Labels

You can customize the labels and messages that shoppers see on your storefront by editing your labels file. The Core theme includes a default list of labels in the `labels/en-US.json` file, but you can also add your own custom labels to the file. While some of the labels in the labels file are exposed on your storefront by default, many are not. These labels are simply suggestions for common information that you may choose to expose on your site.

The labels file contains a list of variables with associated string values. You can access the label variables within any Hypr template in your theme, be it for widgets, page templates, or email templates. When your theme renders on your storefront, it displays the label values wherever you have exposed label variables in your templates.

Create or Modify a Label

You can easily modify existing labels or create new labels of your own. In this example, you modify the default Out of Stock message and create a new label to alert shoppers when an item's quantity is running low:

1. Open the `labels/en-US.json` file for editing.
2. Locate the existing `"outOfStock"` variable and modify the default message to your liking, as shown in the following code block:

```
...
   "outOfStock": "Insert your own OOO message here...",
...
```

3. Create a new variable called `"quantityLow"` and provide a message that lets shoppers know that there aren't many items left, as shown in the following code block:

```
...
   "quantityLow": "Less than 10 units left!",
...
```

4. Save the labels file.

Expose a Label in a Template

You can expose labels in any Hypr template by using the `labels` variable. The syntax is `{{ labels.`*`labelVariableName`*` }}`. The following steps continue the example from the previous section and outline how to expose labels in a Hypr template:

1. The `"outOfStock"` labels is exposed by default in the Kibo eCommerce Core theme, so you don't need to take additonal steps for your modified message to appear when a product is out of stock. To verify this, you can search your theme files for the variable name. The label is used in the `templates/modules/location/location-listing-for-product.hypr.live` template:

```
...
   {% if model.quantity == 0 %}
      {{ labels.outOfStock }}
   {% else %}
      {{ labels.availableNow }}
...
```

Not every label included in the labels file is exposed in a template by default. Some of the included variables in the `en-US.json` file are not exposed, and you need to expose them yourself if you plan to use them.

2. To expose the new `"quantityLow"` label, add it to the template using the `labels` variable, as shown in the following code block:

```
...
   {% if model.quantity == 0 %}
      {{ labels.outOfStock }}
   {% else %}
      {% if model.quantity <= 10 %}
         {{ labels.quantityLow }}
      {% else %}
         {{ labels.availableNow }}
...
```

Use Placeholder Values in Label Messages

It is often useful to apply placeholder values in your label messages when you want to display a variable value to shoppers. To include a placeholder value in a label, use the `{n}` notation, and then use the string_format filter to apply a value to the label within the template itself.

For example, the `"milesAway"` label uses a placeholder value to display the distance to a location, as shown in the following code blocks:

In the *en-US.json* file:

```
...
   "milesAway": "{0} miles away",
...
```

In the *templates/modules/location/* and *location-listing.hypr.live* files:

```
...
   {{ labels.milesAway|string_format(location.distance) }}
...
```

Use Multiple Placeholder Values in Label Messages

If you wanted to include more than one placeholder value, you could add additional placeholder numbers. For example:

```
...
   "milesAway": "{0} miles away or {1} kilometers away",
...
```

You then provide the values in the template according to how they are numbered in the labels file:

```
...
   {{ labels.milesAway|string_format(location.distance, location.distance * 1.6) }}
...
```