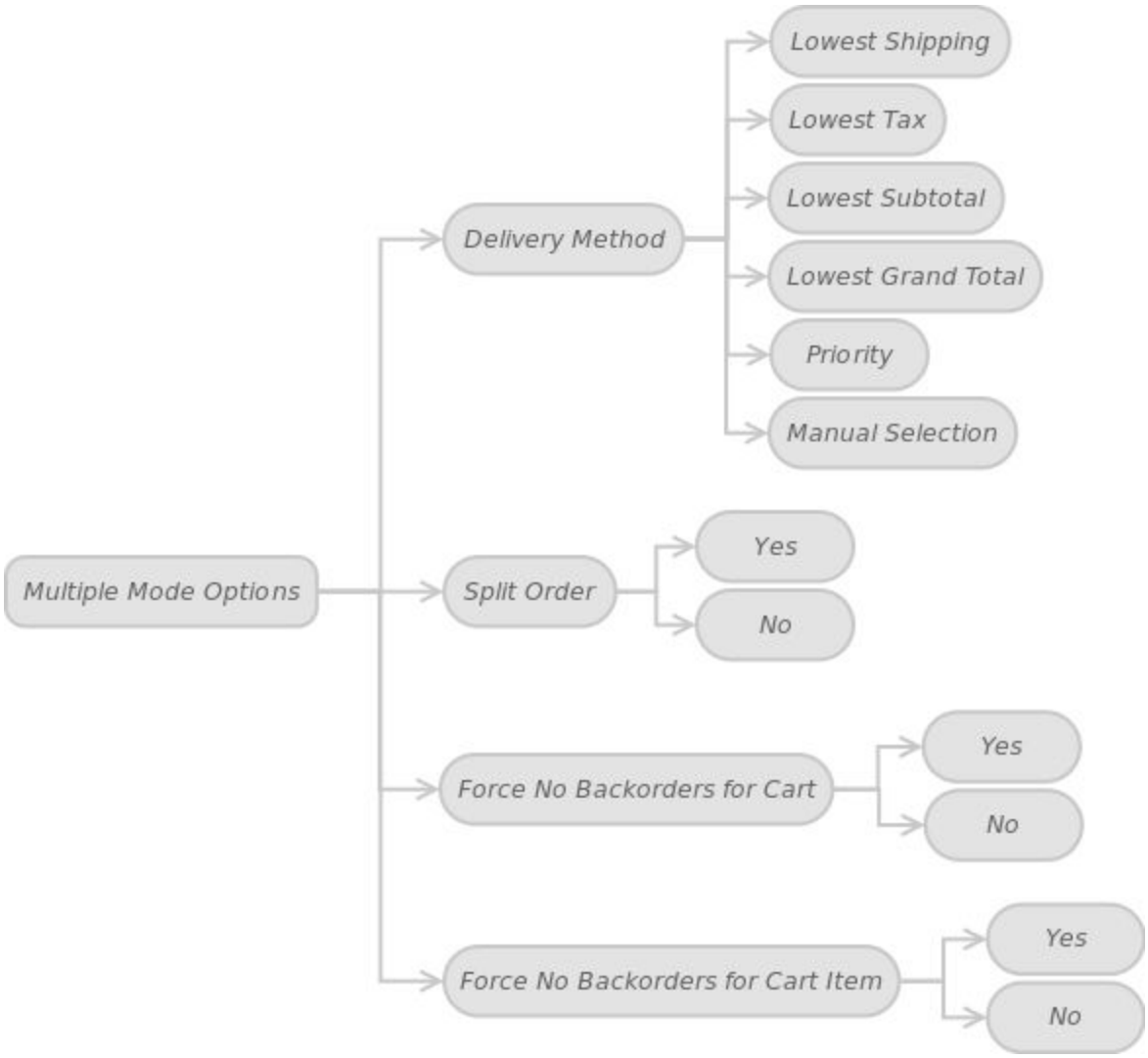
Multiple Mode

We continue our article talking about the Multiple Mode Options. In the first part of the article, we discussed the Delivery Method options. In the second part we'll consider the other options.

In administrator panel: System -> Configuration -> Catalog -> Warehouses -> Multiple Mode Options


Dashboard
Sales
Catalog
Mobile
Customers
Promotions
Newsletter
CMS
Reports
System
Get help for this page
Current Configuration Scope:
Default Config
Manage Stores
Configuration
GENERAL
General
Web
Design
Currency Setup
Store Email Addresses
Contacts
Reports
Content Management
CATALOG
Catalog
Inventory
Warehouses
Google Sitemap
RSS Feeds
Email to a Friend
THE FIND
Feed
Warehouses
Save Config
Options
Mode
Multiple
[GLOBAL]
Enable Shipping Methods Filtering
Yes
[GLOBAL]
This option allows to set custom shipping methods for each warehouse.
Enable Discount
No
[GLOBAL]
If enabled then discounts can be set for each warehouse product.
Enable Shelfs
Yes
[GLOBAL]
Show Warehouse Information
Yes
[GLOBAL]
Show Warehouse Description
Yes
[GLOBAL]
Single Mode Options
Delivery Method
Manual Selection
[GLOBAL]
Multiple Mode Options
Delivery Method
Lowest Grand Total
[GLOBAL]
Split Order
Yes
[GLOBAL]
Create separated orders for each warehouse.
Force No Backorders for Cart
No
[GLOBAL]
Force No Backorders for Cart Item
No
[GLOBAL]

We also displayed this in another form:



The customer John Doe wishes to buy two products item A and item B. The system assigns the customer to the Colorado warehouse by the Lowest Shipping delivery method. In turn, you have two warehouses: in Kansas.

Warehouse	Item A Status	Item B Status
Colorado	In-Stock	Out-of-Stock

Kansas	In-Stock	In-Stock

Table 11. – Product Statuses

Delivery Method

Read our previous article about this option.

Split Order

This option will allow you to divide an order into several orders per warehouse. Assume that a customer has two products in the shopping cart: item A in the Denver warehouse and item B in the Kansas City warehouse. If the option is enabled (Split Order equals to Yes), the system will generate two separate orders for each warehouse, the Item A order in Denver and the item B order in Kansas City. If the option is disabled (Split Order equals to No), the system will create a single (non-split) order for the products.

Force No Backorders for Cart

If you set Force No Backorders for Cart to Yes, the Multi-Warehouse doesn’t allow backordered products and in-stock items to be in the same shopping cart. Furthermore, the customer's order will filled only if all shopping cart items are available in the particular warehouse. If at least one item is unavailable in the warehouse, the system will select another accessible warehouse with products in stock. In the John Doe’s case (see initial data above), the system will allow you to order products A and B from the Kansas City warehouse only.

Force No Backorders for Cart is set to No. This option lets customers order both in-stock items and backordered items. In the example above, the system will fulfill the order from the Denver warehouse.

Force No Backorders for Cart Item

This option is similar to the previous one. The only difference is that it's applied with regards to a particular shopping cart item.

Force No Backorders for Cart Item is set to Yes. With this option you don’t allow specific products to be backordered. In the example above, the item A will be shipped to the customer from the warehouse in Denver, while the item B will be delivered from the Kansas City warehouse.

Force No Backorders for Cart Item is set to No. The system will allow customers to purchase a particular item, when it is out of stock. In this case, both the item A and B will be shipped from the Denver warehouse.

Multiple Mode Scenarios

In the given article, we'll explore some complex scenarios for Multiple Warehouse Mode Options.

Throughout this article, we’ll be referring to the initial data and tables below.

Initial data:

We have a client, John Doe from Colby, Kansas who wishes to purchase products A and B in your webshop. In turn, you gave four warehouses: Denver, Kansas City, New York and California.

Let’s represent the initial data in tabular form:

Customer	City	State	Item A quantity to buy	Item B quantity to buy

John Doe	Colby	Kansas	10	8

Table 5.1. The customer data

Warehouse	City	State	Distance between the warehouse and the customer's place of residence (Colby, Kansas)	Available quantity of the item A	Available quantity of the item B	Priority
Denver	Denver	Colorado	234 Miles / 376 Km	8	10	1
Kansas City	Kansas City	Kansas	371 Miles / 597 Km	8	6	2
New York	New York	New York	1567 Miles / 2522 Km	20	40	3
California	Sacramento	California	1405 Miles / 2260 Km	15	35	4

Table 5.2. Warehouses

Warehouse	Item A		Item B	
Denver	Price:	$10	Price:	$10
	Tax:	$2	Tax:	$2
	Discount:	$0	Discount:	$1
	Shipping Price:	$1	Shipping Price:	$1
	Total Price:	$13	Total Price:	$12
Kansas City	Price:	$10	Price:	$10
	Tax:	$1	Tax:	$1
	Discount:	$1	Discount:	$0
	Shipping Price:	$2	Shipping Price:	$2
	Total Price:	$12	Total Price:	$13
New York	Price:	$10	Price:	$10
	Tax:	$3	Tax:	$3
	Discount:	$4	Discount:	$0
	Shipping Price:	$4	Shipping Price:	$4
	Total Price:	$13	Total Price:	$17
California	Price:	$10	Price:	$10
	Tax:	$3	Tax:	$3
	Discount:	$4	Discount:	$3
	Shipping Price:	$5	Shipping Price:	$5
	Total Price:	$14	Total Price:	$15

Table 5.3. Products prices

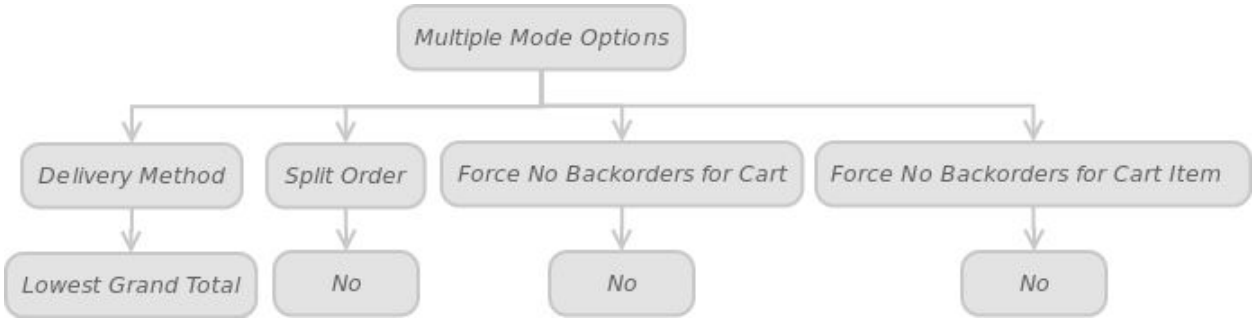
Items	Item A: $13 (Denver)	Item A: $12 (Kansas City)	Item A: $13 (New York)	Item A: $14 (California)
Item B: $12 (Denver)	$13 + $12 = $25	$12 + $12 = $24	$13 + $12 = $25	$14 + $12 = $26
Item B: $13 (Kansas City)	$13 + $13 = $26	$12 + $13 = $25	$13 + $13 = $26	$14 + $13 = $27
Item B: $17 (New York)	$13 + $17 = $30	$12 + $17 = $29	$13 + $17 = $30	$14 + $17 = $31
Item B: $15 (California)	$13 + $15 = $20	$12 + $15 = $27	$13 + $15 = $28	$14 + $15 = $29

Table 5.4. Cart price (Item A+ item B)

Warehouse	Cart price
Denver	$13 + $12 = $25
Kansas City	$12 + $13 = $25
New York	$14 + $16 = $30
California	$13 + $16 = $29

Table 5.5. Cart price if Force No Backorders for Cart is set to Yes

Scenario 1



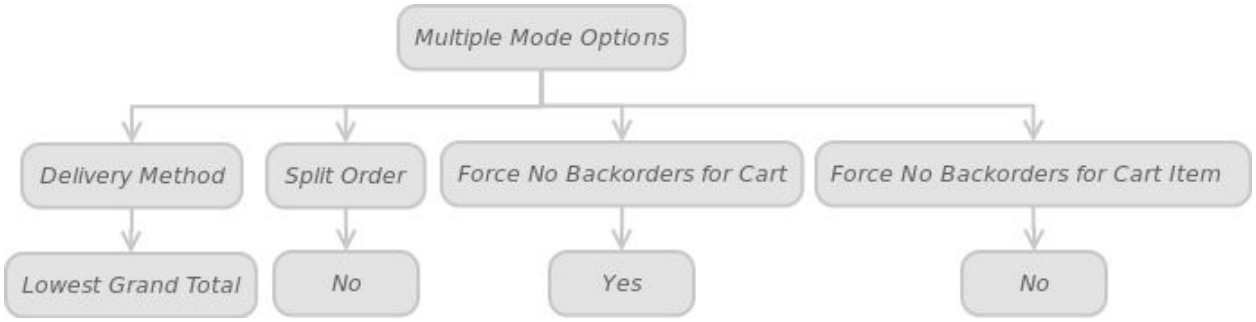
Result

Lowest Grand Total (Lowest Cart Price)	Item A delivery from the warehouse	Item B from the warehouse
$24	Kansas City	Denver

Table 5.6. Results for the scenario 1

Since Force No Backorders for Cart is set to No and Force No Backorders for Cart Item is set to No, item A will be delivered from the Kansas City warehouse, despite the fact that item A's quantity at this depot (available quantity: 8) doesn't satisfy the required conditions (required quantity: 10). The item B order will be fulfilled from the warehouse in Denver (available quantity: 10; required quantity 8).

Scenario 2



Result:

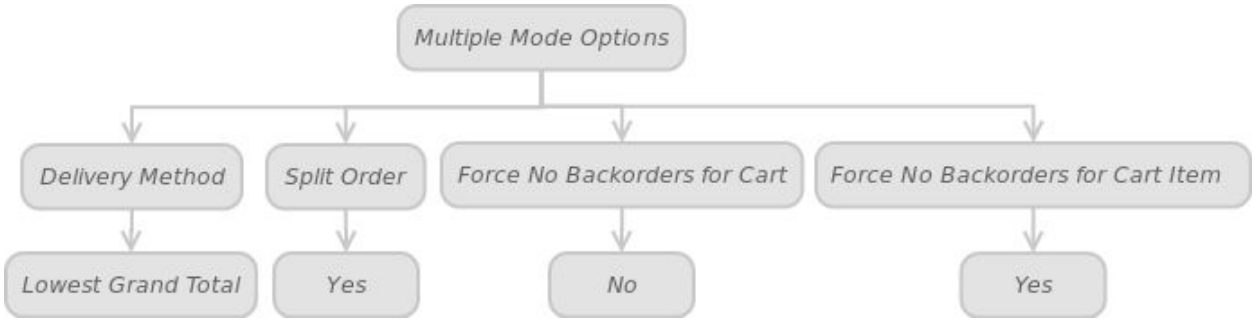
Lowest Grand Total (Lowest Cart Price)	Item A delivery from the warehouse	Item B from the warehouse
$29	California	California

Table 5.7. Results for the scenario 2

Force No Backorders for Cart implies that the order will be fulfilled from a single warehouse that has the required quantity of the items (A and B) in stock. In our case, the only two warehouses, California (item A quantity: 15, required quantity: 10; item B quantity: 35, required quantity: 8; cart price: $29) and New York (itemA quantity: 20, required quantity: 10; item B quantity: 40, required quantity: 8; cart price: 30). If the Split Order is set to No Multi-Warehouse, this will generate a single order for both items.

Note: In Force No Backorders for Cart case, no matter if the Split Order is Yes or No, the system still won't split the order.

Scenario 3



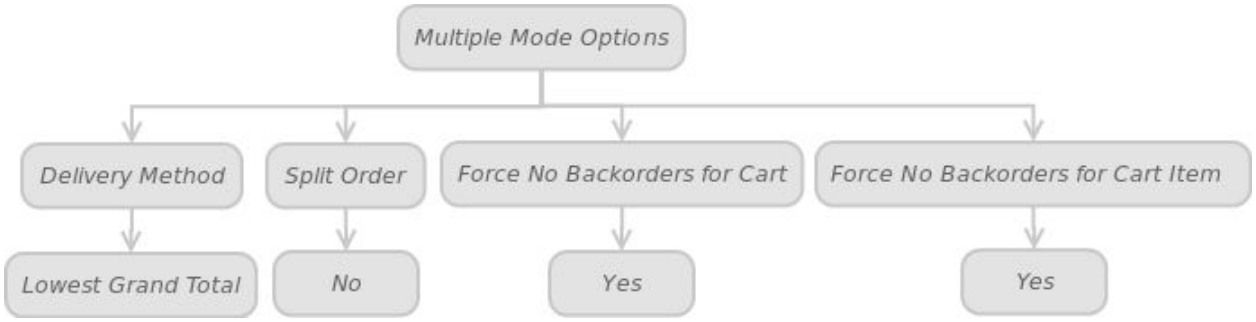
Result

Lowest Grand Total (Lowest Cart Price)	Item A delivery from the warehouse	Item B from the warehouse
$25	New York	Denver

Table 5.8. Results for the scenario 3

Let's suppose the Force No Backorders for Cart Item is set to Yes. In this case, the system will offer products (A and B) from multiple warehouses: the item A from the New York warehouse (the item A available quantity: 20; required quantity: 10) and the item B from the Denver warehouse (the item B available quantity: 10, required quantity: 8). Therefore, the cart price is $25. That would be the optimal solution for scenario 3. If the split Order is Yes, the order will be divided into two separate orders for each warehouse accordingly.

Scenario 4



Result:

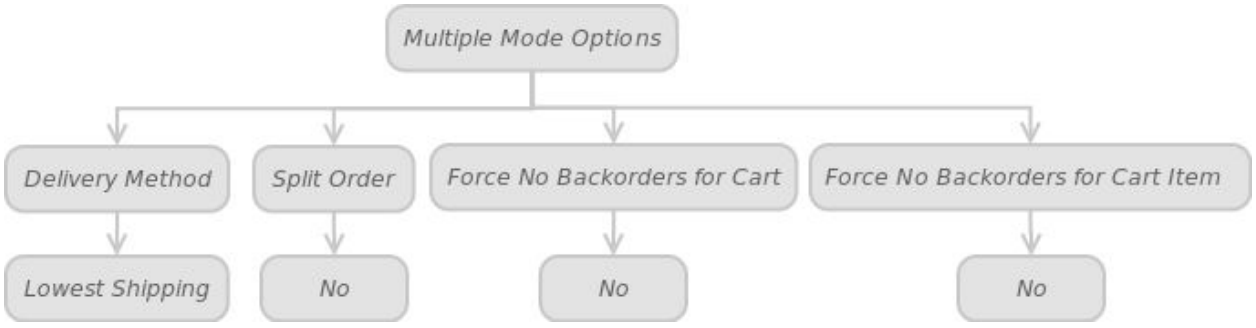
Lowest Grand Total (Lowest Cart Price)	Item A delivery from the warehouse	Item B from the warehouse
$29	California	California

Table 5.9. Results for the scenario 4

Note: In this case, even if the Force No Backorders for Cart Item is Yes or No, the order still will be delivered from a single warehouse.

Now let’s take a brief look at the following scenarios.

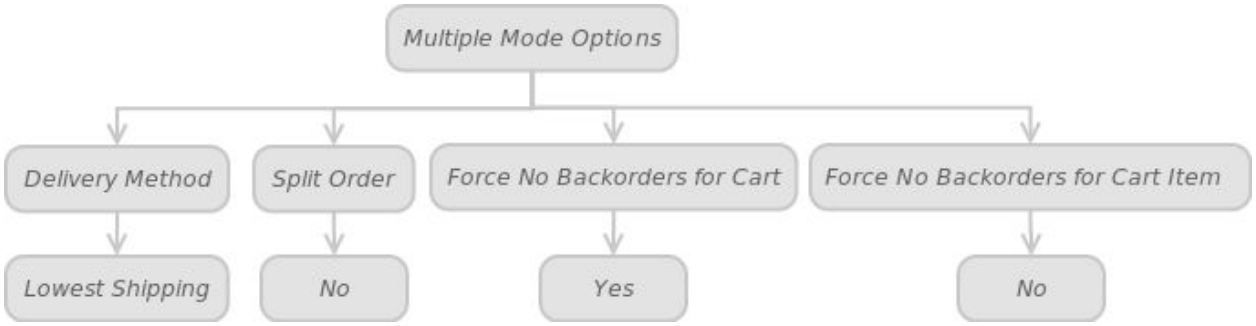
Scenario 5



Result:

Item A		Item B	
Delivery from the warehouse	**Shipping price**	**Delivery from the warehouse**	**Shipping price**
Denver	$1	Denver	$1

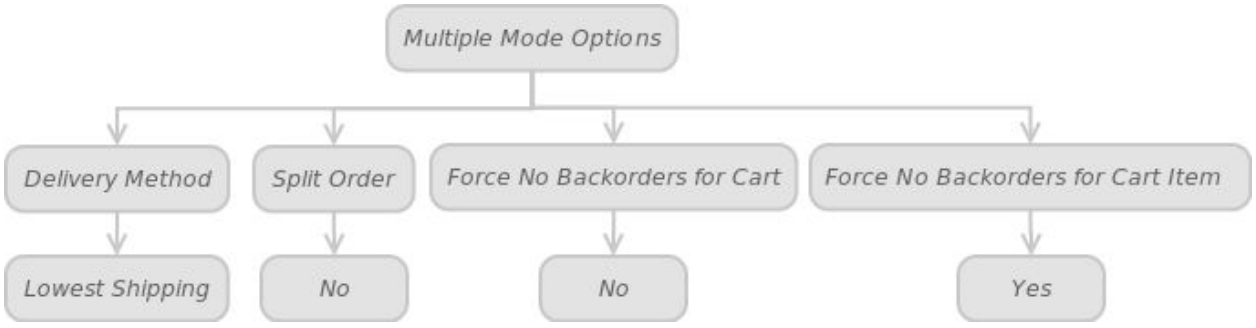
Scenario 6



Result:

Table 5.11. Results for the scenario 6

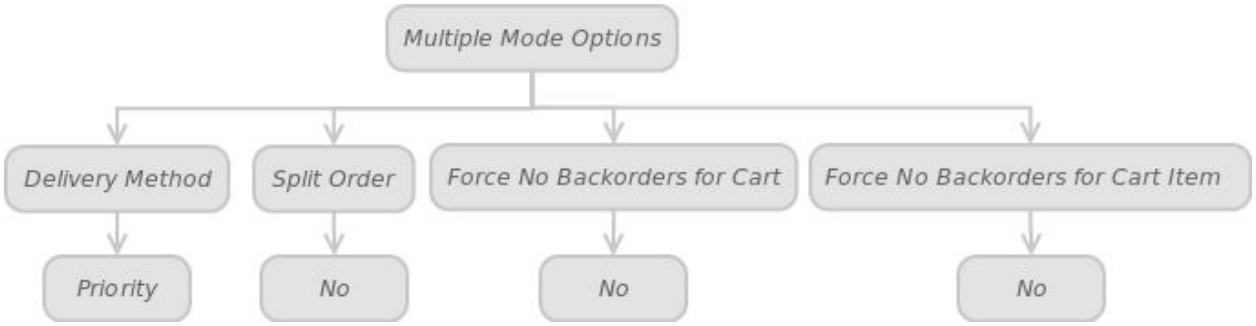
Scenario 7



Result:

Table 5.12. Results for the Scenario 7

Scenario 8

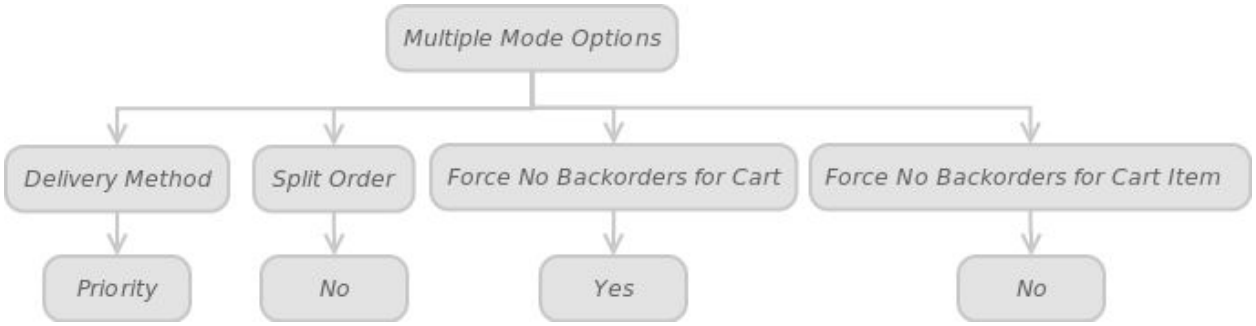


Result:

Item A delivery from the warehouse	Item B delivery from the warehouse
Denver	Denver

Table 5.13. Results for the scenario 8

Scenario 9

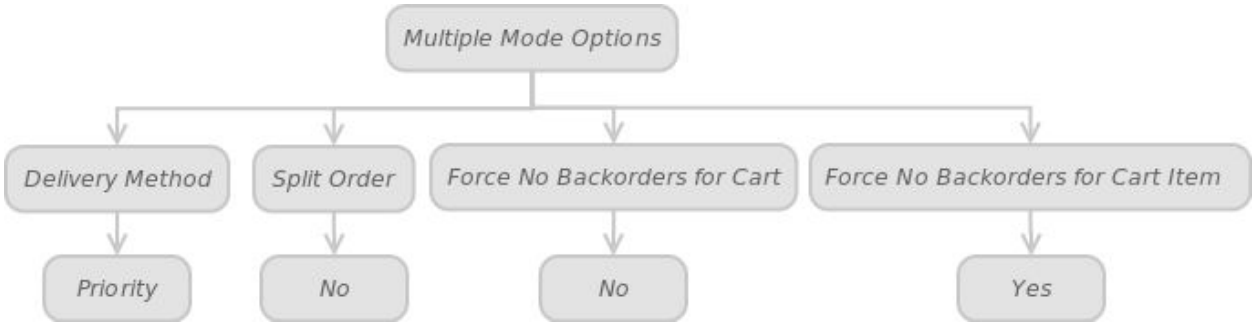


Result:

Item A delivery from the warehouse	Item B delivery from the warehouse
New York	New York

Table 5.14. Results for the scenario 9

Scenario 10



Result:

Item A delivery from the warehouse	Item B delivery from the warehouse
New York	Denver

Table 5.15. Results for the scenario 10

Products

In administrator panel: Catalog -> Manage Products

You can simultaneously see the quality of an individual item on hand at a particular warehouse and the total quantity of this item.

Inventory

Edit the details of a particular product spread across multiple warehouses in one table.

In administrator panel: Catalog -> Manage Product -> Edit -> Inventory

Shelf Information

This feature let's you quickly find any individual item in any of your warehouses. (Which bay? Which shelf? etc).

In administrator panel: Catalog -> Manage Product -> Edit -> Shelf Information

Just click Add Shelf and indicate where in the warehouse the product can be found.

rice Adjustment

Establish different discounts on the same item depending on location (warehouse).

Admin panel: Catalog -> Manage Product -> Edit -> Prices -> Price Adjustment

In the column Apply, you need to select the discount percentage or fixed amount. In the column Discount Amount, type the value that you want.

Sales

Admin panel: Sales

In turn,

Each of the tables shown in Figure 8.1. has a Warehouse column that let's you filter your orders, invoices, shipments and credit memos by a particular warehouse. Sort the data by selecting a required warehouse from the dropdown list.

Low Stock Report

Admin Panel: Reports -> Products -> Low Stock

The system indicates which product's inventory is running low and in which warehouse. Select a warehouse and the amount per product you wish your Magento store to regard as a low quantity amount. If this quantity amount is reached, the Multi-Warehouse will display all items that have fallen below the minimum stock level, and indicate the warehouse in which these items are stored.

Manage Shipping Table Rates

Import Export

The Multi-Warehouse extension extends the Magento Dataflow module in order to handle product stock importing / exporting for each warehouse individually.

You will need to create a separated advanced profile to import / export product stocks for each warehouse. Let's suppose we need profiles for the Kansas warehouse. Kansas' identifier equals 2.

In the administrator panel: System -> Import/Export -> Dataflow – Advanced Profiles.

Enter the next data for the profile:

The Profile Name is "Export Kansas Product Stocks"

Actions XML

```
<action type="catalog/convert_adapter_product" method="load">
                                        <var name="store"><![CDATA[0]]></var>
                                        <var name="warehouse"><![CDATA[2]]></var>
                                    </action>
                                    <action type="catalog/convert_parser_product"
method="unparse">
                                        <var name="store"><![CDATA[0]]></var>
                                        <var name="warehouse"><![CDATA[2]]></var>
                                    </action>
                                    <action type="dataflow/convert_mapper_column"
method="map">
                                        <var name="map">
                                            <map
name="store"><![CDATA[store]]></map>
                                            <map name="sku"><![CDATA[sku]]></map>
                                            <map name="qty"><![CDATA[qty]]></map>
                                            <map
name="is_in_stock"><![CDATA[is_in_stock]]></map>
                                        </var>
                                        <var name="_only_specified">true</var>
                                    </action>
                                    <action type="dataflow/convert_parser_csv"
```

```
method="unparse">
                                        <var name="delimiter"><![CDATA[,]]></var>
                                        <var name="enclose"><![CDATA["]]></var>
                                        <var name="fieldnames">true</var>
                                    </action>
                                    <action type="dataflow/convert_adapter_io"
method="save">
                                        <var name="type">file</var>
                                        <var name="path">var/export</var>
                                        <var
name="filename"><![CDATA[export_kansas_product_stocks.csv]]></var>
                                    </action>
```

Importing the profile may look like this:

The Profile Name is "Import Kansas Product Stocks"

Actions XML

```
<action type="dataflow/convert_adapter_io" method="load">
                                    <var name="type">file</var>
                                    <var name="path">var/import</var>
                                    <var
name="filename"><![CDATA[import_kansas_product_stocks.csv]]></var>
                                    <var name="format"><![CDATA[csv]]></var>
                                </action>
                                <action type="dataflow/convert_parser_csv"
method="parse">
                                    <var name="delimiter"><![CDATA[,]]></var>
                                    <var name="enclose"><![CDATA["]]></var>
                                    <var name="fieldnames">true</var>
                                    <var name="store"><![CDATA[0]]></var>
                                    <var
name="decimal_separator"><![CDATA[.]]></var>
                                    <var
name="adapter">catalog/convert_adapter_product</var>
                                    <var name="warehouse"><![CDATA[2]]></var>
                                </action>
```

API

On this page, we will describe how to manage product stocks for each warehouse with SOAP API remotely. You will need an API user with approprite permissions to be created first.

In the administrator panel: System -> Web Services -> Roles

Let's create the inventory manager role here. Enter the next data:

- Role Name – Warehouse Manager
- Resource Access – Custom
- Resources – check Catalog Inventory

Save the role and then type: System -> Web Services -> Users

The User should be created here:

- User Name – multiwarehouse
- First Name – Warehouse
- Last Name – Manager
- Email – multiwarehouse@innoexts.com
- API Key – multiwarehouse1
- User Role – Warehouse Manager

Save user and you're finished.

List Product Stocks

Save the next php script, set the variables and correct it according to your needs and run it:

```
<?php
                                # API URL. Replace domain name with yours
                                $apiUrl           =
'http://demo.merchantprotocol.com/M1-warehouse/api/soap/?wsdl';
```

```
                                    # API Username
                                    $apiUsername      = 'multiwarehouse';
                                    # API Key
                                    $apiPassword      = 'multiwarehouse1';
                                    # Product SKUs to list
                                    $productSkus      = array('amda64', 'intelc2d');
                                    # Stock identifier. 3 is New York in our case
                                    $stockId          = 3;

                                    $soapClient = new SoapClient($apiUrl);
                                    $sessionId = $soapClient->login($apiUsername,
$apiPassword);
                                    $responce = $soapClient->call($sessionId,
'product_stock.listByStock', array($productSkus, $stockId));

                                    print_r($responce);

                                    /*
                                     Responce:
                                     Array
                                     (
                                         [0] => Array
                                             (
                                                 [product_id] => 148
                                                 [sku] => amda64
                                                 [qty] => 100.0000
                                                 [is_in_stock] => 1
                                                 [stock_id] => 3
                                             )
                                         [1] => Array
                                             (
                                                 [product_id] => 151
                                                 [sku] => intelc2d
                                                 [qty] => 0
                                                 [is_in_stock] => 0
                                                 [stock_id] => 3
                                             )
                                     )
                                    */
                                    ?>
```

Update Product Stocks

Save the next php script, set variables, correct it according to your needs and run it:

```
<?php
                                # API URL. Replace domain name with yours
                                $apiUrl          =
'http://multiwarehouse.innoexts.com/api/soap/?wsdl';
                                # API Username
                                $apiUsername     = 'multiwarehouse';
                                # API Key
                                $apiPassword     = 'multiwarehouse1';
                                # Product SKUs to update
                                $productSku      = 'amda64';
                                # Stock item data
                                $stockItem       = array('qty' => 105);
                                # Stock identifier. 3 is New York in our case
                                $stockId         = 3;

                                $soapClient = new SoapClient($apiUrl);
                                $sessionId = $soapClient->login($apiUsername,
$apiPassword);
                                $soapClient->call($sessionId,
'product_stock.updateByStock', array($productSku, $stockItem, $stockId));
                                ?>
```

List Product Stocks V2

Save the next php script, set variables, correct according to your needs and run it:

```
<?php
                               # API URL. Replace domain name with yours
                               $apiUrl          =
'http://multiwarehouse.innoexts.com/api/v2_soap/?wsdl';
                               # API Username
                               $apiUsername     = 'multiwarehouse';
                               # API Key
                               $apiPassword     = 'multiwarehouse1';
                               # Product SKUs to list
                               $productSkus     = array('amda64', 'intelc2d');
                               # Stock identifier. 3 is New York in our case
                               $stockId         = 3;
```

```
$soapClient = new SoapClient($apiUrl, array('trace' => 1));
$sessionId = $soapClient->login($apiUsername, $apiPassword);
$responce = $soapClient->catalogInventoryStockItemListByStock($sessionId, $productSkus, $stockId);

print_r($responce);

/*
Responce:
Array
(
    [0] => stdClass Object
        (
            [product_id] => 148
            [sku] => amda64
            [qty] => 105.0000
            [is_in_stock] => 1
            [stock_id] => 3
        )
    [1] => stdClass Object
        (
            [product_id] => 151
            [sku] => intelc2d
            [qty] => 0
            [is_in_stock] => 0
            [stock_id] => 3
        )
)
*/
?>
```

Update Product Stocks V2

Save the next php script, set variables, correct according to your needs and run it:

```
<?php
# API URL. Replace domain name with yours
$apiUrl          = 'http://multiwarehouse.innoexts.com/api/v2_soap/?wsdl';
# API Username
$apiUsername     = 'multiwarehouse';
# API Key
$apiPassword     = 'multiwarehouse1';
```

```
                                # Product SKUs to update
                                $productSku      = 'amda64';
                                # Stock item data
                                $stockItem       = array('qty' => 110);
                                # Stock identifier. 3 is New York in our case
                                $stockId         = 3;

                                $soapClient = new SoapClient($apiUrl, array('trace'
=> 1));
                                $sessionId = $soapClient->login($apiUsername,
$apiPassword);

$soapClient->catalogInventoryStockItemUpdateByStock($sessionId, $productSku,
$stockItem, $stockId);
                                ?>
```