J.D. POWER

Build and Price Service

Configuration Endpoints How-To

October 10, 2024
Version 1.2

Build and Price Service Configuration Endpoints How-To and Notes

For the Build and Price Service Configuration endpoints here are some details and a couple of use cases needed to get started.
A Postman collection should accompany this document as it will provide the related calls to the endpoints, requests and responses discussed below.

Contents

Use cases: 3

Properties of an Option 6

Option State 7

Option Flags & Definitions 8

Profiling 9

Use cases:

1. Get a default list of options for a vehicle id
2. Select/toggle options on the vehicle while maintaining state

Use Case 1 - Get a default list of options for a vehicle id

1.1 - Get a token for a vehicle id
PUT …buildandprice/vehicle/token/{vehicleID}
Ex: PUT …buildandprice/vehicle/token/412967
Request Body:
{}
Returns a token in the response (at *result.token*) (Ex: 4154054742) that is used in the next step. The tokens expire around every 8 hours so will have to request new ones periodically.

1.2 Get the options
PUT …buildandprice/configuration/{vehicleToken}
Ex: PUT … buildandprice/configuration/4154054742
Request Body:
{}
Returns an option summary, configuration state string (at *result.configState*) and price summary

Use Case 2 - Toggle an Option

PUT … buildandprice/configuration/{vehicleToken}/toggle/{optionCode}
Ex: PUT … buildandprice/configuration/4154054742/toggle/START1
Request Body:

```
{
     "logicMethod": "SIMPLELMETHOD",
    "useEnforcers": "true",
    "showDelta": "false",
    "paint": "false",
    "differenceOnly": "false",
    "configState":"||STDEN,1,0,0,0,S,^2TB,1,0,0,0,S,^STDTR,1,0,0,0,S,^1P0,1,0,0,0,S,^4F
M,1,0,0,0,S,^VA~~,1,0,0,0,S,^FEE,1,0,0,0,S,^PAINT,1,0,0,0,S,^STDRD,1,0,0,0,S,^1N1,0,0,2
,0,C,^1N7,0,0,2,0,C,^1N9,0,0,2,0,C,^1PZ,0,0,2,0,C,^1W9,0,0,2,0,C,^NONTR1,0,0,2,0,C,^NON
TR2,0,0,2,0,C,^NONTR3,0,0,2,0,C,^258,0,1,0,0,G,^4HB,0,1,0,0,G,^XD5,0,0,1,0,C,^ZBEI_02,0
,0,1,0,C,^ZBEJ_02,0,0,1,0,C,^ZBEX_01,0,0,1,0,C,^ZBHX_02,0,0,1,0,C,^ZBHY_02,0,0,1,0,C,^Z
BHZ_02,0,0,1,0,C,^ZBRI_02,0,0,1,0,C,^ZBSW_02,0,0,1,0,C,^ZB~~,0,0,1,0,C,^655,0,1,0,0,G,^
688,0,1,0,0,G,^6AC,0,1,0,0,G,^6AE,0,1,0,0,G,^6NS,0,1,0,0,G,^6U3,0,1,0,0,G,^6WD,0,1,0,0,
G,^ZAX,0,1,0,0,G,^"
}
```

Returns same as 1.2.

Additional information about the Toggle endpoint

Below some the more frequently used parameters of the toggle endpoint are explained further:

useEnforcers : An enforcer is a category of option that has been identified by the data entry team as being a one-and-only-one category. Typically this is Engines, Transmission, Packages, Wheels, Tires, Seats/Seat Trims, GVWR, Radios, etc.. Possible values are true or false. Setting this to true will ensure all enforcer categories have one and only one option selected. I highly recommend you use a value of true here.

differenceOnly : Controls if only options that have been changed as part of the toggling process are returned and thus reducing payloads. Possible values are true or false. You can experiment, but using false should suffice for your needs

logicMethod : This controls the level of decisions that are automatically made by the underlying configurator engine. Possible values are: SIMPLELMETHOD, COMPLETELMETHOD or EASYLMETHOD. These can be further defined as:

COMPLETELMETHOD : No decisions to select or unselect an option that is dependent on the option being toggled is handed automatically by the engine and results in the user having to approve all dependent option selections and unselections.

SIMPLELMETHOD : All decisions to unselect an option that is conflicting with the toggled option are done automatically by the configurator engine. If the toggling of an option requires other options to be selected and there is only one path to go down, that option is selected automatically. However, if there is more than one path to go down, then the possible options are returned to the user and a decision must be made.

EASYLMETHOD : All decisions are made by the engine. In the case of more than one path, the first one in the list is chosen.

This table illustrates the user experience that happens in each scenario:

	Logic Methodology		
Scenario	**Complete**	**Simple**	**Easy**
A can't have X (X is currently selected)	User is asked if they want to unselect X in order to select A	X is unselected automatically by the configurator engine	X is unselected automatically by the configurator engine
A must have B (B is not currently selected)	User is asked if they want to select B in order to select A	B is selected automatically by the configurator engine	B is selected automatically by the configurator engine
C must have D or E (Neither D or E is selected)	User is asked if they want to select D or E in order to select C	User is asked if they want to select D or E in order to select C	D is selected automatically by the configurator engine

Simple is used 90% of the time and gives the best user experience, typically. I recommend using it to start with at least. Easy is usually used for automated testing purpose. Complete is typically used when the user base Is very passionate about vehicle configuration data. We can also create custom logic methodologies.

showDelta : Possible values are true or false. In cases where decisions are being made by the configurator engine (as shown above) this includes a section in the response, when set to true, that is a summary of all options that have been added, removed or changed as part of the toggle request. Assuming you are going to use SIMPLELETHOD , I would set this as true

configState : This is the Configuration State String and represents the current state of the options for the current configuration. Our configurator engine and service is stateless and this is used to pass option states back and forth. The default state is returned in the response of the /configuration/{token} endpoint and would then be passed as the value for this parameter on the first option toggle. That toggle would return a new configuration state string that would be used on the subsequent toggle, etc. A sample value is provided below.

previousConfigState : This is needed if true is passed for the showDelta parameter. The value is a configuration state string from the last "complete" configuration state. On an initial selection of an option this and configState are equal. In the case where the toggle command returns a MUSTHAVE or CANTHAVE result and another toggle command is issued after the users decision, this value stays the same, but configState gets updated

Sample configuration state string:

```
||STDEN,1,0,0,0,S,^2TB,1,0,0,0,S,^STDTR,1,0,0,0,S,^1P0,1,0,0,0,S,^4FM,1,0,0,0,S,^VA~~,1,0,0,0,
S,^FEE,1,0,0,0,S,^PAINT,1,0,0,0,S,^STDRD,1,0,0,0,S,^1N1,0,0,2,0,C,^1N7,0,0,2,0,C,^1N9,0,0,2,0,
C,^1PZ,0,0,2,0,C,^1W9,0,0,2,0,C,^NONTR1,0,0,2,0,C,^NONTR2,0,0,2,0,C,^NONTR3,0,0,2,0,C,^258,0,1
,0,0,G,^4HB,0,1,0,0,G,^XD5,0,0,1,0,C,^ZBEI_02,0,0,1,0,C,^ZBEJ_02,0,0,1,0,C,^ZBEX_01,0,0,1,0,C,
^ZBHX_02,0,0,1,0,C,^ZBHY_02,0,0,1,0,C,^ZBHZ_02,0,0,1,0,C,^ZBRI_02,0,0,1,0,C,^ZBSW_02,0,0,1,0,C
,^ZB~~,0,0,1,0,C,^655,0,1,0,0,G,^688,0,1,0,0,G,^6AC,0,1,0,0,G,^6AE,0,1,0,0,G,^6NS,0,1,0,0,G,^6
U3,0,1,0,0,G,^6WD,0,1,0,0,G,^ZAX,0,1,0,0,G,^
```

Properties of an Option

code	the option code to use when issuing a toggle command
displayCode	If showing the option code to the end user, then use this value
state	See Option State section
internal	will be true if the OEM did not provide an option code and we had to create on ourselves
description	primary option description
extendedDescription	secondary option description that typically includes information about an option that the OEM provides and/or textual content that is not represented by other options on the vehicle
note	typically, an internal note about ordering or availability, not usually shared with consumers
optionClass	
ecc	an internal option classification system
enforcer	internal flag whether the option is part of a one and only one category. Can be used to indicate if a radio button vs check box should be used
contentList	all of the options that are included as part of groupContent of this option
flags	See Option Flags section
priceType	indicates information about the current state of the pricing of the option. Possible values are: PRICED, INCLUDED, VARIABLE, STANDARD, NO_CHARGE, NOT_AVAILABLE
stateSetByList	includes the options that have put the option in the current state. It is sometimes shared with end users, but not usually in a consumer facing context.

Option State

The possible values for state are:
SELECTED – currently configured on the vehicle
CANTHAVE – a currently selected option is restricting this option and usually indicated to the user with an “X”
GROUP – a currently selected option includes this option as content
REMOVED – this option was group, but another selected or grouped option has replaced it. (Typically an upgrade/downgrade of content scenario). In a consumer facing setting I would display this to the user the same way you indicated can’t haved options
UNSELECTED – option is available to be selected

The option will return a price, but the price of the option only gets included in the total price of the vehicle returned by the service if the option state = SELECTED

Option Flags & Definitions

Name	**Description**
NOTAVAIL	the pricing is not available for this option.
NOCHARGE	the option is no charge
NONORDER	a non orderable option and not likely to be returned in the “visible” set of options that are returned
FLEET	option is for fleet orders/customers only
STDEQUIP (or STDOPTION)	option is part of the standard equipment on the vehicle
REGIONAL	option only available in certain regions. Region availability typically defined by a note
POSTPRODUCTIONOPT	installed after production
POSTINSTALLOPT	option installed at the Port, installation cost included
PORTUNINSTALLOPT	option installed at the Port, installation cost is not included
DEALERINSTALLED	option is installed by the dealer and the price of installation is included
DLRUNINSTALLEDOPT	option is installed by the dealer and the price of installation is not included
CONSUMER	Not available for fleet ordering
INTERNAL	the option code was created by Autodata as all of our options require an option code and that is the unique identifier on the vehicle
LTDPRODUCTIONOPT	limited production option
DISCONTINUED	option no longer available for ordering, but was at one point
TRANSMISTCODE	option to included in the order string that is submitted to the manufacturer for ordering the vehicle
OPTISLOCKED	Option has been locked and it’s current state cannot to be changed (Locking an option is not currently available in the Configuration Service)
ONLYAVAILASCONTENT	Option can only be selected by selecting another option
OPTHASDEPENDENCY	In order to select this option another option is required to be selected. (Note: that option may already be selected)

For determining what to display for pricing, a typical implementation uses the following logic:

If the state of the option is Group, show “Included”
Else If NOTAVAIL, show “NA”
Else if STDEQUIP, show “Standard” or “N/C”
Else if NOCHARGE, show “N/C”
Else, show the price returned for the option

Profiling

The act of profiling edits the default set of options on the vehicle being used. Profiling is an optional step if there are integration use cases that require it. Currently, profiling can be used to remove and reorder options on the option list. Profiling is done at the time of the token request and the options stay in that profiled state for the life of the token.

Sample buildandprice/vehicle/token/<vehicle_set>/<vehicle_id> response body to perform profiling:

```
{
    "profileInfo" : [
        {
             "command" : "exclude",
             "key" : "orderInfo",
             "values" : [
              ]
        },
        {
             "command" : "include",
             "key" : "colorPrimary",
             "values" : [
                  "gray",
                  "blue"
              ]
        },
        {
             "command" : "exclude",
             "key" : "flags",
             "values" : [
                  "FLEET"
              ]
        },
        {
             "command" : "reorder",
             "key" : "eccfilter",
             "values" : [
                  "CLIENTA"
              ]
        }
    ]
}
```

The behaviour of the four entries in the profilieInfo array above are summarized as:

1. Remove all ordering information options (See note 1 below for more detail) from the set of returned options for the vehicle
2. Keep only primary colour options that have a generic colour of gray or blue on the option list.

3. Remove all options that are flagged as fleet options
4. Reorder the options based on the ECC group that have been defined as part of the "CLIENTA" profile. These profile are defined within the Configuraiton service project in the ECCFitlers.xml file

Definition of elements within the profileInfo array:

- **command** – Possible valies are include or exclude. Include will contain only options that contain the values in the result. Values will be treated as an OR condition. Exclude will remove the options that contain the values in the result. Values will be treated as an OR condition.
- **key** - The attribute to be filtered in the results. Supported values for key are - code, ecc, flags, orderInfo, colorPrimary, colorSecondary, colorExtra, colorInterior, and reorder
  Note 1: "orderInfo": value is not required for this key and typically used with the exclude command. If used, results will exclude options where the code starts with either "BUILD", "START", "FINAL" or "ORDER

  Note 2: All color keys will use the Generic Color for filtering the include or exclude and not the full OEM marketing description of the color.
- **values** - The passed list of strings to be included or excluded. Case should match what is returned in the optionList response when an unfiltered request is made.