

SHOPKOIN – Interview Test

Introduction

The following pages give you a visual idea and explanation of the objective of the code you will deal with.

We will focus on the ProductPriceBlock

When there is no “price” definition (the JSON is empty or does not exist), it simply displays “No price definition”

Price definition
















[EDIT](#)

Region / Country	Price	Normal	VAT Rate(%)
No price definition			

When the user clicks “EDIT”, the block expands and shows a hierarchical list of countries as follows:


Price definition

[SAVE](#) [CANCEL](#)

Region / Country	Price	Normal	VAT Rate(%)	
<input type="checkbox"/> Europe	<input type="text"/>	<input type="text"/>	<input type="text"/>	
<input type="checkbox"/> Austria	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> Belgium	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> Bulgaria	<input type="text" value="?"/>	<input type="text" value="?"/>	<input type="text"/>	
<input type="checkbox"/> Switzerland	<input type="text" value="?"/>	<input type="text" value="?"/>	<input type="text"/>	
<input type="checkbox"/> Cyprus	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> Czech Republic	<input type="text" value="?"/>	<input type="text" value="?"/>	<input type="text"/>	
<input type="checkbox"/> Denmark	<input type="text" value="?"/>	<input type="text" value="?"/>	<input type="text"/>	
<input type="checkbox"/> Germany	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> Estonia	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> Greece	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> Spain	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> Finland	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> France	<input type="text" value="€"/>	<input type="text" value="€"/>	<input type="text"/>	
<input type="checkbox"/> Croatia	<input type="text"/>	<input type="text"/>	<input type="text"/>	





The user can input values for any of the listed countries.

In order to be able to edit the price for multiple countries at once, the user can toggle the checkboxes of several countries under the condition that they share the same currency.


<input checked="" type="checkbox"/> Germany	<input type="text"/>	€	<input type="text"/>	€	<input type="text"/>	
<input checked="" type="checkbox"/> Estonia	<input type="text"/>	€	<input type="text"/>	€	<input type="text"/>	
<input checked="" type="checkbox"/> Greece	<input type="text"/>	€	<input type="text"/>	€	<input type="text"/>	
<input checked="" type="checkbox"/> Spain	<input type="text"/>	€	<input type="text"/>	€	<input type="text"/>	

When the click one of the Text Fields related to the selected countries, if multiple are selected, a popup is shown to the user. The user then can enter information that will be “replicated” to all the selected countries.

Price definition

<input type="checkbox"/> Germany	<input type="text" value="15"/>	↕	€	<input type="text" value="15"/>	€	<input type="text" value="20"/>	
<input type="checkbox"/> Estonia	<input type="text" value="15"/>		€	<input type="text" value="15"/>	€	<input type="text" value="20"/>	
<input type="checkbox"/> Greece	<input type="text" value="15"/>		€	<input type="text" value="15"/>	€	<input type="text" value="20"/>	
<input type="checkbox"/> Spain	<input type="text" value="15"/>		€	<input type="text" value="15"/>	€	<input type="text" value="20"/>	

However, the user can still edit individual price information.

<input type="checkbox"/> Germany	<input type="text" value="15"/> €	<input type="text" value="15"/> €	<input type="text" value="20"/>	
<input type="checkbox"/> Estonia	<input type="text" value="15"/> €	<input type="text" value="15"/> €	<input type="text" value="16"/>	
<input type="checkbox"/> Greece	<input type="text" value="15"/> €	<input type="text" value="15"/> €	<input type="text" value="12.5"/>	
<input type="checkbox"/> Spain	<input type="text" value="15"/> €	<input type="text" value="15"/> €	<input type="text" value="20"/>	

When the user clicks the “SAVE” button, the information is first validated before being sent to the server (see later).

When a price definition exists for a series of countries and the price definition is in “read” mode, only the countries for which price data exists are displayed.

Price definition

[EDIT](#)

Region / Country	Price	Normal	VAT Rate(%)
Germany	<input type="text" value="15"/> €	<input type="text" value="15"/> €	<input type="text" value="20"/>
Estonia	<input type="text" value="15"/> €	<input type="text" value="15"/> €	<input type="text" value="16"/>
Greece	<input type="text" value="15"/> €	<input type="text" value="15"/> €	<input type="text" value="12.5"/>
Spain	<input type="text" value="15"/> €	<input type="text" value="15"/> €	<input type="text" value="20"/>

Description of the initial task

Here is the task as given to the developer.

Regions:

The notion of region is described in details in

A dedicated endpoint: GET /admin/refdata/region returns the whole list of all regions as follows:

```
1{
2  "items": [
3    {
4      "regionId": <int>    (not to be used)
5      "region": <string>  (e.g. '000001001')
6      "regionDescription": <string> (e.g. 'Austria')
7      "currency": <string> (e.g. 'eur')
8      "countryCode": <string> (e.g. 'at')
9    },
10   ...
11  ]
12}
```

An initial idea is to build a "grid" that lists, on the left hand, the list of regions, in a hierarchical manner, skipping the "000" (world).

At the moment, there are only 3 levels (e.g. world > Europe > Austria) but this could be extended to 4 to 5 very max.

This grid needs to foresee up to 5 levels. Of course, unused levels are not to be displayed.

We do not display the region "codes" (000001) but their description (Europe). Internally, we will only use the region "codes" (e.g. 000001001) to reference a region.

Next to each region, there will be 3 numeric input boxes: "price", "normalPrice" & "vatRate"
For the first 2 ones (price & normalPrice), a currency symbol will be added to the input field decoration.

Input validation rules:

price & normalPrice: numeric, 2 decimals, positive

vatRate: numeric, 2 decimals: min 0, max:99.99

Editing business logic and rules:

1. We can only edit in "edit" mode => the "edit" button has been clicked.
2. Each "row" can be edited individually.
3. If the user hits the "trash" icon (on the right), all the input fields related to the row are emptied.
4. The user can edit multiple rows at a time. In order to be able to do it, the user may select multiple "checkboxes" (on the left hand-side). When multiple checkboxes are selected, when the user clicks on a textfield, a popup will be prompted with the 3 fields (price, normalPrice & vatRate) to let the user enter the values. When the user confirms the edition, ALL the selected rows' fields will be updated according to the entry made by the user. If the user does not confirm the editing, then nothing is changed.

Additional rules:

1. It is not possible to select countries that do not share the same currency. Example: it is not possible to select "Belgium" and "United Kingdom" because Belgium is using "eur" and "United Kingdom" is using "gbp".
2. When the user selects a "higher" level region (e.g. Europe), rather than selecting all the rows under it (because of the previous rule), it will select all the rows with the highest common currency.
So here, if the user wants to turn the checkbox "Europe", as "€" is the most common currency in all the underlying countries, only the countries with "€" as a currency will be automatically selected.
3. When multiple checkboxes are turned on and the user clicks on the "trash icon", a popup will ask the user to confirm.
4. The Button "Save" is only shown when differences will exist between the "initial dataset" and the "edited one".

Save rules

When the user will click the "Save" button, the routine will automatically "fill the holes" and proceed with the final validation.

1. vatRate MUST BE SYSTEMATICALLY mentioned **except** if price **AND** normalPrice are empty.
2. if price is empty but normalPrice is not empty => price = normalPrice
3. if normalPrice is empty but price is not empty => normalPrice = price
4. if both normalPrice and price are not Empty, price MUST be lower or equal to normalPrice
5. if price, normalPrice & vatRate are empty => the row is not considered

Price definition JSON

The price definition is a JSON of the following structure:

```
1{
2  "<region>": {
3    "price": <double>,
4    "normalPrice": <double>,
5    "vatRate": <double>
6  },
7  "<region>": {
8    "price": <double>,
9    "normalPrice": <double>,
10   "vatRate": <double>
11  },
12  ...
13}
```

example:

```
1{
2  "000001001": {
3    "price": 29.99,
4    "normalPrice": 34.99,
5    "vatRate": 19.6
6  },
7  "000001002": {
8    "price": 29.99,
9    "normalPrice": 34.99,
10   "vatRate": 19.6
11  },
12  "000001003": {
13    "price": 124.99,
14    "normalPrice": 159.99,
15    "vatRate": 12.8
16  },
17  ...
18}
```

Read view (layout)

When the block is not in "editing mode", it only shows the list of "rows" for which data exists.

Example:

Suppose that there are only price definition for "Belgium", "UK" & "France", then, only these 3 countries are shown.

When the user hits the "edit" button, the block turns to the editing form, showing all the possible rows".

If no price definition exists, the block displays "No price definition".

Objective of the exercise

We need to extend the current code to meet the following additional criteria:

1. The popup that allows the user to enter data for multiple countries at once, should also display the currency symbol for both “price” and “normalPrice”. How would you implement it?
2. The popup should pre-fill the entries ONLY when ALL selected countries share the very same “values”. Example: suppose the following 2 countries:
 - a. Belgium : price 19.99, normal price: 24.99 and vatRate 21.0
 - b. Italy: price 19.99, normal price: 24.99 and vatRate: 19.6
 - c. France: price 19.99, normal price: 27.50 and vatRate: 19.6

In this case, only the common “price” (19.99) is pre-filled, the other 2 fields remain empty

3. How would you implement the “Additional rule number 4” (4. *The Button “Save” is only shown when differences will exist between the “initial dataset” and the “edited one”*)
4. Could you please tell me whether the following rule is correctly implemented: “2. When the user selects a “higher” level region (e.g. Europe), rather than selecting all the rows under it (because of the previous rule), it will select all the rows with the highest common currency” If yes, where and how? If not, where would you implement it and how?
5. One of the basic requirements of the website is to support multiple languages. To meet the requirement, we use “FormattedMessage”. However, “ProductPriceBlock.js” does not fully comply with the requirement. Why and how to fix it?
6. Bonus:

In the file “DataBlock.js”, the “DataBlockFormProvider” is a generic component, used hundreds of times in the project.

Lines 198-204: this corresponds to a “hack” that was initially used by the developer to “automatically” prepare the JSON that is returned by the Server, so that it is done once for all at data reception time.

However, the fact that this code is inside that generic component is not very clean, as it will be executed by the whole application even when Blocks are not related to the notion of price.

Therefore, where would you “move” (and adapt) that part of the code so that it is only run when related to the “ProductPriceBlock”?