|  |  |
| --- | --- |
|  | Mitchell1 Information Systems  16067 Babcock Street  San Diego, CA 92127 |

Mitchell1 Online Catalog SDK

*Mitchell1’s Extension for Creating Online Shop Manager SE Catalogs – Version 3*

Mitchell 1 Shop Management Solutions

San Diego, California

# Revision Notes

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Changes** |  |
| 06/21/2018 | 1.0 | Initial version of documentation |  |
| 04/22/2020 | 1.1 | Updated for Order Tracking |  |
| 03/30/2021 | 1.2 | Updated with Metadata, Supplier and Shipping Updated to explain webpage capability for PriceCheck/OrderParts. Updated various objects and fixed formatting. |  |
|  |  |  |  |

# 

# General

## Introduction

## Goals

This document is meant to assist in the development of a parts catalog to run from and interact with the Mitchell1 Manager SE products. Using this guide, a developer should be able to create a parts catalog which will be dynamically loaded into the host and allow the user to setup a vendor link (authentication), shop for parts, check pricing, and order parts.

## Scope

This document covers the Mitchell1 Catalog Software Development Kit which contains details on how to develop a parts catalog that includes vendor setup, shopping for parts, price check, and parts ordering.

The host application was designed to persist information such as the vehicle qualifiers, vendor setup, and all selected and ordered parts. Vendor information is **not** passed back to host application. You **must not** return sensitive information: either personal, confidential, user credentials, or other easily identifiable information from vendor setup. Instead, use tokens/cookies, which your server knows how to validate/process/identify when making calls to Vendor Setup, Go Shopping, Price Check, and Parts Ordering.

This document does not specify how the catalog should list parts, search for parts or maintain a shopping cart. These details are determined by the catalog implementer. We only require you to conform to the in/out JSON objects. For web based interaction you should utilize the provided JavaScript functions.

## Applicability

Any parts catalog which is compatible with the Catalog Driver, will also be compatible with the major Manager SE and Shopkey Management SE applications. Mitchell1 requires the catalog developers/distributers to work with Mitchell1 for inclusion into the Manager SE Product. The Shop management application only allows properly permitted parts catalogs to be used.

## Conventions

The following font conventions are used in this document:

|  |  |
| --- | --- |
| STYLE | MEANING |
| FillCart | Indicates program code or verbatim text |
| **AddLocation** | Indicates a method or property name |
| **CartItem** | Indicates a struct, class, or interface name |

# Mitchell1 Catalog Driver

The Mitchell1 Catalog Driver is used to test the catalog development in place of the Mitchell1 ShopStream Manager SE software. It tests all the essential features of a parts catalog without requiring them to be digitally signed. Only catalogs that have been validated and digitally signed by Mitchell1 can be run in the actual Manager SE product.

In general terms, the steps to test a parts catalog under Catalog Driver are:

1. Build your parts catalog which implements all of the required interfaces outlined in the SDK
2. Build and run the Catalog Driver application (in the **Driver** folder)
3. Select your catalog under the *Catalog* drop-down on the user interface
4. Push the vendor setup button to enter the authentication details
5. Click the “Go Shopping” button which appears on the Catalog Driver
6. Select parts and add them to your cart
7. Transfer the carts back to the Catalog Driver
8. Verify that all of the items from the shopping cart were transferred to Catalog Driver
9. Click on “Price Check” and initiate a check of prices for your transferred items
10. Verify that the prices, quantity, and availability were updated
11. Check the “Price Check” log to verify that the request and response are correct
12. Click on “Order Parts” and initiate an ordering of the parts you price checked
13. Verify that the status of the parts has changed to indicate the order status
14. Check the “Order Parts” log to verify that the request and response are correct

## Online Catalog Configuration/Definition

Using the sample catalog driver application, you can provide the basic settings of your catalog.

**Note:** We will require the **OnlineConfig.xml** file (generated by Catalog Driver after configuring online catalog) when we validate and prepare to have your Catalog enabled in Manager SE.

### Basic Configuration

### 

Note: Use the Catalog Driver’s main vendor setup / shopping carts for all validation.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **Name** | Name of Catalog that will be presented to users |
| **Description** | Description of Catalog |
| **Support Site** | URL For Support – shown when there are HTTP/Web Errors |
| **ApiVersionLevel** | Minimum API level that this Catalog supports (should be 1 for all catalogs) |
| **Support Phone** | Phone for Support – shown when there are HTTP/Web Errors |
| **Base Catalog Url** | Base URL all other endpoints will utilize. E.g. https://sample.com/ |
| **Icon** | Url for getting an Icon to in various parts of host application. Url gets appending with ?width=x&height=y to give Vendor a chance to provide an icon size ideal for requested area. However, can return any size icon, it will be scaled to fit where used. |
| **Setup** | Vendor Setup page where customer would provide needed credentials to access your catalog |
| **Shopping** | Shopping Cart page. E.g. shopping.aspx |
| **Price Check** | Price Check API location – JSON Sent/Returned. E.g. api/pricecheck.aspx |
| **Order Parts** | Order Parts API location – JSON Sent/Returned. E.g. api/orderparts.aspx |
| **Order Tracking** | **Optional:** API for returning tracking information. Will be passed a tracking number, and API returns URL and short status info |
|  |  |
|  |  |

### Advanced Configuration

### 

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **bool** AllowsBlankManufacturerCode | Indicates whether the catalog allows an empty (undefined) manufacturer code for parts used in PriceCheck or OrderParts. |
| **bool** AllowsNotFoundPartsToBeOrdered | Indicates whether the catalog allows parts which were not found in the PriceCheck to be ordered. |
| **bool** ShowsDeliverWillCall | Indicates whether the host application (Manager SE) will show a “Deliver” “WillCall” choice for placing orders. Manager SE passes which ever choice user made. |
| **bool** RequiresPriceCheck | Indicates whether the host application should only allow ordering of parts that have been price checked. |
| **bool** SupportsAlternateLocations | Indicates whether the catalog supports alternate locations within the PriceCheck method. If true, the host application will allow the user to select from a list of Locations. |
| **bool** SupportsAlternateParts | Indicates whether the catalog supports alternate parts within the PriceCheck method. If true, the host application will allow the user to select from a list of AlternateParts. |
| **bool** SupportsLocation | Indicates whether the catalog can specify a location for a part. If true, the host application will display the part location. |
| **bool** SupportsOrderMessage | Indicates whether the catalog supports the OrderMessage property within the OrderRequest object. If true, the host application will allow the user to enter an order message that will be sent with the order. |
| **bool** SupportsPriceCheck | Indicates whether the catalog supports the PriceCheck method. If true, the host application will allow the user to perform a price check. If false, the host application will not allow the user to perform a price check. |

Required Catalog URLs/APIs

### Note: All configured URLs are relative to the “Base Catalog Url”. If needed to be outside this, your web server can implement Url Redirects, Url Rewrite, or other methods where Manager SE would still call the [base]/[component] Url, but your server would handle it or tell Manager SE to look elsewhere.

* Icon – Needs to return an Image File (PNG, BMP, JPG) to caller. An ideal size parameter is passed in Url: [baseUrl]/[icon]?width=58&height=28. This information can be used by your web server to find the best fitting image for the requested area (e.g. square aspect / wide aspect / etc). No resizing is needed, Manager SE will scale image (but, if source image is much bigger than requested size, will lose detail when scaled down, so finding the smallest/biggest image based on size is up to you).
* Setup – This needs to be a public web site that gets loaded into an internal browser (currently, Chromium based)
  + The site must have loaded the catalog-v3.0.0.js file provided (see ExampleCatalog app – configuration.html)
  + When site is loaded, any existing vendor setup data (qualifier) is available (**See** HTML help for details on JavaScript API)
  + Do not save plain text credentials or personally identifying information. You should store a token/cookie and some other small objects as necessary that you can validate the user in future calls – you will be passed this object as part of Vendor Configuration, Go Shopping, Price Check, and Order parts to verify the user / account has access
  + You have the ability to save new login token data, or cancel (**See** HTML help for details on JavaScript API)
* Shopping – This will be the web site that gets loaded to find and transfer parts back into hosting application (Manager SE)
  + Using the same JavaScript file as Setup, and qualifier (included in query string – catalogSdk.qualifier).
    - Note: You should verify all credentials as part of server back-end, example app does it in JavaScript as example only. **Important**: If credentials are missing or invalid, you must return a 403 HTTP Code. Do not add any custom UI message on your end. When Manager SE encounters the 403, it will message the customer and will prompt the user to go back into vendor setup area and the browser window will close automatically.
  + You can add labor/parts/notes to shopping cart – which will get price checked/ordered at some later time
  + And/or you can create a purchase order (adding parts to it) and transfer that back. Use this if your Go Shopping session directly orders/ships parts and does not need to be ordered later
  + The object passed to the JavaScript function Catalog.transferParts (defined in catalog-v3.0.0.js) should be a json string that matches the type ShoppingCart (see “Object Definitions” below.)
  + **See** HTML help for details on JavaScript API and also look at the Example Catalog.
* Price Check **– API used for checking current prices and available quantity of requested parts (normally parts returned from go shopping). If the response contains the field AbsoluteRedirectUrl (when HostApiLevel is 3 or greater) then a web page is opened using the value from this field, and the JavaScript function Catalog.transfer can be used to return a response back to Manager SE. See example catalog app.**
  + **Request Body – application/json (**PriceCheckRequest**)**
    - **HostData** : (HostData) (contains HostApiLevel)
    - **Vendor** : (Vendor)
    - **PriceCheck** : (PriceCheck)
    - **Vehicle** : (Vehicle)
  + **Response Body – application/json (**PriceCheckResponse**)**
    - **AbsoluteRedirectUrl** : (string)(when HostApiLevel 3+)
    - **Parts** : (IList<PriceCheckPart>)
  + Important: Ensure you return price check items in the same order as were sent to you (see example app). As well as using correct casing for JSON objects/properties, otherwise price check will not match items correctly.
* Order Parts **– API used for making an order. If the response contains the field AbsoluteRedirectUrl (when HostApiLevel is 3 or greater) then a web page is opened using the value from this field, and the JavaScript function Catalog.transfer can be used to return a response back to Manager SE. See example catalog app.**
  + **Request Body – application/json (**OrderRequest**)**
    - **HostData** : (HostData) (contains HostApiLevel)
    - **Vendor : (**Vendor)
    - **Order** : (Order)
    - **Vehicle** : (Vehicle)
  + **Response Body – application/json (**OrderResponse**)**
    - **AbsoluteRedirectUrl** **: (**string) (when HostApiLevel 3+)
    - **ConfirmationNumber** : (string)
    - **Parts :****(**IList<OrderPart>)
    - **TrackingNumber** : (string)
  + Important: Ensure you return order parts items in the same order as were sent to you (see example app). As well as using correct casing for JSON objects/properties, otherwise parts ordering will not match items correctly.

Optional Catalog URLs/APIs

### As above, these sub URL components are built off the base url:

* Order Tracking **– API used to query for an external link to launch in external browser. Can also include a short order status message. This call will pass via HTTPS/Post the previously returned order tracking number, as well as the vendor setup qualifier. This allows you to authenticate the request (if required), and provide a URL safe to load/show in an External browser – if URL is temporary, it should stay valid for at least 8hr’s) See example catalog app.**
  + **Request Body – application/json (**TrackingRequest**)**
    - **HostData** : (HostData)
    - **Vendor : (**Vendor)
    - **OrderTrackingNumber** : (string)
  + **Response Body – application/json (**TrackingResponse**)**
    - **ExternalTrackingUrl**: (Uri)
    - **StatusDisplay**: (string)

**Object Definitions**

This section contains definitions for all of the objects which the online catalog uses to pass data.

## ShoppingCart

ShoppingCart is a list of Items (**List<CartItem>**) that can be transferred from the Shopping Cart page. The Items can be PartItem, LaborItem, NoteItem, or ShoppingCartOrder. The JavaScript function Catalog.transferParts can be used to return the list of items to Manager SE. Note that for backwards compatibility (versions of Manager where HostApiLevel is less than 3) this must be a json object rather than a json string. **See example catalog app.**

## PartItem

PartItem represents a part that belongs to a cart. The **PartNumber** and **ManufacturerLineCode** combined uniquely identify an individual part from the catalog. The JavaScript function Catalog.createPartItem2 can be used to create this object. **See example catalog app.**

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **PartNumber** | Part Number of part. (Max length 20) |
| **string** **ManufacturerLineCode** | Manufacturer Line Code for the part. (Max length 6) |
| **string** **ManufacturerName** | Manufacturer Name for the part. (Max length 30) |
| **string** **Description** | Description for the part. (Max length 50) |
| **decimal** **UnitList** | Suggested unit price for the part |
| **decimal** **UnitCost** | Cost for a single unit of this part |
| **decimal** **UnitCore** | Cost of the core charge for the part. |
| **decimal** **Quantity** | Quantity of this part requested |
| **bool** **IsTire** | Determines if the part is a tire or not |
| **string** **Size** | Size of the part (usually a tire size like “255x55 r17”) (Max length 20) |
| **string** **UpcCode** | Universal Product Code for the part (Not persisted in Manager SE) |
| **PartCategory PartCategory** | Category of part transferred. (Can be **Tire**, **Wheel**, or **Unspecified**) (Defaults to Unspecified) |
| **string** **SupplierName** | Uniquely identifies (within this Catalog) the source supplier for this part. This may be displayed to the user. (Max length 25) (Optional - Available when HostApiLevel 3+) |
| **string** **Metadata** | Custom Metadata which may include Supplier information. (Max length 4000 characters) (Optional - Available when HostApiLevel 3+) |
| **decimal ShippingDescription** | Description of Shipping Cost. (Max length 25) (Optional - Available when HostApiLevel 3+) |
| **string** **ShippingCost** | Shipping cost for this part. (Optional - Available when HostApiLevel 3+) |

## LaborItem

LaborItem represents a labor work item that belongs to a cart. The JavaScript function Catalog.createLaborItem can be used to create this object. **See example catalog app.**

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **Description** | Description of the labor to be performed. (Max length 255) |
| **decimal** **Hours** | Estimated time that this labor item will take to complete |
| **decimal** **Price** | The suggested price to complete this labor item. If this property is 0, then the host application will calculate the price based on the Hours submitted and host’s labor rate. |

## NoteItem

NoteItem represents a note or comment that is added to the shopping cart. Note items should be added just before the PartItem or the LaborItem that they annotate. A NoteItem is not figured into the total cost for a repair order or any other billing item. The JavaScript function Catalog.createNoteItem can be used to create this object. **See example catalog app.**

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **Note** | A note or comment to be placed on a repair order |

## ShoppingCartOrder

ShoppingCartOrder represents a catalog order made through the Shopping Cart page. It contains a list of ShoppingCartOrderPart objects along with various other properties which uniquely identify the order. The JavaScript function Catalog.createOrderItem can be used to create this object. **See example catalog app.**

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **OrderMessage** | User entered message sent to the catalog along with the order. |
| **string** **DeliveryOptions** | Delivery Option for parts order. – Possible values include: Deliver, WillCall (if catalog marked as showing delivery options). |
| **string ConfirmationNumber** | A confirmation number for the entire order used to verify the order was successful. If Null/Empty, order is assumed to be failure. |
| **string TrackingNumber** | **Optional:** Used for catalog to return a tracking id/number: (Max 20 chars). |
| **IList<** **ShoppingCartOrderPart>** **Parts** | A list of parts the user ordered directly from within the catalog. (See ShoppingCartOrderPart definition) |

## ShoppingCartOrderPart

The ShoppingCartOrderPart object represents an individual line item found in a parts order (ShoppingCartOrder). Unlike the CartItem, an order does not deal with notes or labor. Only physical parts can be ordered. The JavaScript function Catalog.addOrderPart2 can be used to add ordered parts to the ShoppingCartOrder object. **See example catalog app.**

Together, the **ManufacturerLineCode** and the **PartNumber** uniquely identify a part in the order.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **LocationId** | Id of the location (as determined by the catalog) (Optional) (Max length 8 characters) |
| **string** **LocationName** | Name of the location (as determined by the catalog) (Optional) (Max length 20 characters) |
| **string** **Status** | Custom Status of a processed part. Reserved for future use. |
| **string** **PartNumber** | Part Number of part. (Max length 20) |
| **string** **ManufacturerLineCode** | Manufacturer Line Code for the part. (Max length 6) |
| **string** **ManufacturerName** | Manufacturer Name for the part. (Max length 30) |
| **string** **Description** | Description for the part. (Max length 50) |
| **decimal** **UnitList** | Suggested price for a single unit of this item |
| **decimal** **UnitCost** | Cost for a single unit of this item. |
| **decimal** **UnitCore** | Cost of a single core item. |
| **decimal** **QuantityRequested** | Quantity of part requested to be ordered. |
| **decimal** **QuantityOrdered** | Quantity of part ordered. |
| **decimal** **QuantityAvailable** | Quantity of this part item available at this location. |
| **string** **Size** | The size of the part (usually a tire size like “255x55 r17”) (Max length 20) |
| **PartCategory PartCategory** | Category of part transferred. (Can be **Tire**, **Wheel**, or **Unspecified**) (Defaults to Unspecified) |
| **string** **SupplierName** | Uniquely identifies (within this Catalog) the source supplier for this part. This may be displayed to the user. (Max length 25) (Optional - Available when HostApiLevel 3+) |
| **string** **Metadata** | Custom Metadata which may include Supplier information. (Max length 4000 characters) (Optional - Available when HostApiLevel 3+) |
| **decimal ShippingDescription** | Description of Shipping Cost. (Max length 25) (Optional - Available when HostApiLevel 3+) |
| **string** **ShippingCost** | Shipping cost for this part. (Optional - Available when HostApiLevel 3+) |

## OrderRequest

An OrderRequest is used for ordering parts. It contains an Order object which contains a list of OrderPart objects along with various other properties which uniquely identify the order. An instance of this type is passed to the Order Parts API call.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **HostData** **HostData** | Information that the host application makes available to catalogs. (See HostData object definition) |
| **Vendor** **Vendor** | Represents a parts Vendor (See Vendor object definition) |
| **Order** **Order** | The list of parts being ordered (and other information). (See Order object definition) |
| **Vehicle** **Vehicle** | Represents the vehicle for which parts are being ordered (See Vehicle object definition) |

## OrderResponse

An OrderResponse object holds the results of an Order Parts operation. An instance of this type is returned from the Order Parts REST API.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **AbsoluteRedirectUrl** | **Optional**: If the response contains this field (when HostApiLevel 3+) then a web page is opened using this Url, and the JavaScript function Catalog.transfer can be used to return the OrderResponse back to Manager SE. **See example catalog app.** |
| **string** **ConfirmationNumber** | A confirmation number for the entire order used to verify the order was successful. If Null/Empty, order is assumed to be failure. |
| **string** **TrackingNumber** | **Optional:** Used for catalog to return a tracking id/number: (Max 20 chars). |
| **IList<OrderPart>** **Parts** | Should contain same parts that were passed in as request – however, qty/cost/etc values updated with what was ordered |

## Order

An Order represents a catalog order made through the host application. It contains a list of OrderPart objects along with various other properties which uniquely identify the order. An instance of this type is passed as the Order property of the JSON object given to the **Order Parts** API.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **DeliveryOption** | Delivery Option for parts order. – Possible values include: Deliver, WillCall (if catalog marked as showing delivery options) |
| **string** **OrderMessage** | User entered message sent to the catalog along with the order. |
| **string** **PurchaseOrderNumber** | Purchase Order Number as determined by host application. Passed to catalog. |
| **IList<OrderPart>** **Parts** | A list of parts the user requested for ordering. Note that this list of parts must match the list in OrderResponse with respect to position and length. |

## OrderPart

The OrderPart object represents an individual line item found in a parts order (OrderRequest.Order and OrderResponse). Unlike the CartItem, an order does not deal with notes or labor. Only physical parts are manipulated when performing parts ordering.

Together, the **ManufacturerLineCode** and the **PartNumber** uniquely identify a part in the order.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **bool** **Found** | Indicates whether the catalog recognize this part. (Set by the catalog) |
| **string** **LocationId** | Id of the location (as determined by the catalog) (Optional) (Max length 8 characters) |
| **string** **LocationName** | Name of the location (as determined by the catalog) (Optional) (Max length 20 characters) |
| **string** **Status** | Custom Status of a processed part. Reserved for future use. |
| **string** **PartNumber** | Part Number of part. (Max length 20) |
| **string** **ManufacturerLineCode** | Manufacturer Line Code for the part. (Max length 6) |
| **string** **ManufacturerName** | Manufacturer Name for the part. (Max length 30) |
| **string** **Description** | Description for the part. (Max length 50) (This property should not be updated unless it is passed in as a blank string) |
| **decimal** **UnitList** | Suggested price for a single unit of this item. (Set by the catalog). |
| **decimal** **UnitCost** | Cost for a single unit of this item. (Set by the catalog). |
| **decimal** **UnitCore** | Cost of a single core item. (Set by the catalog). |
| **decimal** **QuantityRequested** | Quantity of part requested to be ordered. |
| **decimal** **QuantityOrdered** | Quantity of part ordered. (Set by the catalog) |
| **decimal** **QuantityAvailable** | Quantity of this part item available at this location. (Set by the catalog). |
| **string** **SupplierName** | Uniquely identifies (within this Catalog) the source supplier for this part. This may be displayed to the user. (Max length 25) (Optional - Available when HostApiLevel 3+) |
| **string** **Metadata** | Custom Metadata which may include Supplier information. (Max length 4000 characters) (Optional - Available when HostApiLevel 3+) |
| **decimal ShippingDescription** | Description of Shipping Cost. (Max length 25) (Optional - Available when HostApiLevel 3+) |
| **string** **ShippingCost** | Shipping cost for this part. (Optional - Available when HostApiLevel 3+) |

## PriceCheckRequest

A PriceCheckRequest is used for checking current prices and available quantity of requested parts (normally parts returned from go shopping). It contains a PriceCheck object which contains a list of PriceCheckPart items along with various other properties which can be used in the Price Check operation. An instance of this type is passed to the Price Check API call.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **HostData** **HostData** | Information that the host application makes available to catalogs. (See HostData object definition) |
| **Vendor** **Vendor** | Represents a parts Vendor (See Vendor object definition) |
| **PriceCheck** **PriceCheck** | The list of parts being price checked (and the delivery option). (See PriceCheck object definition) |
| **Vehicle** **Vehicle** | Represents the vehicle for which parts are being ordered (See Vehicle object definition) |

## PriceCheckResponse

A PriceCheckResponse object holds the results of a Price Check operation. An instance of this type is returned from the Price Check API call.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **AbsoluteRedirectUrl** | **Optional**: If the response contains this field (when HostApiLevel 3+) then a web page is opened using the value from this field, and the JavaScript function Catalog.transfer can be used to return the PriceCheckResponse back to Manager SE. **See example catalog app.** |
| **IList<PriceCheckPart>** **Parts** | The list of parts returned from a Price Check operation. Note that this list must be of the same length and in the same order as the original list passed in the PriceCheckRequest.PriceCheck object. (See PriceCheck and PriceCheckPart object definition) |

## PriceCheck

A PriceCheck object is used to hold all of the information necessary for performing a price check on the catalog. It holds information necessary to build the request and also store the response. This is passed as PriceCheck property to the Price Check API call.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **DeliveryOption** | Delivery Option to consider for price check. – Possible values include: Deliver, WillCall (if catalog marked as showing delivery options) |
| **IList<PriceCheckPart>** **Parts** | The list of parts to use when doing a price check ( |

## PriceCheckPart

The PriceCheckPart represents an individual part that is used for performing a price check. A list of PriceCheckPart items is contained within a PriceCheck object.

Information about the unit cost, unit list, and quantity available are contained within the Location object. The SelectedLocation is the original location, and it must be contained in the list of Locations. The others Locations are alternate locations. Even if a catalog does not support alternate locations, one location entry must be created to hold the pricing and availability of the part.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **IList<PriceCheckAlternatePart>** **AlternateParts** | List of alternate parts that can be substituted for this part. (Set by the catalog - Empty in the PriceCheckRequest) |
| **bool** **Found** | Indicates whether the catalog recognizes this part. (Set by the catalog) |
| **IList<Location> Locations** | A list of locations where the part may be ordered from. (Set by the catalog - Empty in the PriceCheckRequest) This first location in this list will be considered as the primary location. |
| **string** **PartNumber** | Part Number of part. (Max length 20) (Can be updated during price check) |
| **string** **ManufacturerLineCode** | Manufacturer Line Code for the part. (Max length 6) (Can be updated during price check) |
| **string** **ManufacturerName** | Manufacturer Name for the part. (Max length 30) (Can be updated during price check) |
| **string** **Description** | Description for the part. (Max length 50) (This property should not be updated unless it is passed in as a blank string) |
| **decimal** **QuantityRequested** | Quantity of part requested to be ordered. |
| **Location** **SelectedLocation** | The selected location for the part. (Set by Manager SE in the PriceCheckRequest) |
| **string** **Status** | Custom Status of a processed part. Reserved for future use. |
| **string** **Metadata** | Custom Metadata which may include Supplier information. (Max length 4000 characters) (Optional - Available when HostApiLevel 3+) |

## PriceCheckAlternatePart

PriceCheckAlternatePart represents an alternate part that can be substituted for an original part

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **IList<Location> Locations** | List of locations where the alternate part may be ordered from |
| **string** **PartNumber** | Part Number of alternate part. (Max length 20) |
| **string** **ManufacturerLineCode** | Manufacturer Line Code for the alternate part. (Max length 6) |
| **string** **ManufacturerName** | Manufacturer Name for the alternate part. (Max length 30) |
| **string** **Description** | Description for the alternate part. (Max length 50) |
| **decimal** **QuantityRequested** | Quantity of alternate part requested. |
| **string** **Status** | Custom Status of a processed part. Reserved for future use. |
| **string** **Metadata** | Custom Metadata which may include Supplier information. (Max length 4000) (Optional - Available when HostApiLevel 3+) |

## Location

A Location object references a location where a part may be ordered from. It is used in a PriceCheckPart and PriceCheckAlternatePart to indicate the location of the part along with the pricing and availability.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **Id** | Id of the location (as determined by the catalog) (Max length 8 characters) |
| **string** **Name** | Name of the location (Max length 20 characters) |
| **decimal** **QuantityAvailable** | Quantity of this part item available at this location. |
| **decimal** **UnitList** | Suggested price for a single unit of this item |
| **decimal** **UnitCost** | Cost for a single unit of this item |
| **decimal** **UnitCore** | Core charge for this item |
| **string** **SupplierName** | Uniquely identifies (within this catalog) the source supplier for this part. This may be displayed to the user. (Max length 25) (Optional - Available when HostApiLevel 3+) |
| **string** **ShippingDescription** | Description of Shipping Cost. (Max length 25 characters) (Optional - Available when HostApiLevel 3+) |
| **decimal** **ShippingCost** | Shipping cost for this part. (Optional - Available when HostApiLevel 3+) |

## HostData

HostData represents information that the host application makes available to catalogs.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **ApplicationTitle** | Title of the host application. |
| **string** **ApplicationVersion** | The first 3 components of the current version of the host application. |
| **decimal** **LaborRate** | The default hourly rate the host application uses to calculate labor items |
| **int HostApiLevel** | The ApiLevel that Manager SE supports. For versions less than 3 this will not be passed to the catalog. |

## Vendor

Vendor represents a parts Vendor. The Code and Name are configured in the host application. The Qualifier represents custom information as defined by the catalog – this is passed to all calls, but only vendor setup/configuration can make changes to this that will be saved.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **Code** | A short code which uniquely identifies the vendor |
| **string Name** | Name of the vendor |
| **string** **Qualifier** | String that can be used by the catalog to store custom vendor information. **Important:** This value can only be changed and persisted when returned from VendorSetup. Any other time, changes to this field will be discarded. |

## Vehicle

Vehicle represents the user selected vehicle for use in filtering catalog part searches. The vehicle is defined in the host application and then passed to the catalog on a call to **GetCatalog**. Not all properties in the Vehicle object are guaranteed to be defined. Catalogs should attempt to identify the vehicle based on available information (starting with VIN), and fallback to other values given (e.g. VIN is not right, check Year / Make / Model and/or Aces Id’s). Important: This object can also be omitted in older versions of Manager SE.

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **Vin** | VIN (Vehicle Identification Number) of the vehicle |
| **int** **Year** | Model year of the vehicle |
| **string** **Make** | Make of the vehicle |
| **string** **Model** | Model of the vehicle |
| **string** **SubModel** | Submodel of the vehicle |
| **string** **Body** | Body style of this vehicle (like: "sedan" or "wagon") |
| **string** **Transmission** | The type of transmission installed on this vehicle. |
| **string** **Engine** | Description of engine in this vehicle |
| **string** **DriveType** | Drive train type for this vehicle |
| **string** **Brake** | Description of brakes installed on this vehicle |
| **string** **Gvw** | GVW (Gross Vehicle Weight) of this vehicle |
| **int AcesId** | ACES identifier for this vehicle. (i.e. ACES VehicleId) |
| **int AcesBaseId** | ACES (Aftermarket Catalog Enhanced Standard) BaseVehicleID identifier for this vehicle |
| **int AcesEngineId** | ACES Engine identifier for this vehicle (i.e. ACES EngineBaseId) |
| **int AcesEngineBaseId** | Same as the AcesEngineId (i.e. ACES EngineBaseID) |
| **int AcesEngineConfigId** | ACES EngineConfigID |
| **int AcesSubmodelId** | ACES SubModelID |

***TrackingRequest***

Request object for returning basic tracking information and a URL to show in external browser (e.g. Chrome/Firefox/Safari/Edge)

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **HostData** **HostData** | Information that the host application makes available to catalogs. (See HostData object definition) |
| **Vendor** **Vendor** | Represents a parts Vendor (See Vendor object definition) |
| **string OrderTrackingNumber** | A tracking number that short message (180 chars or less) about status |

***TrackingResponse***

Response object for returning basic tracking information and a URL to show in external browser (e.g. Chrome/Firefox/Safari/Edge)

|  |  |
| --- | --- |
| PROPERTY | MEANING |
| **string** **ExternalTrackingUrl** | Valid URL – either https or http that shows the order status when viewed in external browser |
| **string StatusDisplay** | **Optional:** A short message (180 chars or less) about status |

Price Check / Order Parts Logging / Debugging

* Request & Response messages (raw JSON) is logged to: C:\Users\<currentuser>\AppData\Local\M1-SK\<CatalogName>.OrderParts.log and PriceCheck.log. Useful when determining what was sent & received to your catalog from Manager SE.
* Catalog Driver is your first debugging tool
* However, when there is a question about behavior and Manager SE, Fiddler can be used on Local system to capture API calls (enable Fiddler HTTPS capture)
* Additionally, the Dev Tools can be activated from the window’s application menu to debug your JavaScript / web sources.
* If Vendor Setup or Transfer Parts in Go Shopping does not appear to work, check the JavaScript dev tools for any console JavaScript issues, and/or run DebugView (from Sys Internals) to capture any possible JavaScript to Managed code conversion errors in browser process. Also, if running Manager SE, check C:\ProgramData\M1-SK\ShopStreamShell.log for any logged errors