This Document

Conventions

- Line breaks will be inserted in the middle of MTL tags for clarity throughout this document, but line breaks are NOT LEGAL inside MTL tags. When/if you copy sample code in this document into templates, REMOVE ALL LINE BREAKS that appear between instances of "<mcc" and ">".
- Non-commentary text, including code to be typed into a template and sample template output

```
will look like this
```

Majors & Minors

None

Standard Majors (and sub-majors)

None

Configuring a template

TOC is the default template type.

If you navigate to http://classifieds.[your domain].com/classifieds-bin/classifieds

And you have one active, default, level 1, TOC template in your property (presumably directly under Classifieds,) that's the template you'll get. This is called "default derivation."

But, if other active default level 1 TOC templates exist in the template tree, the default derivation will not succeed. If you change the level of the other TOC templates to something other than 1, the default derivation will succeed.

Template organization

None

Accessing a template

Here are the things ptemp looks for when deciding which template to serve up:

1) tid
2) category_id and template_type
3) ad_id and template_type==CLIPBOARD
4) ad_number and template_type
5) category_number and template_type
6) category and template_type
7) classification and template_type
8) none: assume template_type=TOC and classification=classifieds

Property

The assignment of $property based on URL will be muddled if more than one property has the same third level domain assigned to the property's "Classifieds URL" attribute.

This attribute is set/edited/deleted by editing the property record in Template Manager.

You can freely access another property's data and/or templates using tp=[other property] and/or property=[other property/properties] in your URL.

However, the properties must have parallel classification/category structures for this to work well. Specifically, each property with classifieds data has a classification named "classifieds" so this URL:

```
…?classification=classifieds&tp=propA&property=propB…
```

should always work. But this:

```
…?classification=Exotic+Birds&tp=propA&property=propB…
```

will only work if both propA and propB have majors (or sub-majors) named "Exotic Birds." Similarly, this:

```
…?classification=classifieds&category_number=309&tp=propA&property=propB…
```

Will only work if both properties have a category numbered "309" under the specified classification

Classifieds.Category

None

Classifieds.Ad - basic query

If "&page=[X]" appears in your URL, you'll get page number X of ads, whether or not you've qualified your query by page=$page.

You can override this behavior by hard-coding into your query the specific page of results you want, for example, page=1.

Keywords

The value of a URL key named keywords, i.e. `&keywords=blue`, will automatically be populated into **both** `$keywords` and `$keyword_query`.

The lifecycle of a liner

None

Date, Status

None

Id versus Number

None

Clipboard

This functionality doesn't work with ptemp (versus classifieds-bin/classifieds) unless your template uses ptemp's cookie functions to explicitly grab the cookie value.

Ptemp and classifieds-bin/classifieds are two paths to the same thing, and for the most part, there's little difference between how they act. But using the classifieds-bin/classifieds path triggers different behavior if the template type is "clipboard."

If you use classifieds-bin/classifieds and temp_type=clipboard, the cart gets the cookies passed from the browser, and if those cookies contain a key named "ad_id" it puts the value into the $ad_id variable. Clipboard is the only template type this behavior is triggered for.

Again, if you'd like to use ptemp for the same functionality, you'll have to do something like this:

```
<mcc $ad_id=web.cookie(ad_id)>
```

Display Ads

None

Prices, Features

Feature qualifiers do exist for the classifieds.ad query.

Classifieds.ad features are unique in MTL: it's not how you qualify your query for features (because you can't,) it's how you name the inputs in your search form.

Specifically, if you name search inputs following this pattern:

```
feature_[feature name]
feature_[modifier]_[feature name]
```

Their values will automatically be used as selection criteria when Forge retrieves the ads.

That means that you don't have to qualify your query by feature to narrow your results down by a feature value. Conversely, there's no way you can prevent your search from being narrowed; it happens "automagically."

Again, this situation is unique in MTL. No where else in MTL does it matter what your inputs/variables are named.

Ordering

Be aware that advanced sorting i.e. orderby=colname(val1,val2), explained in "Basic Concepts of MTL" does not work for the classifieds.ad query. It was not implemented because the result sets for classifieds.ad are usually very large, and the larger the result set, the more CPU-intensive that particular function becomes.

Also, while the number of orderby values you can use is virtually unlimited, the number of features you can orderby is limited to four.

i.e.:

```
orderby=col1,col2,col3,…,col78,…,col94,&etc
```

But scattered in that unlimited number of columns you'd like your results ordered by, you can have only four features.

Customers and Real Estate

None

Classifieds.Ad_Customer

None

Classifieds.Customer_Attribute

None

Classifieds.Ad_Feature

None