Smith Cart API
v.6.70

Developers Guide

Table of Contents

I. Smith Cart API 4
II. QueryString Methods 4
- A. Product Listing Page 4
  - 1. Filtering Products by Department 4
  - 2. Filtering Products by Category 5
  - 3. Filtering Products by Vendor/Registrant 5
- B. Cart Page 5
  - 1. Adding a Single Product to the Cart 5
  - 2. Adding a Product with Quantity to the Cart 6
  - 3. Adding Multiple Products with Quantity to the Cart 6
  - 4. Adding a Product with Variants to the Cart 6
  - 5. Adding Product Variants with Quantities to the Cart 7
  - 6. Adding Product Variants with User Entered Amounts to the Cart 7
  - 7. Adding a Product to the Cart by Sku 8
  - 8. Adding a Product with a Custom Field to the Cart 8
  - 9. Passing a Coupon to the Cart 8
  - 10. Adding a Product with Serial Number to the Cart 10
- C. Product Details Page 11
  - 1. Product ID 11
  - 2. ProductID and VariantID 11
  - 3. ProductID and Multiple VariantID’s 11
  - 4. Custom Textbox Field (tbfield) 12
- D. Tracking Affiliate Sales using a Link 12

III. Session Variables Supported 13
- A. Product Listing Page 13
  - 1. Filtering Products by Product ID 13

IV. Dot Net API 13
- A. Visual Studio Configuration 13
- B. Add To Shopping Cart 14
  - 1. Input Parameters 14
  - 2. Output Parameters 16
  - 3. Sample Code 16
- C. List Products 16
  - 1. Input 17
  - 2. Output 17
  - 3. Sample Code 20
- D. Get Customers By Date 21
  - 1. Input 21

2. Output ... 21
3. Sample Code ... 23

E. Get Store Order Details By Date ... 23
1. Input ... 23
2. Output ... 24
3. Store Order Fields ... 24
4. Store Order Detail Fields ... 25
5. Sample Code ... 26

F. List Payment History By Date ... 27
1. Input ... 27
2. Output ... 27
3. Sample Code ... 28

I. Smith Cart API

The Smith Cart API is used to execute common eCommerce operations in Smith Cart from other 3rd party DNN modules or DNN modules that you develop.

You can integrate with the Smith Cart API in the following ways:

- QueryString methods
- Dot Net API
- Restful Web Service (Coming Soon)

II. QueryString Methods

SmithCart supports a number of different querystring operations to support common ecommerce functions. The querystring is the part of the URL that contains data to be passed to SmithCart to perform a dynamic operation in the cart. The following sections outline the querystring operations supported in SmithCart.

A. Product Listing Page

1. Filtering Products by Department

The product listing page of the cart allows you to filter products by department by passing the "DepartmentID" in the querystring.

For example:

http://www.mydomain.com/ProductList.aspx?departmentid=1

When DepartmentID is passed in the querystring the products will be filtered by department. Additionally, if you have the category module on the page categories will be filtered by department. This feature will allow you to create your own separate department landing page using the DNN Html module with links or images to filter your products by department.

2. Filtering Products by Category

The product listing page of the cart allows you to filter products by category by passing the "CategoryID" in the querystring.

For example:

http://www.mydomain.com/ProductList.aspx?categoryid=1

When CategoryID is passed in the querystring, products will be filtered by category. This feature will allow you to create your own separate category landing page using the DNN Html module with links or images to filter your products by category.

3. Filtering Products by Vendor/Registrant

The product listing page of the cart allows you to filter products by Vendor ID or Registrant by passing the "VendorID" in the querystring.

For example:

http://www.mydomain.com/Default.aspx?vendorid=5

When the vendor id is passed in the querystring the products or gift registry will be filtered by vendor.

Please Note: *This feature requires the Smith Gift Registry module. For more information on the please see the Smith Gift Registry module page.*

B. Cart Page

SmithCart supports the following querystring operations to dynamically add items to your cart.

1. Adding a Single Product to the Cart

To add a product directly to your cart, the "productid" must be passed in the querystring. Follow these steps to obtain the correct link for hyper linking:

1. Navigate to the BuyNow module Store Admin Menu and click the “Manage Products” button to view your product listing page. Note the product id for the product you want to add to the cart using the querystring. (If you don’t have a product added yet, add your product from the product setup screen).

2. Navigate to the BuyNow module product listing page and click the “View Cart” button to go to the cart page.

3. Cut and paste the url from the browser window and paste it into notepad or your favorite editor.

4. Add “?productid=1” to the end of the url where product id is the product id you obtained in step 1 above.

Example URL:

http://www.mydomain.com/Cart/Default.aspx?productid=1

2. Adding a Product with Quantity to the Cart

To add a single product with quantity to the cart via querystring the format of the url is the following:

http://www.mydomain.com/Cart/Default.aspx?productid=1&qty=2

3. Adding Multiple Products with Quantity to the Cart

SmithCart supports the ability to pass multiple “ProductIDs” and “Quantities” to dynamically add products to the cart using the querystring.

The format of the URL when passing multiple product ids and quantities in the querystring is the following:

https://www.mydomain.com/BuyNow/cart.aspx?ProductID=5,6,12,19,28&Qty=1,2,1,2,1

4. Adding a Product with Variants to the Cart

SmithCart supports adding products with variants directly to your cart by adding “productid” and “variantid” in the querystring. Follow these steps to obtain the correct link for hyper linking:

1. Navigate to the “Manage Products” page and locate the “Product ID” of the product you want to add to the cart using the querystring.

2. Navigate to the “Product Variant Setup” page and locate the “Variant ID” of the product variant you want to add to the cart using the querystring.

3. The format of the URL when passing multiple product ids and variant ids in the querystring is the following:

https://www.mydomain.com/BuyNow/cart.aspx?Variants=1~3~5;14~34~35~38

A semi colon is used to delimit “productids” and the tilde is used to delimit “variantids”. This format allows you to add multiple products and multiple variants and associate products with a specific set of variants.

5. Adding Product Variants with Quantities to the Cart

SmithCart supports adding products with variants and quantities directly to your cart by passing the “ProductID”, “VariantID” and “Quantity” in the querystring. Follow these steps to obtain the correct link for hyper linking:

Example URL:

https://www.mydomain.com/BuyNow/cart.aspx?Variants=1~2390;2~1&Qty=2,3

6. Adding Product Variants with User Entered Amounts to the Cart

SmithCart supports adding products with variants to the cart and passing a user entered amount.

Example URL:

https://www.mydomain.com/BuyNow/cart.aspx?Variants=1~3&Price=50.00

Please Note: In order to pass a user entered amount in the querystring you must have "User Entered Amounts" enabled in the product setup screen for the product you want to add.

7. Adding a Product to the Cart by Sku

SmithCart supports adding products directly to your cart by passing the "sku" in the querystring. When product sku is passed in the querystring to the cart page, the cart will look up the product by sku and add the product to the cart. Follow these steps to obtain the correct link for hyper linking:

1. Navigate to the "Manage Products" page and locate the "Sku" of the product you want to add to the cart using the querystring.

2. The format of the URL when passing the product sku in the querystring is the following:

https://www.mydomain.com/BuyNow/cart.aspx?Sku=ABC123

To add a product to the cart using product sku and quantity format the url as follows:

https://www.mydomain.com/BuyNow/cart.aspx?Sku=ABC123&Qty=3

8. Adding a Product with a Custom Field to the Cart

To add product to the cart and pass a custom field via querystring the format of the url is the following:

https://www.mydomain.com/BuyNow/cart.aspx?ProductID=1&tbfield=Project99

The "tbfield" passed in the url will be added to the end of the product name similar to how product variants are added to the cart. If the same base product is added to the cart but with different values in the tbfield, the products will show as separate products (line items) in the cart and invoice.

9. Passing a Coupon to the Cart

All of the querystring operations described above also support passing a coupon code to the cart in the querystring. When coupon code is passed in the querystring the cart will look up the coupon code and apply the appropriate discount you have defined in the "Manage Coupon" screen. Coupon codes can apply to entire order or to a specific product.

<u>Single Product and Coupon</u>
To add a single product to the cart using a product id with a coupon format the url as follows:

https://www.mydomain.com/BuyNow/cart.aspx?ProductID=2&Coupon=50dollarsoff

<u>Multiple Products and Coupon</u>
To add multiple products to the cart using product ids with a coupon format the url as follows:

https://www.mydomain.com/BuyNow/cart.aspx?ProductID=5,6&Coupon=50dollarsoff

<u>Product Variants and Coupon</u>
To add products with variants to the cart with a coupon format the url as follows:

https://www.mydomain.com/BuyNow/cart.aspx?Variants=1~3~5;14~34~35~38&Coupon=50dollarsoff

<u>Product Sku and Coupon</u>
To add a product to the cart using product sku with a coupon format the url as follows:

https://www.mydomain.com/BuyNow/cart.aspx?Sku=ABC123&Coupon=50dollarsoff

In the BuyNow module settings coupons section the following radiobutton list affects coupon querystring operations:

- Disable Coupons - When selected coupons are not allowed on the cart page. The coupon textbox is hidded and coupons are not allowed to be passed in the querystring.
- Show Coupons - When selected a coupon textbox is displayed on the cart page and coupons can be passed in the querystring to apply discounts.
- Hide Coupons - When selected the coupon textbox on the cart page is hidden but coupons are still allowed to be passed in the querystring.

10. Adding a Product with Serial Number to the Cart

SmithCart supports adding products with serial numbers directly to your cart by adding "ProductID" and "SerialNumber" in the querystring. See the section titled Serial Number Management for more information on the serial number feature in SmithCart and how to create serial numbers in your store.

The value you pass for serial number can be a license key or any string field generated from an external system. Upon successful order the serial number passed in the querystring is saved with the order detail record of the order and associated with a specific product ordered. After an order is completed the serial number is marked as "Assigned" so it is not used by another order and is viewable in the manage order detail screen.

Follow these steps to obtain the correct link for hyper linking:

1. Navigate to the "Manage Products" page and locate the "Product ID" of the product you want to add to the cart using the querystring.

2. Navigate to the "Serial Number Management" screen in product setup and locate the "SerialNumber" you want to add to the cart using the querystring.

3. The format of the URL when passing a product and serial number in the querystring is the following:

Product and Serial Number
To add a product with a serial number to the cart format the url as follows:

https://www.mydomain.com/BuyNow/cart.aspx?ProductID=2&SerialNumber=123AE894582

Product Sku and Serial Number
To add a product to the cart using product sku and serial number format the url as follows:

https://www.mydomain.com/BuyNow/cart.aspx?Sku=ABC123&SerialNumber=123AE894582

C. Product Details Page

The Product Details module supports the following querystring variables:

- ProductID
- ProductID and VariantID
- ProductID and Multiple VariantID’s
- Custom Textbox field (tbfield)

1. Product ID

The product details page of the cart allows you link directly to the product details page by passing the product id in the querystring.

For example:
http://www.yoursite.com/productdetails/Default.aspx?productid=1

When the product id is passed in the querystring the products details page will be loaded with the product information associated with the product id passed in the querystring.

2. ProductID and VariantID

The product details page supports linking directly to the product details page by passing the product id and the variant id in the querystring.

For example:
http://www.yoursite.com/productdetails/Default.aspx?productid=1&variantid=4

When the product id and variant id are passed in the querystring the products details page will be loaded with the product and variant information associated with the product id and variant id passed in the querystring.

3. ProductID and Multiple VariantID’s

The product details page supports linking directly to the product details page by passing the product id and multiple variant id’s in the querystring.

For example:
http://www.yoursite.com/productdetails/Default.aspx?productid=1&variantid=4&variantid=7

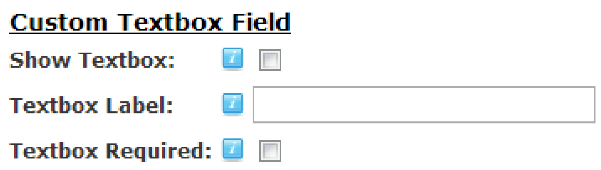
When the product id and multiple variant id's are passed in the querystring the products details page will be loaded with the product and variant information associated with the product id and variant id's passed in the querystring.

4. Custom Textbox Field (tbfield)

The product details page supports linking directly to the product details page by passing the product id and a custom textbox value in the querystring. To pass a value in the querystring to populate a custom textbox field on the product details page format your URL as follows:

http://www.yoursite.com/productdetails/Default.aspx?productid=1&tbfield=Project99

Passing the "tbfield" querystring will auto populate the product details page custom textbox field. To turn on the display of the product details page custom textbox field navigate to the product details module settings custom textbox field section and check the box titled "Show Custom Textbox".


Custom Textbox Field

Show Textbox: ☐

Textbox Label:

Textbox Required: ☐

Next, you will need to go to the product setup screen and check the box titled "Show Custom Fields" to turn on the display of custom fields at the product level.

D. Tracking Affiliate Sales using a Link

SmithCart supports passing the "AffiliateID" in the querystring to any page in your portal that you have added the Smith Affiliate Tracking module to. For more information on the Smith Affiliate Tracking module see the section titled Affiliate Tracking Module.

To pass an affiliate id in the querystring format your URL as follows:

http://www.mydomain.com/Default.aspx?AffiliateId=27

When the affiliate id is passed in the querystring to your store and an order is successful, SmithCart will save the affiliate id with the order giving you the ability to track affiliate sales and pay out commissions.

III. Session Variables Supported

SmithCart supports a number of different session variables to dynamically pass data into various pages in your store. Session variables contain data stored in memory and can be used to be pass dynamic data to SmithCart. This is useful if you have are using another 3rd party DNN module or have developed your own DNN module that you want to integrate with SmithCart. The following session variables are supported in SmithCart.

A. *Product Listing Page*

1. Filtering Products by Product ID

The product listing page of the cart allows you to filter products by Product ID by passing a comma separated list of "Product ID's" in a session variable titled "SearchPID".

For example:

Session["SearchPID"] = "1,2,7,8"

When the session variable "SearchPID" is populated with a comma separated list of "Product ID's" and the user navigates to the product listing page, the product listing page will be filtered to display all the products contained in the "SearchPID" session variable.

IV. Dot Net API

A. *Visual Studio Configuration*

In order to use the SmithCart API you need to create a reference in Visual Studio to the following dll located in the \bin folder off the root of your DNN installation:

SmithBuyNow.dll

This SmithBuyNow.dll should already exist in your \bin folder if you have installed SmithCart.

B. Add To Shopping Cart

Use the “AddToShoppingCart” API function is used to add items to the Smith Shopping Cart.

1. Input Parameters

The following are the input parameters for the "AddToShoppingCart" API function call.

Field	Description	Type	Rqrd.
productID	The unique identifier of your SmithCart product. The productID can be found in the Smith_Products table in SQL Server.	integer	Yes
productName	The name of your SmithCart product. The productName field can be found in SQL Server Smith_Products table in the “ModelName” field. Any data passed in this field will be saved with the order.	nvarchar(200)	Yes
quantity	The number of items you would like to add to cart.	decimal(10, 2)	Yes
price	The price of the product. The price of the product can be passed in two ways: 1. Using the “UnitCost” field found in the Smith_Products table in SQL Server. 2. User Enter Amount – Any price you pass.	decimal(18, 2)	Yes

minPartialAmount	The minimum partial amount the customer is allowed to pay for this product. Pass -1 to indicate no partial amount allowed.	decimal(18, 2)	Yes
weight	The weight of your product. The weight field can be found in the Smith_Products table in SQL Server.	decimal(12, 2)	Yes
inBundle	If this product belongs to a bundle.	boolean	No
bundleID	The BundleID of the product. The BundleID field can be found in the Smith_ProductBundle table in SQL Server.	integer	No
smithDiscounts	The type of discount you want to apply to the product. The following discount types are supported: • ProductDiscounts – Product discounts by amount. • ProductDiscountPercent – Product discount percent. • ProductDiscountByUser – Product discounts by user. • CategoryDiscounts – Category discounts • PriceClassDiscounts – Price class discounts. For no discounts pass an empty string.	string	No
attributeName	The name of your custom attribute. On successful order the value passed in this field is saved in your SQL Server StoreOrderDetailAttributes table in the "Name" field. The attributeName field can also be viewed in the SmithCart Manage Orders Screen.	nvarchar(150)	No

attributeValue	The value of your custom attribute. On successful order the value passed in this field is saved in your SQL Server StoreOrderDetailAttributes table in the “Value” field. The attributeName field can also be viewed in the SmithCart Manage Orders Screen.	nvarchar(200)	No

2. Output Parameters

The “AddToShoppingCart” function returns No Parameters.

3. Sample Code

The following c# code is an example of how to call AddToShoppingCart method:

```
using Smith.DNN.Modules.BuyNow;

//Instantiate SmithCart API
SmithCart cart = new SmithCart();

//Call AddToShoppingCart method
cart.AddToShoppingCart(1, "My Product Name", 1, 25, -1, 12, false, 5, "",
"AttendeeID", "25");
```

C. List Products

The “ListIProducts” API function is used to retrieve a listing of all products by DotNetNuke Portal ID.

1. Input

The following are the input parameters for the " ListIProducts " function.

Field	Type
PortalId	Integer

2. Output

The " ListIProducts " function returns an “ICollection” of type SmithProduct.

The SmithProduct object contains the following properties:

Field Name	**Type**
ProductID	int
CategoryID	int
SubCategoryID	int
RelatedProductID	int
CategoryName	string
Manufacturer	string
SortOrder	int
ModelNumber	string
ModelName	string
UnitCost	decimal
SalePrice	decimal
SaleStartDate	DateTime
SaleEndDate	DateTime
WholesalePrice	decimal
MemberPrice	decimal

PriceUnits	string
PriceClass	string
TaxableAmount	decimal
TaxRate	double
QuantityOnHand	int
Weight	decimal
Width	int
Height	int
Length	int
Items	int
MinOrderQty	int
MaxOrderQty	int
LeadTime	string
ShipCost	decimal
ShipCodes	string
ShipText	string
DownloadURL	string
AutoGUID	bool
UrlGUID	string
EncryptURL	bool
ExcludeCoupon	bool
RequireCoupon	bool
AddDnnRole	string
RoleExpireDays	string
RemoveDnnRole	string
BundleID	int
MasterBundleID	int
UsePriPriceWght	bool

Summary	string
Description	string
Description2	string
Description3	string
Description4	string
Description5	string
TabName1	string
TabName2	string
TabName3	string
TabName4	string
TabName5	string
Share	bool
Featured	bool
Upsell	bool
GiftCard	bool
Achieved	bool
UserEnteredAmount	bool
TaxExempt	bool
ChargeHandling	bool
RequireLogin	bool
ShowPDUDF	bool
ShowProductRole	bool
HideThumbnail	bool
HidePrice	bool
HideQuantity	bool
HideProduct	bool
CompareGroup	string
AffiliateUrl	string

ProductDetailUrl	string
TitleTag	string
DescriptionTag	string
KeywordTag	string
SeoUrl	string
Recurring	bool
SubscriptionInterval	string
RecurringOccurances	int
RecurringStartDate	string
RecurStartDateSetup	string
ExcludeVariantRecur	bool
EnableTrial	bool
TrialInterval	string
TrialOccurences	string
TrialAmount	decimal
Booking	bool
BookingUnits	string
CreatedByUser	string
CreatedDate	DateTime
LogicallyDeleted	bool

3. Sample Code

The following c# code is an example of how to call the ListIProduct function:

```
using Smith.DNN.Modules.BuyNow;

var products
      = from product in ProductsController.ListIProducts(PortalId)
      select product as IProduct;
```

```
foreach (IProduct product in products)
{
     //Get data from object
     int pid = product.ProductID;
     string sku = product.ProductSKU;

     //get the rest of the data
     //add your code here
}
```

D. *Get Customers By Date*

The “GetCustomersByDate” API function is used to retrieve a listing of all customers by create date.

1. Input

The following are the input parameters for the " GetCustomersByDate " function.

Field Name	Type
DateCreated	DateTime
PortalId	Integer

2. Output

The "GetCustomersByDate " function returns an “ICollection” of type CustomerInfo.

The CustomerInfo object contains the following properties:

Field Name	Type

CustomerID	int
FirstName	string
LastName	string
CompanyName	string
Address1	string
Address2	string
City	string
Country	string
State	string
Zip	String
HomePhone	string
CellPhone	string
WorkPhone	string
Email	string
DNNLogin	string
Contact	boolean
IPAddress	string
UDF	string
UDTB1	string
UDTB2	string
UDDate	datetime
BuyerRole	string
ClubCode	string
AnniversaryDate	datetime
TaxID	string
DateCreated	datetime
PortalID	integer

3. Sample Code

The following c# code is an example of how to call the GetCustomersByDate function:

```
using Smith.DNN.Modules.BuyNow;

var customers
      = from customer in RegistrationController.GetCustomersByDate(DateCreated,
PortalId)
      select customer as ICustomer;

      foreach (ICustomer customer in customers)
      {
           //Get data from object
           int cid = customer.CustomerID;
           string FirstName = customer.FirstName;

           //get the rest of the data
           //add your code here
      }
```

E. *Get Store Order Details By Date*

The “GetStoreOrderDetailsByDate” API function is used to retrieve a listing of all orders and order details by create date.

1. Input

The following are the input parameters for the "GetStoreOrderDetailsByDate" function.

Field Name	Type
DateCreated	DateTime
PortalId	Integer

2. Output

The "GetStoreOrderDetailsByDate" function returns a dataset of the following fields:

3. Store Order Fields

Field Name	Type
OrderID	integer
CustomerID	integer
OrderDate	datetime
PayHistID	integer
ShippingTotal	money
TaxTotal	money
HandlingCharge	money
Surcharge	money
OrderTotal	String
ShipFirstName	String
ShipLastName	string
ShipAddress1	string
ShipAddress2	string
ShipCity	string
ShipState	string
ShipZipcode	string
ShipCountry	string
Status	string
ShipMethod	string
AffiliateId	string
CouponId	integer
Discount	money

MemberDiscount	money
MemberRole	string
TrackingNumber	string
SpecialInstructions	string
ShipDate	datetime
ShipCarrier	string
ResidentialStatus	string
PDTextbox	string
UDF1	string
UDF2	string
UDDropDown	string
UDDate	datetime
UDDate1	datetime
UDDate2	datetime
DNNProfileField	string
DNNUserID	string
IPAddress	string
ModifiedBy	string
DateModified	datetime
PortalID	integer

4. Store Order Detail Fields

Field Name	Type
OrderDetailID	integer
ProductID	integer
ProductName	string

ProductSKU	string
Quantity	string
UnitCost	money
Shipped	boolean
Returned	boolean
TaxRate	float
Commission	money
SerialNumber	string
WebServiceData	string
OrderDetailGUID	string
VendorID	integer
DateUpdated	datetime
PortalID	integer

5. Sample Code

The following c# code is an example of how to call the GetStoreOrderDetailsByDate function:

```
using Smith.DNN.Modules.BuyNow;

private ProductsController pc = new ProductsController();

//get orderDetail records from database
DataSet ds = pc.GetStoreOrderDetailsByDate(OrderDate, PortalId);
DataTable objDT = ds.Tables[0];

//Start For Loop - Get Product Data
foreach (DataRow row in objDT.Rows)
{
       string productName = row["ProductName"].ToString();
       // Get the rest of your data here
}
```

F. List Payment History By Date

The “ListIPaymentHistory” API function is used to retrieve a listing of all payment history records by create date.

1. Input

The following are the input parameters for the "ListIPaymentHistory" function.

Field Name	Type
fromDate	DateTime
toDate	DateTime
PortalId	Integer

2. Output

The "ListIPaymentHistory" function returns an “ICollection” of type PaymentHistoryInfo.

The PaymentHistoryInfo object contains the following properties:

Field Name	Type
PayHistID	integer
CustomerID	integer
OrderID	integer
GiftCardID	integer
PayType	string
PayDate	datetime
Amount	money
Quantity	decimal

AcctNo	string
ExpRoute	string
CVV	String
BillingName	string
BillingAddress	string
BillingCity	string
BillingState	string
BillingZip	string
BillingCountry	boolean
Coupon	string
TransID	string
Notes	string
Status	string
AuthCode	string
InvoiceNo	string
PONumber	string
Success	boolean
IPAddress	string
ModifiedBy	string
DateModified	datetime
PortalID	integer

3. Sample Code

The following c# code is an example of how to call the ListIPaymentHistory function:

```
using Smith.DNN.Modules.BuyNow;

var payhists
```

```
      = from payhist in RegistrationController.ListIPaymentHistory(fromDate,
toDate, PortalId)
      select payHist as PaymentHistoryInfo;

      foreach (PaymentHistoryInfo payhist in payhists)
      {
           //Get data from object
           int phid = payhist.PayHistID;
           string BillingName = payhist.BillingName;

           //get the rest of the data
           //add your code here
      }
```