spreecommerce™
BEHIND THE BEST STOREFRONTS™

Orders - Developer Guide | Spree Commerce

Overview

The `Order` model is one of the key models in Spree. It provides a central place around which to collect information about a customer order - including line items, adjustments, payments, addresses, return authorizations, and shipments.

Orders have the following attributes:

- `number`: The unique identifier for this order. It begins with the letter R and ends in a 9-digit number. This number is shown to the users, and can be used to find the order by calling `Spree::Order.find_by_number(number)`.
- `item_total`: The sum of all the line items for this order.
- `adjustment_total`: The sum of all adjustments on this order.
- `total`: The result of the sum of the `item_total` and the `adjustment_total`.
- `state`: The current state of the order. To read more about the states an order goes through, read The Order State Machine section of this guide.
- `email`: The email address for the user who placed this order. Stored in case this order is for a guest user.
- `user_id`: The ID for the corresponding user record for this order. Stored only if the order is placed by a signed-in user.
- `completed_at`: The timestamp of when the order was completed.
- `bill_address_id`: The ID for the related `Address` object with billing address information.
- `ship_address_id`: The ID for the related `Address` object with shipping address information.
- `shipment_state`: The current shipment state of the order. For possible states, please see the Shipments guide.
- `payment_state`: The current payment state of the order. For possible states, please see the Payments guide.
- `special_instructions`: Any special instructions for the store to do with this order. Will only appear if `Spree::Config[:shipping_instructions]` is set to `true`.
- `currency`: The currency for this order. Determined by the `Spree::Config[:currency]` value that was set at the time of order.
- `last_ip_address`: The last IP address used to update this order in the frontend.

Some methods you may find useful:

- `outstanding_balance`: The outstanding balance for the order, calculated by taking the `total` and subtracting `payment_total`.
- `display_item_total`: A "pretty" version of `item_total`. If `item_total` was `10.0`, `display_item_total` would be `$10.00`.
- `display_adjustment_total`: Same as above, except for `adjustment_total`.
- `display_total`: Same as above, except for `total`.
- `display_outstanding_balance`: Same as above, except for `outstanding_balance`.

The Order State Machine

Orders flow through a state machine, beginning at a `cart` state and ending up at a `complete` state. The intermediary states can be configured using the Checkout Flow API.

The default states are as follows:

- `cart`
- `address`
- `delivery`
- `payment`
- `confirm`
- `complete`

The `payment` state will only be triggered if `payment_required?` returns `true`.

The `confirm` state will only be triggered if `confirmation_required?` returns `true`.

The `complete` state can only be reached in one of two ways:

1. No payment is required on the order.
2. Payment is required on the order, and at least the order total has been received as payment.

Assuming that an order meets the criteria for the next state, you will be able to transition it to the next state by calling `next` on that object. If this returns `false`, then the order does *not* meet the criteria. To work out why it cannot transition, check the result of an `errors` method call.

Line Items

Line items are used to keep track of items within the context of an order. These records provide a link between orders, and Variants.

When a variant is added to an order, the price of that item is tracked along with the line item to preserve that data. If the variant's price were to change, then the line item would still have a record of the price at the time of ordering.

- Inventory tracking notes

**************** TODO ****************

Update this section after Chris+Brian have done their thing.

****

Addresses

An order can link to two `Address` objects. The shipping address indicates where the order's product(s) should be shipped to. This address is used to determine which shipping methods are available for an order.

The billing address indicates where the user who's paying for the order is located. This can alter the tax rate for the order, which in turn can change how much the final order total can be.

For more information about addresses, please read the Addresses guide.

Adjustments

Adjustments are used to affect an order's final cost, either by decreasing it (Promotions) or by increasing it (Shipping, Taxes).

For more information about adjustments, please see the Adjustments guide.

Payments

Payment records are used to track payment information about an order. For more information, please read the Payments guide.

Return Authorizations

**************** TODO ****************

document return authorizations.

****

OrderPopulator

**************** TODO ****************

Add documentation about the OrderPopulator class here

****

Updating an Order

If you change any aspect of an `Order` object within code and you wish to update the order's totals – including associated adjustments and shipments – call the `update!` method on that object, which calls out to the `OrderUpdater` class.

For example, if you create or modify an existing payment for the order which would change the order's `payment_state` to a different value, calling `update!` will cause the `payment_state` to be recalculated for that order.

Another example is if a `LineItem` within the order had its price changed. Calling `update!` will cause the totals for the order to be updated, the adjustments for the order to be recalculated, and then a final total to be established.

Product

- Terms of Service

Developers

- Overview
- Documenation
- Community
- License

Partners

- Pro Services
- Training
- Partnership Program
- Payments

About Spree Commerce

Spree Commerce is an automated enterprise solution built specifically for ecommerce. We effectively manage your operations so you can focus on serving your customers and growing your business.

Take it for a test drive

You can even create your own personal store.

Try The Demo