## ## Introduction

This guide will teach you how to use a combination of the product, Spec, and SpecOption resources to create a configurable product with conditional pricing based on the options selected by the buyer user. This is often referred to as <em>Configure, Price, Quote</em> in commerce terms.

The example we will walk through is offering a product where the user can select their type of bindings on a Notebook, as well as choose whether or not they would like an engraving on the cover of the Notebook. Based on their selection, the price will be different.

## ## Prerequisites

* [Create and Assign Users and User Groups](https://documentation.ordercloud.io/use-case-guides/buyer-and-seller-organization-management/create-and-assign-users-and-user-groups)
* [Create and Assign a Product](<https://documentation.ordercloud.io/use-case-guides/product-catalog-management/create-and-assign-a-product>)
* [Make an Assignment]([https://documentation.ordercloud.io/use-case-guides/product-catalog-management/make-an-assignment](https://documentation.ordercloud.io/product-catalog-management/make-an-assignment))

## 1. Create Specs

For this scenario, we will create two Specs: one Spec will represent the choice of bindings, and one to represent the choice of engraving.

<div class="api-reference">API Reference: [Create a Spec](<https://documentation.ordercloud.io/api-reference#Specs_Create>)</div>

```

POST https://api.ordercloud.io/v1/specs HTTP/1.1

Authentication: Bearer put\_access\_token\_here

Content-Type: application/json; charset=UTF-8

{

"ID": "Leatherbound",

"ListOrder": 1,

"Name": "Leatherbound",

"DefaultValue": null,

"Required": true,

"AllowOpenText": false,

"DefaultOptionID": null,

"DefinesVariant": false,

"xp": null

}

```

<div class="api-reference">API Reference: [Create a Spec](<https://documentation.ordercloud.io/api-reference#Specs_Create>)</div>

```

POST https://api.ordercloud.io/v1/specs HTTP/1.1

Authentication: Bearer put\_access\_token\_here

Content-Type: application/json; charset=UTF-8

{

"ID": "Engraved",

"ListOrder": 2,

"Name": "Engraved",

"DefaultValue": null,

"Required": true,

"IsOpenText": true,

"DefaultOptionID": null,

"DefinesVariant": false,

"xp": null

}

```

<div class="note">`DefaultValue` and `DefaultOptionID` should be left off of the request body or left null when creating a new Spec since there are no options at this point in time.</div>

## 2. Create SpecOptions for Your Specs

In this step, we will create options under both of our Specs that will represent the selections customers will be able to make on the storefront. Using our Notebook example, the `Leatherbound` Spec and the `Engraved` Spec will have two SpecOptions to choose from.

Starting with the `Engraved` SpecOption, we will declare `"IsOpenText": true` so the user can enter custom text to be engraved on the Notebook. In our example, an `Engraved` Notebook has a markup; therefore, we will define the `PriceMarkupType` and `PriceMarkup` values for this Spec.

Refer to the following table to understand the values available for `PriceMarkupType`:

<table class="table table-bordered table-responsive">

<tr>

<th>Value</th>

<th>Definition</th>

<th>Use Case</th>

</tr>

<tr>

<td>`NoMarkup` or null</td>

<td><em>This will have no effect on `PriceMarkup`.</em></td>

<td></td>

</tr>

<tr>

<td>`AmountTotal`</td>

<td>The integer value saved in `PriceMarkup` will be applied to the line item total one time, regardless of quantity.</td>

<td>This is common in print production for creating a one-time setup fee. A user could order 25 shirts but only pay one

$50 setup fee for the entire order.</td>

</tr>

<tr>

<td>`AmountPerQuantity`</td>

<td>The integer value saved in `PriceMarkup` will be multiplied by the quantity purchased and applied to the line total.</td>

<td>Since each `Engraved` Notebook in our example incurs an additional charge, we will use the `AmountPerQuantity` value

to add a markup on each item ordered.</td>

</tr>

<tr>

<td>`Percentage`</td>

<td>The integer value saved in `PriceMarkup` represents the percentage increase/decrease of the line item total.</td>

<td>The integer value saved in `PriceMarkup` represents the percentage increase/decrease of the line item total.</td>

</tr>

</table>

<div class="note">A negative integer in `PriceMarkup` would decrease the price per quantity, once in total or by a percentage as appropriate depending on the `PriceMarkupType` value selected.</div>

<div class="api-reference">API Reference: [Create a Spec Option](<https://documentation.ordercloud.io/api-reference#SpecOption_Create>) </div>

```

POST https://api.ordercloud.io/v1/specs/Leatherbound/options HTTP/1.1

Authentication: Bearer put\_access\_token\_here

Content-Type: application/json; charset=UTF-8

{

"ID": "engraved-yes",

"Value": true,

"ListOrder": 1,

"AllowOpenText": true,

"PriceMarkupType": “AmountPerQuantity”,

"PriceMarkup": 5.00,

"xp": null

}

```

```

POST https://api.ordercloud.io/v1/specs/Leatherbound/options HTTP/1.1

Authentication: Bearer put\_access\_token\_here

Content-Type: application/json; charset=UTF-8

{

"ID": "engraved-no",

"Value": false,

"ListOrder": 0,

"IsOpenText": false,

"PriceMarkupType": null,

"PriceMarkup": null,

"xp": null

}

```

Since there is no markup or open text field for the `Leatherbound` Notebook option, we will keep the `"PriceMarkupType": null` and `"IsOpenText": false`:

<div class="api-reference">API Reference: [Create Spec Option](<https://documentation.ordercloud.io/api-reference#Specs_CreateOption>)</div>

```

POST https://api.ordercloud.io/v1/specs/Leatherbound/options HTTP/1.1

Authentication: Bearer put\_access\_token\_here

Content-Type: application/json; charset=UTF-8

{

"ID": "Leatherbound",

"Value": false,

"ListOrder": 0,

"IsOpenText": false,

"PriceMarkupType": null,

"PriceMarkup": null,

"xp": null

}

```

```

POST https://api.ordercloud.io/v1/specs/Leatherbound/options HTTP/1.1

Authentication: Bearer put\_access\_token\_here

Content-Type: application/json; charset=UTF-8

{

"ID": "Leatherbound",

"Value": true,

"ListOrder": 0,

"IsOpenText": false,

"PriceMarkupType": null,

"PriceMarkup": null,

"xp": null

}

```

## ## 3. Assign Specs to Your Product

The final step is to assign the Spec to the product, which will enable the `SpecOptions` when a line item is created from that product. Continuing our previous example, we will assign both the `Engraved` and `Leatherbound` Specs to our `Notebook`. We can also use assignments to designate the default option and values that should be displayed for that specific product.

<div class="api-reference">API Reference: [Assign a Spec to a Product](https://documentation.ordercloud.io/api-reference#Specs\_SaveProductAssignment)</div>

```

POST https://api.ordercloud.io/v1/specs/productassignments HTTP/1.1

Authentication: Bearer put\_access\_token\_here

Content-Type: application/json; charset=UTF-8

{

"SpecID": "Leatherbound",

"ProductID": "Notebook",

"DefaultValue": null,

"DefaultOptionID": "No"

}

```

<div class="api-reference">API Reference: [Assign a Spec to a Product](https://documentation.ordercloud.io/api-reference#Specs\_SaveProductAssignment)</div>

```

POST https://api.ordercloud.io/v1/specs/productassignments HTTP/1.1

Authentication: Bearer put\_access\_token\_here

Content-Type: application/json; charset=UTF-8

{

"SpecID": "Engraved",

"ProductID": "Notebook",

"DefaultValue": “”,

"DefaultOptionID": "No"

}

```

<div class="note">`DefaultValue` on the `Engraved` Spec to `Notebook` product assignment is an empty string so that when the user selects this option, they start with a blank field to enter their desired engraving text.</aside>

## ## Conclusion

By following this guide, you should have a starting point for creating even more complex products with a variety of SpecOptions and markups. Remember that the combination of Specs and SpecOptions will allow you to present specific customizations and configurations on whichever products you’d like. Furthermore the configurations based on the SpecOptions applied in the platform can be presented any number of ways in your own custom buyer applications to meet whatever business requirement.