Use Advanced Dynamic Message Content

When you enable and use Dynamic Message Content , you can insert placeholders into a message that will be replaced in real-time with the corresponding values for the receiving visitor. You configure this on the *Promoted Content* tab in *Message Settings* . However, in some cases, the *Promoted Content* tab will not provide the configurability needed for very specific or advanced requirements and you need to use advanced dynamic message content (ADMC) syntax instead, which is added in the message HTML, accessible selecting the *Message Source Code* tab.

This Section Explains

This section details how to use advanced dynamic message content.

Sections in this Article

Articles in this Section

- Advanced Dynamic Content Cheat Sheet

Advanced Dynamic Message Content should be used by experienced developers only. If you are unsure of any requirements or need further assistance with Advanced Dynamic Message Content, please contact Evergage Support.

When you enable and use Dynamic Message Content, you can insert placeholders into a message that will be replaced in real-time with the corresponding values for the receiving visitor. You configure this on the *Promoted Content* tab in *Message Settings*. However, in some cases, the *Promoted Content* tab will not provide the configurability needed for very specific or advanced requirements and you need to use advanced dynamic message content (ADMC) syntax instead, which is added in the message HTML, accessible selecting the *Message Source Code* tab. For example, you must use ADMC if you want to:

- Access the Product, Article, Blog, or Category the visitor is currently viewing
- Access the Products currently in the visitor's cart
- Access more than one specific item by ID
- Specify a minimum and maximum number of items leveraging the #foreach loop syntax to generate a list up to the maximum specified
- Access the items the visitor has viewed, sorted by recency, and not restricted to the current visit
- Match the least viewed or purchased items globally
- Sort matching Items by name, price, or other fields for display
- Match items based on statistics from a lookback period other than one week (i.e. the past day)
- Only match items that have fields meeting certain conditions (e.g. products that have image URLs, or products with descriptions containing “clearance”)

This article will give an overview of ADMC. For more information, please see *Advanced Dynamic Content Cheat Sheet* .

NOTE

You must enable campaign content to use ADMC. For more information, please see *Use Dynamic Message Content* .

Promoting Items

When the *Promoted Content* tab is used to configure promoted Items, any matching items that are referenced in the message content are automatically marked as having been promoted. However, any items referenced by ADMC syntax will not be marked since they can also be used as inputs to query filters or for other purposes, not strictly as promoted content.

When using advanced dynamic message content, you can use the ***$tools.promote*** function to mark promoted items. This function would wrap the ADMC query you added to the message HTML:

```
$tools.promote($product in $items)
```

OR

```
$itemOptions.promote($product in $items)
```

Alternatively, to mark all Items returned by a query as promoted, you can use *$itemOptions.promote*. See the *Item Options* section below for details.

Currently Viewed Item or Category

To access the item (Product, Article, or Blog) the visitor is currently viewing, use the *$page.item* variable. To access the category the visitor is currently viewing (or the category of the Item currently being viewed), use the *$page.category* variable. The *$page.category* variable can either be used directly within the message content or passed to a global or user query using *$itemOptions.withCategory($page.category.id)*. You can also use *$page.brand* and *$page.gender* to include the brand and gender. If there is more than one applicable category, brand, or gender, Evergage will show the first one found on the page (based on how the page is coded). For more information, please see the *Item Options* section below.

Products Currently in Cart

To show data about the current items contained in the visitor's shopping cart, use ***$tools.user.order.current()*** in the syntax shown below:

```
Your cart has a total value of $tools.user.orders.current().totalValue dollars.
```

Or to show only the list of unique products in the visitor's cart, use ***$tools.user.order.currentItems()*** in the syntax shown below:

```
Your cart contains the following products:
<ul>
#foreach ( $product in $tools.user.orders.currentItems() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

Specific Items

A single, specific item can be promoted using the **Static** sub-tab of the *Promoted Content* tab, but to promote more than one specific Item, use the ***$tools.global.products.findById*** function in the syntax shown below:

```
#set( $levi501Jeans = $tools.global.products.findById("501") )

Check out the amazing sales on this product:
<a href=”${levi501Jeans.url}”>${levi501Jeans.name}</a>
```

You can replace “products” with “categories”, “brands”, or other groupings configured in your Evergage Catalog, to show items in those groupings. For example, to show brands, use ***$tools.global.brands.findById.*** To show multiple items, use ***$tools.global.products.findByIds*** which works similarly and accepts a list of up to 10 IDs. This function returns all items found in order by ID. Items that are not found will be skipped:

```
#set( $products = $tools.global.products.findByIds( ["501", "993"] ) )

Check out the amazing sales on these products:
<ul>
   #foreach( $product in $products )
   <li><a href=”${product.url}”>${product.name}</a></li>
   #end
</ul>
```

User’s Recently Viewed Items

Refer to the *Item Options* section at the end of this document for options that can be passed to the “viewed” function to specify minimum/maximum results, sorting, filtering, and other options.

To show items the visitor has recently viewed, with last viewed item first, the $tools.user.products.viewed function can be used. Additional item types can accessed by replacing “products” with “categories”, “brands”, or other groupings configured in your Evergage Catalog. The lookback time period cannot be adjusted and is always *this visit*.

For example, ***$tools.user.brands.viewed*** :

```
These are the products you’ve viewed most recently:
<ul>
#foreach ( $product in $tools.user.products.viewed() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

User’s Most Viewed/Purchased Items

Refer to the *Item Options* section at the end of this document for options that can be passed to the “viewCount”, “viewTime”, and “purchaseCount” functions to specify minimum/maximum results, sorting, filtering, lookback time period, and other options.

To show items that the visitor has viewed the most times, use the ***$tools.user.products.viewCount*** function:

```
These are the products that you have viewed the most times:
<ul>
#foreach ( $product in $tools.user.products.viewCount() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

To show items that the visitor has spent the most time viewing, use the ***$tools.user.products.viewTime*** function. Additional item types can be shown by replacing “products” with “categories”, “brands”, or other groupings configured in your Evergage Catalog. For example, ***$tools.user.brands.viewTime*** :

```
These are the products that you have spent the most time viewing:
<ul>
#foreach ( $product in $tools.user.products.viewTime() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

To show items that the visitor has purchased the most times, use the ***$tools.user.products.purchaseCount*** function:

```
These are the products that have been viewed the most times:
<ul>
#foreach ( $product in $tools.user.products.purchaseCount() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

Most Viewed/Purchased Items Globally

To access items that have been viewed the most times globally, use the ***$tools.global.products.viewCount*** function:

```
These are the products that have been viewed the most times:
<ul>
#foreach ( $product in $tools.global.products.viewCount() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

To access Items that visitors have spent the most time viewing globally, use the ***$tools.global.products.viewTime*** function. Additional item types can be shown by replacing “products” with “categories”, “brands”, or other groupings configured in your Evergage Catalog. For example, ***$tools.global.brands.viewTime*** :

```
These are the products that users have spent the most time viewing:
<ul>
#foreach ( $product in $tools.global.products.viewTime() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

To access Items that been purchased the most times globally, use the ***$tools.global.products.purchaseCount*** function:

```
These are the products that have been viewed the most times:
<ul>
#foreach ( $product in $tools.global.products.purchaseCount() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

Most Recently Published Items Globally

Refer to the *Item Options* section at the end of this document for options that can be passed to the "publishedDate” function to specify minimum /maximum results, sorting, filtering, lookback time period, and other options.

To find items were recently published, sorted by published date, use the ***$tools.global.products.publishedDate*** function:

```
These are our newest products:
<ul>
#foreach ( $product in $tools.global.products.publishedDate() )
   <li><a href=”${product.url}”>${product.name}</a></li>
#end
</ul>
```

Item Options

The ***$tools.(user|global).products.(viewed|viewCount|viewTime|purchaseCount)*** query functions described above can all be passed a set of item options to refine the results. Item options can specify minimum/maximum results, lookback time period, filtering, sorting, etc.

Item options are created by calling multiple functions on the $itemOptions object. For example, to create a set of item options that shows the top three items people spent the most time viewing today, and sorted by product name, you could use the following call:

```
#set( $opts = $itemOptions.minItems(3).maxItems(3).time($time.today()).displaySort($productSort.name()) )
```

These options would then be passed to any of the query functions. For example:

```
#set( $products = $tools.global.viewTime($opts) )
```

Min/Max Results

By default, a message containing advanced dynamic queries will only be rendered if there is at least 1 result, and a maximum of 5 results will be returned. You can increase the minimum using ***$itemOptions.minItems(n)*** and the maximum using ***$itemOptions.maxItems(n)*** . You cannot increase the maximum beyond 10.

For example, to render the message only if there are at least 3 results and to limit the number of results shown to 3, use:

```
$itemOptions.minItems(3).maxItems(3)
```

Lookback Time Period

By default, queries will consider item data from the past week. To use a different lookback period, use the $itemOptions.time function. For example, to consider today only, use:

```
$itemOptions.time($time.today())
```

The following functions are provided by the ***$time*** singleton object:

- *today* - today
- *thisWeek* - current calendar week (with Monday as first day of week)
- *thisMonth* - current calendar month
- *since(date)* - since the specified date; date format is “mm/dd/yyyy” (e.g. “07/07/2017”)
- *lastNDays(n)* - last n days, including today; lastNDays(1) would mean today and yesterday
- allTime - do not define a specific date range

Filtering by Category, Brand or Gender

To filter results to contain products with a particular category, use ***$itemOptions.withCategory(categoryId)*** . For example, to filter by the category the visitor is currently viewing, use:

```
$itemOptions.withCategory($page.category.id)
```

To filter by the visitor's most purchased category, use:

```
$itemOptions.withCategory($tools.user.categories.purchaseCount()[0])
```

To filter results to contain only products with a specific brand, use $itemOptions.withBrand(brandId). For example, to filter by visitor's most purchased brand, use:

```
$itemOptions.withCategory($tools.user.brands.purchaseCount()[0])
```

To filter by gender:

```
$itemOptions.withGender($page.gender)
```

Filtering Based on Item Fields

Results can be filtered based on Item field values using any of the following conditions:

- Field has a value
- Field does not have a value
- Field’s value contains a substring
- Field’s value does not contain a substring

To only include Items that have image URLs, use:

```
$itemOptions.whereImageUrl().exists()
```

To only include Items that have “clearance” in their descriptions, use:

```
$itemOptions.whereDescription().contains(“clearance”)
```

The following functions are available, each corresponding to a different field:

- whereName
- whereCreated
- wherePublished
- whereDescription
- whereUrl
- whereImageUrl
- wherePriceDescription (Products only)

And the following functions correspond to the four types of conditions:

- exists
- doesNotExist
- contains(substring)
- doesNotContain(substring)

> **NOTE**
>
> The functions "contains" and "doesNotContain" are case-insensitive

Query Result Order

By default, query results are returned in descending order with the most popular or most recently viewed Items listed first. To reverse the order, and return the least popular or least recently viewed Items first, use:

```
$itemOptions.descending()
```

Display Sorting

To sort the results for display, use the ***$itemOptions.displaySort*** function. Use ***$productSort*** for products, ***$articleSort*** for articles, and ***$blogSort*** for blogs. For example, to sort the results by product name, use:

```
$itemOptions.displaySort($productSort.name())
```

The following functions are available on ***$productSort*** , corresponding to different product fields:

- name
- price

The following functions are available on ***$articleSort*** and ***$blogSort*** , corresponding to different article/blog fields:

- name
- publishedDate

By default, display sorting is done in ascending order. To use descending order instead, you can use the ***descending*** function:

```
$itemOptions.displaySort($productSort.name().descending())
```

Include Items Being Viewed or Products in Cart

By default, items the visitor is currently viewing are excluded from query results. To include them, use:

```
$itemOptions.includeItemsBeingViewed()
```

By default, Products that are currently in the visitor's cart are excluded from query results. To include them, use:

```
$itemOptions.includeItemsInCart()
```

Exclude Items Previously Viewed or Purchased

By default, items the visitor has viewed can be included in query results. To exclude any item the visitor has viewed, use:

```
$itemOptions.excludeViewedItems()
```

By default, Products that the visitor has purchased can to be included in query results. To exclude any product the visitor has purchased, use:

```
$itemOptions.excludePurchasedItems()
```

Promoting Items

To mark all of the items the query returns as promoted, use:

```
$itemOptions.promote()
```

Formatting Numbers

Typically, prices are the only numbers you need formatted. There are two options for how to format them.

Option 1

The simpler option formats the price with the default dataset currency symbol using the default dataset locale:

```
$tools.formatPrice(${item.price})
```

If the default dataset currency is US dollars and the default dataset locale is US, the formatted currency would look like:

```
$1,234.50
```

And for Euros with a French locale:

```
1 234,50 €
```

Option 2

The other option is to format the number only using a fixed number of digits after the decimal point. Then, the currency symbol or word can be added as part of the message. For example:

```
$tools.formatNumber(${item.price}, 0)
```

Which would display as:

```
1,234
```

Using Contextual Bandit

To fetch promotions and render the images for the proper content zone and dimension as dictated by the message's dynamic content, use the following syntax:

```
<!--#foreach( $item in $items )-->
<a href="${item.url}" style="text-align: center; display: block;margin: auto;">
    <img src="$tools.bandit.imageToServe(${item})" class="">
</a>
<!--#end-->
```