Interchange Programmer Variables Reference

Table of Contents

1. Interchange Special Variables

Interchange defines some special variables which control behavior. They can be of several types, and the conventions for using them depend on whether you have based your catalog and server on the standard "foundation" distribution.

We will distinguish between these by calling intrinsic variables CORE variables, noting the distribution variables as DISTRIBUTION, and noting the foundation catalog practices as FOUNDATION.

"Variable" configuration file definitions

Defined in interchange.cfg or catalog.cfg with the `Variable` configuration directive, these are accessed with:

```
  Access in ITL with           From
  -----------------------      -------------------
  __VARNAME__                  (catalog.cfg only)
  @_VARNAME_@                  (catalog.cfg, falls back to interchange.cfg)
  @@VARNAME@@                  (interchange.cfg only)
  [var VARNAME]                (catalog.cfg only)
  [var VARNAME 1]              (interchange.cfg only)
  [var VARNAME 2]              (catalog.cfg, falls back to interchange.cfg)

  Embedded Perl                From
  -----------------------      -------------------
  $Variable->{VARNAME}         (catalog.cfg only)
  $Tag->var('VARNAME')         (catalog.cfg only)
  $Tag->var('VARNAME', 1)      (interchange.cfg only)
  $Tag->var('VARNAME', 2)      (catalog.cfg, falls back to interchange.cfg)

  Global subs / Usertags       From
  -----------------------      -------------------
  $::Variable->{VARNAME}       (catalog.cfg only)
  $Tag->var('VARNAME')         (catalog.cfg only)
  $Tag->var('VARNAME', 1)      (interchange.cfg only)
  $Tag->var('VARNAME', 2)      (catalog.cfg, falls back to interchange.cfg)
  $Global::Variable->{VARNAME} (interchange.cfg only, only in Global code)
```

Variables set with Variable are not normally modified dynamically, though you can do it as a part of the Autoload routine or in other code. They will not retain the value unless DynamicVariables is in use.

Scratch

User scratch variables are initialized whenever a new user session is created. They start with whatever is defined in the `ScratchDefault` directive in catalog.cfg; otherwise they are not defined.

```
  Access in ITL with           Attributes
  -----------------------      -------------------
  [scratch varname]            Displays
  [scratchd varname]           Displays and deletes

  Embedded Perl                From
  -----------------------      -------------------
  $Scratch->{varname}          Accessor
  $Session->{scratch}{varname} Equivalent
```

```
Global subs / Usertags       From
-----------------------      -------------------
$::Scratch->{varname}          Accessor
$::Session->{scratch}{varname} Equivalent
```

They can be set in several ways:

```
Set in ITL with              Attributes
-----------------------      -------------------
[set varname]VAL[/set]       Sets to VAL, no interpretation of ITL inside
[seti varname]VAL[/seti]     Sets to VAL, interprets ITL inside
[tmpn varname]VAL[/tmpn]     Sets to VAL, no ITL interpretation, temporary
[tmp  varname]VAL[/tmp]      Sets to VAL, interprets ITL inside, temporary

Embedded Perl                From
-----------------------      -------------------
$Scratch->{varname} = 'VAL'; Sets to VAL
$Tag->tmp(varname);          Set as temporary, must set value afterwards.

Global subs / Usertags       From
-----------------------      -------------------
$::Scratch->{varname}='a'    Sets to a
```

CGI

CGI variables are the raw data which comes from the user.

WARNING: It is a security risk to use these variables for display in the page.

You can use them for testing without worry, though you should never set their value into a database or display on the page unless you have processed them first, as they can have arbitrary values. The most common security risk is displaying HTML code, which allows remote scripting exploits like cookie−stealing.

```
    [calc]
        ####  DO NOT DO THIS!!!!
        my $out = $CGI->{varname};
        return $out;
    [/calc]
```

That will transform the value. If you wish to output a safe value but keep the actual value intact, do:

```
    [calc]
        ####  This is safe, makes value safe for rest of page
        my $out = $Tag->cgi( { name => 'varname', filter => 'entities' } );
        ####  This is safe too, doesn't transform value
        my $other = $Tag->filter($CGI->{varname}, 'entities');

        ### Now you can return stuff to the page
        return $out . $other;
    [/calc]
```

The access methods are:

```
Access in ITL with                 Attributes
-----------------------            -------------------
[cgi varname]                      Doesn't stop ITL code, don't use!
```

```
[cgi name=varname filter=entities] Use this for safety

Embedded Perl                 From
-----------------------       -------------------
$CGI->{varname}               Don't use for output values!
```

They can be set as well.

```
Set in ITL with                        Attributes
-----------------------                -------------------
[cgi name=varname set="VAL"]           Sets to VAL, VAL can be ITL, shows VAL
[cgi name=varname set="VAL" hide=1]    Sets to VAL, VAL can be ITL, no output

Embedded Perl                 From
-----------------------       -------------------
$CGI->{varname} = 'VAL';      Sets to VAL, next access to [cgi varname]
                              shows new value
```

Values

User form variables are initialized whenever a new user session is created. They start with whatever is defined in the <ValuesDefault> directive in catalog.cfg; otherwise they are not defined except as called out in other configuration directives, i.e. the obsolete DefaultShipping.

```
Access in ITL with            Attributes
-----------------------       -------------------
[value varname]               Displays

Embedded Perl                 From
-----------------------       -------------------
$Values->{varname}            Accessor
```

They can be set as well, though the normal method of setting is from user input via form. If Interchange receives an action which performs the update of values (by default `go` or `return`, `refresh`, or `submit`), the value of CGI variables will be transferred to them subject to other considerations (FormIgnore settings, credit card variables, etc., discussed below).!block example

```
Set in ITL with                          Attributes
-----------------------                  -------------------
[value name=varname set="VAL"]           Sets to VAL, VAL can be ITL, shows VAL
[value name=varname set="VAL" hide=1]    Sets to VAL, VAL can be ITL, no output

Embedded Perl                            Attributes
-----------------------                  -------------------
$Values->{varname} = 'VAL';              Sets to VAL, next access to
                                         [value varname] shows new value
```

Session variables

You can also directly access the user session. Normally you don't set these values unless you are an experienced Interchange programmer, but there are several values that are frequently used.

One example is `username`, which holds the logged−in user's username.

```
Access in ITL with            Attributes
-----------------------       -------------------
```

```
[data session username]      Displays

Embedded Perl                From
-----------------------      -------------------
$Session->{username}         Accessor
```

They can be set as well, but if you are experienced enough to contemplate doing these things you will easily be able to figure it out.

Values not transmitted from CGI

The following variables are never copied from CGI:

```
        mv_todo
        mv_todo.submit.x
        mv_todo.submit.y
        mv_todo.return.x
        mv_todo.return.y
        mv_todo.checkout.x
        mv_todo.checkout.y
        mv_todo.todo.x
        mv_todo.todo.y
        mv_todo.map
        mv_doit
        mv_check
        mv_click
        mv_nextpage
        mv_failpage
        mv_successpage
        mv_more_ip
        mv_credit_card_number
        mv_credit_card_cvv2
```

You can define more with the `FormIgnore` catalog.cfg directive.

Global program variables

If you are programming a GlobalSub or global UserTag, you have access to all Interchange facilities including all the preset variables and configuration directives.

The `Global` package is used to hold variables that are set at program start and whose value is retained.

The `Vend` package is used for variables that might be set at some point during program execution, but that will always be reset to undefined at the end of the transaction.

One example is `$Vend::Cookie`, which holds the raw cookie value sent by the user.

If you are going to set or access these variables, you should be getting your documentation from the source code.

2. IC Variables

2.1. Standard global (interchange.cfg) Variable values

CGIWRAP_WORKAROUND
IMAGE_MOGRIFY

CGIWRAP_WORKAROUND

Used in `Vend/Dispatch.pm`. Implemented to Fix Cobalt/CGIwrap problem. If set to 1, it removes the scriptname from the URL pathinfo.

IMAGE_MOGRIFY

Used in `code/SystemTag/image.tag`. Specifies the location of mogrify command. If ImageMagick is installed, you can display an arbitrary size of the image, creating it if necessary.

LANG

MV_DOLLAR_ZERO

Used in `scripts/interchange.PL.` This parameter specifies how Interchange will be displayed in ps command. This may not work on BSD based Kernels.

MV_FILE

Used in `lib/Vend/File.pm.` This is the filename of the most recently returned contents. This variable is not set in `interchange.cfg`, but is set by interchange while interchange is runnning.

MV_GETPPID_BROKEN

Used in `lib/Vend/Server.pm.` If configured, the server uses a syscall(MV_GETPPID_PROKEN) instead of the perl function getppid() to find the parent PID. If `MV_GETPPID_BROKEN` is set to 1, the system uses syscall(64).

MV_MAILFROM

Used in `lib/Vend/Util.pm.` If configured, it specifies the default email user address if it has not been set in catalog.cfg or variables.txt. Overrides the `MailOrdersTo` directive.

```
my $from = $::Variable->{MV_MAILFROM}
                || $Global::Variable->{MV_MAILFROM}
                || $Vend::Cfg->{MailOrderTo};
```

MV_NO_CRYPT

Used in `lib/Vend/UserDB.pm and lib/Vend/Util.pm.` If configured, it disables the use of crypt or md5 hashing of passwords serverwide.

MV_PAGE

Used systemwide. This is the relative path of the page being served without the suffix. This value is not set in interchange.cfg, but is set by interchange while interchange is running. This is often referenced as `@@MV_PAGE@@.`

MV_PREV_PAGE

Used systemwide. This is the relative path of the last (previous) page that was served without the suffix. This value is not set in `interchange.cfg`, but is set by interchange while interchange is running. This is often referenced as `@@MV_PREV_PAGE@@`.

MV_SESSION_READ_RETRY

Used in `lib/Vend/Session.pm.` This variable specifies the number of times that interchange will attemt to read the session file before failing.

The default value is `5`.

UI_BASE

Used systemwide. This variable specifies the relative path to the admin interface.

The default value is `'admin'.`

UI_IMAGE_DIR

Used within the admin and by `code/SystemTag/image.tag`. This variable specifies the relative path to the admin images and CSS definitions. It is set in `dist/lib/UI/ui.cfg`.

The default value is `'/interchange-5/'`.

UI_IMAGE_DIR_SECURE

Used within the admin and by `code/SystemTag/image.tag`. This variable specifies the relative path to the admin images and CSS definitions while connecting via SSL. It is set in `dist/lib/UI/ui.cfg`.

The default value is `'/interchange-5/'`.

UI_SECURITY_OVERRIDE

Used in `dist/lib/UI/Primitive.pm`. If configured and no UI_ACCESS_TABLE found, then it will return that there is an acl set. This would allow you to test acls See sub `ui_acl_enabled()` for more details.

2.2. Standard catalog (catalog.cfg) Variable values

ACTIVE_SESSION_MINUTES

Used in `code/UI_Tag/dump_session.coretag`, `lib/UI/pages/admin/genconfig.html` and `lib/UI/pages/admin/show_session.html`. This variable overrides the SessionExpire directive.

ADL_COMPONENT

Used in `lib/UI/ui.cfg`. This variable can be overriden by ADL_COMPONENT_TEMPLATE. Apparently not used anywhere.

The default is:

```
    [page href="admin/content_editor" form=|
        ui_name=[control component]
        ui_type=component
    |][loc]edit[/loc]&nbsp;[control component]</A>
```

ADL_COMPONENT_TEMPLATE

Used in `lib/UI/ui.cfg`. This variable can be used to override ADL_COMPONENT. Apparently not used anywhere.

ADL_ITEM

Used in `lib/UI/ui.cfg`. The default is:

```
  <a href="[area
                        href=admin/item_edit
                        form=|
                                item_id=[item-code]
                                ui_return_to=index
                        |
                ]">[loc]edit[/loc]&nbsp;[loc]item[/loc]</A>
```

ADL_ITEM_TEMPLATE

Used in `lib/UI/ui.cfg`. This variable can be used to override ADL_ITEM. Apparently not used anywhere.

ADL_MENU

Used in `foundation/templates/components/product_flyout`, `foundation/templates/components/product_tree` and `lib/UI/ui.cfg`.

The default value is:

```
        $::Variable->{ADL_MENU} = $::Variable->{ADL_MENU_TEMPLATE} || <<EOF;
  <a class="[control link_class]"
        href="[area
                        href=admin/menu_editor
                        form=|
                                qmenu_name=[either][control menu_name][or][var MV_PAGE 1][/either
                        |
                ]">[loc]edit[/loc]&nbsp;[loc]menu[/loc]</A>
```

ADL_MENU_TEMPLATE

Used in `lib/UI/ui.cfg`. This variable can be used to override ADL_MENU. Apparently not used anywhere.

ADL_PAGE Used in the foundation templates LEFTONLY_BOTTOM,LEFTRIGHT_BOTTOM, and NOLEFT_BOTTOM and `lib/UI/ui.cfg`. This variable defines how the admin edit page links are displayed.

The default is:

```
 [page href="admin/content_editor" form="
                        ui_name=[var MV_PAGE 1][var ADL_SUFFIX]
                        ui_type=page
                ")[loc]edit[/loc]&nbsp;[loc]page[/loc]</A>
[page href="[var MV_PAGE 1]" form="
                        ui_mozilla_edit=1
                ")[loc]show&nbsp;tags[/loc]</A>
```

ADL_MENU_TEMPLATE

Used in `lib/UI/ui.cfg`. This variable can be used to override ADL_PAGE.

ADL_SUFFIX

Used in `lib/UI/ui.cfg`. It defaults to the value of the [HTMLsuffix] directive. It specifies the page suffix for links coming from ADL_PAGE at the bottom of the foundation template files.

BACKUP_DIRECTORY

Used in `code/UI_Tag/backup_database.coretag`. This variable will override where the backup_database tab places the database backups. By default, the backups are placed in `<VendRoot`/backup>.

BAR_LINK_TEMPLATE

Used in the `bar_link` subroutine found in `catalog_before.cfg`. It defines how the links are displayed in the `foundation/templates/components/category_horizontal`, `foundation/templates/components/category_vert_toggle` and `foundation/templates/components/category_vertical` The default value is configured in `variables.txt`. The default is:

```
        <A HREF="$URL$" CLASS="barlink">$ANCHOR$</A>
```

CREATE_COMMAND_MYSQL

Used in `code/UI_Tag/xfer_catalog.coretag`. It allows one to override the command that is used create tables under mysql. The default is `'mysqladmin create %s'`.

CREATE_COMMAND_PG

Used in `code/UI_Tag/xfer_catalog.coretag`. It allows one to override the command that is used create tables under PostgresSQL. The default is `'createdb %s'`.

CUSTOMER_VIEW_LARGE

Used in `lib/UI/pages/admin/customer.html`. It allows one to select not to build huge lists of customers every time you access the customer tab. The default is `0`.

CYBER_ID

Used in `eg/globalsub/authorizenet`, `eg/globalsub/signio` and `lib/Vend/Payment/ECHO.pm`. It specifies the vendor's ID for communicating with a payment gateway.

CYBER_PORT

Used in `eg/globalsub/authorizenet`, `eg/globalsub/signio` and `lib/Vend/Payment.pm`. Specifies the port over which to communicate with the gateway server.

CYBER_PRECISION

Used in `eg/globalsub/authorizenet`, `eg/globalsub/signio`, `lib/Vend/Payment.pm` and `lib/Vend/Payment/ECHO.pm`. It defines the precision of (the number of decimal points) used with the vendor gateway. If not defined, the default is `2`.

CYBER_SCRIPT

Used in `eg/globalsub/authorizenet`. It defines the path of the program on the payment gateway. if not set, it uses the default of `'/gateway/transact.dll'`. It is overriden by MV_PAYMENT_SCRIPT

CYBER_SECRET

Used in `eg/globalsub/authorizenet`, `eg/globalsub/signio`, and `lib/Vend/Payment/ECHO.pm`. It defines the password the vendor used for autorization to the payment gateway.

CYBER_SERVER

Used in `eg/globalsub/authorizenet`, `eg/globalsub/signio`, and `lib/Vend/Payment/ECHO.pm`. It overrides the address of the payment gateway.

DUMP_COMMAND_MYSQL

Used in `code/UI_Tag/xfer_catalog.coretag`. It allows one to override the command that is used create tables under mysql.

The default is `'mysqldump --add-drop-table'`.

DUMP_COMMAND_PG

Used in `code/UI_Tag/xfer_catalog.coretag`. It allows one to override the command that is used create tables under PostgresSQL. The default is `'pg_dump -c -O'`.

ECHO_PAYMENT_ID

Used in `lib/Vend/Payment/ECHO.pm`. Specifies your ECHO ID.

ECHO_PAYMENT_PRECISION

Used in `lib/Vend/Payment/ECHO.pm`. Specifies the number of digits of precision for the gateway.

ECHO_PAYMENT_SECRET

Used in `lib/Vend/Payment/ECHO.pm`. Specifies the password used to indentify the vendor.

ECHO_PAYMENT_SERVER

Used in `lib/Vend/Payment/ECHO.pm`. Specifies the address of the payment gateway.

FORUM_EMAIL_NOTIFY

Used in `foundation/pages/forum/reply.html`. It is initially set to `__MVC_EMAILSERVICE__` when the catalog is initially created.

FORUM_SUBMIT_EMAIL

Used in `foundation/pages/forum/submit.html`. It specifies the email address that the forum submission should be sent to.

LANG

Used in `lib/Vend/Dispatch.pm` and `lib/Vend/File.pm`.

MV_AUTOLOAD

Used in `lib/Vend/Interpolate.pm`. It specifies the value to be placed in the beginning of the html if the $Vend::PageInit is defined and not 0.

MV_COMPONENT_DIR

Used in `code/UserTag/component.tag`. If defined, it specifies a directory location where the components will be located. If not defined, `code/UserTag/component.tag` looks in `templates/components`.

MV_COUNTRY_FIELD

Used in `lib/Vend/Interpolate.pm`. If defined, it specifies the field to be used in tax/vat calculations. If undefined, the value `'country'` is used.

MV_COUNTRY_TABLE

Used in `lib/Vend/Interpolate.pm`. If defined, it specifies the table to be used in tax/vat calculations. If undefined, the value `'country'` is used.

MV_COUNTRY_TAX_FIELD

Used in `lib/Vend/Interpolate.pm`. If defined, it specifies the field used to look up the tax. If undefined, the value `'tax'` is used.

MV_CREDIT_CARD_INFO_TEMPLATE

Used in `sub build_cc_info()` in `/lib/Vend/Order.pm`. If not defined, the following template is used:

```
                join("\t", qw(
                        {MV_CREDIT_CARD_TYPE}
                        {MV_CREDIT_CARD_NUMBER}
                        {MV_CREDIT_CARD_EXP_MONTH}/{MV_CREDIT_CARD_EXP_YEAR}
                        {MV_CREDIT_CARD_CVV2}
                )) . "\n";
```

MV_DEFAULT_SEARCH_DB

Used in `lib/Vend/Scan.pm`. Specifies that an unspecified (default) search will be a db search, not a text search. It is set to 1 in variable.txt

MV_DEFAULT_SEARCH_FILE

Used in `lib/Vend/Config.pm`, `lib/Vend/Glimpse.pm` and `lib/Vend/TextSearch.pm`. It specifies the file to be used for text searches by default. It is set to products in `catalog_before.cfg` and `foundation/catalog.cfg`.

MV_DEFAULT_SEARCH_TABLE

Used in `lib/Vend/Config.pm`, `lib/Vend/DbSearch.pm` and `lib/Vend/RefSearch.pm`. It specifies the table to be used for searches by default. It is set to products in `catalog_before.cfg` and `foundation/catalog.cfg`.

MV_ERROR_STD_LABEL

Used in `code/SystemTag/error.coretag`. If defined, it overrides the default error imessage in the stdlabel field. If undefined, the following template is used.

```
    <FONT COLOR=RED>{LABEL} <SMALL><I>(%s)</I></SMALL></FONT>
    [else]{REQUIRED <B>}{LABEL}{REQUIRED </B>}[/else]
```

MV_NO_CRYPT

Used in `lib/Vend/UserDB.pm` and `lib/Vend/Util.pm`. If configured, it disables the use of crypt or md5 hashing of passwords for the individual catalog.

MV_OPTION_TABLE_MAP

Used in `lib/Vend/Config.pm`, `lib/Vend/Data.pm` and `lib/Vend/Options/Old48.pm`. It is a quoted space−delimited list of fields in the form of "field1=field2" to map options into.

MV_OPTION_TABLE

Used in `foundation/products/variable.txt`, `lib/UI/pages/admin/item_option_phantom.html`, `lib/UI/pages/admin/item_option_old.html`, `lib/UI/ui.cfg`, `lib/Vend/Config.pm`, `lib/Vend/Options.pm` and `lib/Vend/Options/Simple.pm`. If the Interchange variable MV_OPTION_TABLE is not set, it defaults to "options", which combines options for Simple, Matrix, and Modular into that one table. This goes along with foundation and construct demos up until Interchange 4.9.8.

MV_PAYMENT_ID

Used in `eg/globalsub/authorizenet`, `eg/globalsub/signio`, `/foundation/catalog.cfg`, `/foundation/products/variable.txt`, `lib/Vend/Payment/AuthorizeNet.pm`, `lib/Vend/Payment/BoA.pm`, `lib/Vend/Payment/ECHO.pm`, `lib/Vend/Payment/PSiGate.pm`, `lib/Vend/Payment/Signio.pm`, `lib/Vend/Payment/Skipjack.pm`, `lib/Vend/Payment/TestPayment.pm`, `lib/Vend/Payment/WellsFargo.pm`, `lib/Vend/Payment/iTransact.pm`. Specifies your merchant ID for your payment gateway.

MV_PAYMENT_MODE

Payment Gateway Which payment processor module you wish to use. The default value is not set. Valid values are : authorizeneti, boa, echo, mcve, psigate,signio, skipjack, trustcommerce, testpayment, wellsfargo, itransact and linkpoint.

MV_PAYMENT_PRECISION

Specifies the number of digits of precision for the gateway.

MV_PAYMENT_SECRET

Specifies the password used to indentify the vendor.

MV_PAYMENT_SERVER

Specifies the address of the payment gateway.

MV_PAYMENT_TEST

Used in `eg/globalsub/authorizenet`, `lib/Vend/Payment/AuthorizeNet.pm` and `lib/Vend/Payment/PSiGate.pm`. Specifies that the gateway is in testing mode.

MV_SHIP_ADDRESS_TEMPLATE Used in `lib/Vend/Interpolate.pm`. Overrides the default template used in `tag_address()`.

If not set, the template used is:

```
                        $template .= "{company}\n" if $addr->{"${pre}company"};
                        $template .= <<EOF;
{address}
{city}, {state} {zip}
{country} -- {phone_day}
```

MV_SHIP_MODIFIERS

Used in `shipping()` in `lib/Vend/Ship.pm`.

MV_STATE_REQUIRED

Used in `sub _multistate()` in `lib/Vend/Order.pm`.

MV_STATE_TABLE

Used in `sub tax_vat()` in `lib/Vend/Interpolate.pm`. Specifies an alternate table with tax information by state. If undefined, the subroutine uses `'state'`.

MV_STATE_TAX_FIELD

Used in `sub tax_vat()` in `lib/Vend/Interpolate.pm`. Specifies an alternate field with tax information by state. If undefined the subroutine uses `'tax'`.

MV_TAX_CATEGORY_FIELD

Used in `sub tax_vat()` in `lib/Vend/Interpolate.pm`. If undefined, the subroutine uses `'tax_category'`.

MV_TAX_TYPE_FIELD

Used in `sub tax_vat()` in `lib/Vend/Interpolate.pm`. Specifies an alternate field with tax information by state. If undefined the subroutine uses `'tax_name'`.

MV_TREE_TABLE

Used in `lib/UI/pages/admin/menu_editor.html`, `lib/UI/pages/admin/menu_loader.html` and `lib/Vend/Menu.pm`. It allows one to specify another table other that `'tree'`.

MV_USERDB_REMOTE_USER

Used in `sub check_security()` in `lib/Vend/Util.pm`. Enabling this variable allows anyone logged in to override all existing ALCs.

MV_VALID_PROVINCE

Used in `sub _state_province()` in `lib/Vend/Order.pm`. Allows you to supply an alternate string to override the standard province validation.

MV_VALID_STATE

Used in `sub _state_province()` `lib/Vend/Order.pm`. Allows you to supply an alternate string to override the standard State validation.

MV_ZIP_REQUIRED

Used in `sub _multizip()` in `lib/Vend/Order.pm`.

ORDER_VIEW_LARGE

Used in `lib/UI/pages/admin/order.html`. It allows one to select not to build huge lists of orders every time you access the orders tab. The default is `0`.

PAGE_TITLE_NAME

Used in `lib/UI/ContentEditor.pm`. It allows you to override the page title in preview mode. If not configured, the page title will be `'page_title'`.

PUBLISH_NO_PAGE_ROOT

Used in `lib/UI/ui.cfg`. It allows one to prevent any publishing of pages in the admin to the root of pages directory.

PUBLISH_QUIT_ON_RCS_ERROR

Used in `lib/UI/ui.cfg`. It allows one to not publish new pages in the admin if the page cannot be succesfully checked into RCS.

PUBLISH_TO_PREVIEWS

Used in `lib/UI/ui.cfg`. It allows one to publish new pages in the admin into a preview directory.

RESTORE_COMMAND_MYSQL

Used in `code/UI_Tag/xfer_catalog.coretag`. It allows one to override the command that is used create tables under mysql. The default is `'mysql'`.

RESTORE_COMMAND_PG

Used in `code/UI_Tag/xfer_catalog.coretag`. It allows one to override the command that is used create tables under PostgresSQL. The default is `'createdb %s'`.

SERVER_NAME

Used in `code/UI_Tag/xfer_catalog.coretag`, `foundation/etc/ship_notice`, `lib/UI/pages/admin/transfer_catalog.html`, `lib/Vend/Server.pm` and `scripts/makecat.PL`. It specifies the domain name of your catalog.

TAXCOUNTRY

Used in `sub fly_tax()` in `foundation/include/checkout/tax_popup` and `lib/Vend/Interpolate.pm`. Allows you to supply an alternate string of valid countries to override the standard Country validation.

THEME_IMG_DIR

It allows you to specify the location of the images in the foundation specified themes.

UI_ACCESS_KEY_LIMIT

Used in `code/UI_Tag/list_keys.coretag` and `lib/UI/Primitive.pm`. It allows you to define the number of keys returned. By default 500 (primary) keys are returned.

UI_ACCESS_TABLE

It allows you to specify the UserDB file to be used for access to the admin. It is set to `'access'` by `catalog_before.cfg`.

UI_BACKUP_TABLES

Used by `lib/UI/pages/admin/dbdownload.html`. It specifies the tables to be backed up by `lib/UI/pages/admin/dbdownload.html` It is set in variable.txt to: `'affiliate area cat country inventory locale merchandising options order_returns orderline pricing products state survey transactions tree userdb variants'`

UI_COMPONENT_DIR

Used by `lib/UI/pages/admin/content_publish.html` and `lib/UI/ContentEditor.pm`. By default, it is set to `'templates/components'` in variable.txt

UI_DBCONFIG

Used in `lib/UI/pages/admin/gentable.html`. If defined it adds the option `'Config'` to gentable.html in the admin.

UI_ERROR_PAGE

Used in `code/UI_Tag/flex_select.coretag`. IT allows one to override the admin error page from `'admin/error'`.

UI_IMG

Used in `lib/UI/pages/admin/content_push.html`, `lib/UI/pages/admin/customer_mailing.html`, `lib/UI/pages/admin/help.html`, `lib/UI/pages/admin/order.html`, `lib/UI/pages/include/templates/ui_type1`, `lib/UI/pages/include/templates/ui_type2`, `lib/UI/pages/include/templates/ui_type3`, `lib/UI/pages/include/templates/ui_type5`, and `lib/UI/vars/UI_STD_HEAD`. It specifies where the images for the admin are kept. It is set in `lib/UI/vars/UI_STD_HEAD`.

UI_LARGE_TABLE

Used in `code/UI_Tag/flex_select.coretag` and `lib/UI/pages/admin/pref_select.html`. It specifies that flex−select should not use ra=all if UI_LARGE_TABLE is set.

UI_META_LINK

Used in `lib/UI/pages/admin/menu_editor.html`, `lib/UI/pages/admin/preferences.html`, and `lib/Vend/Table/Editor.pm`. It determines whether edit metadata links are enabled by default. It is set to 1 by default in variables.txt.

UI_META_SELECT

Used in `code/UI_Tag/flex_select.coretag`.

UI_META_TABLE

Allows you to specify an alternate table where metadata is kept.

UI_SECURE

Used in `lib/UI/pages/admin/flex_select.html`, `code/UI_Tag/flex_select.coretag`, `lib/UI/pages/admin/menu_editor.html`, `lib/UI/pages/admin/order.html`, `lib/UI/vars/UI_STD_HEAD`, `lib/Vend/Table/Editor.pm`. If set, it determines whether or not to force UI into secure mode.

UI_TEMPLATE_DIR

Used by `lib/UI/pages/admin/content_publish.html` and `lib/UI/ContentEditor.pm`. t specifies the directory where the admin templates are kept. It is set to `'templates'` by default in variables.txt.

UPS_COUNTRY_REMAP

Used in `code/UserTag/ups_query.tag` and `lib/Vend/Ship/QueryUPS.pm`.

```
        # Remap Monaco to France for UPS
        Variable UPS_COUNTRY_REMAP   MC=FR
```

UPS_ORIGIN

Used in `code/UserTag/fedex_query.tag`, `lib/UI/pages/admin/ship_edit.html`, `lib/Vend/Ship/QueryUPS.pm`. It sets a default value for the shipping origin.

UPS_QUERY_MODULO

Used in `code/UserTag/ups_query.tag` and `lib/Vend/Ship/QueryUPS.pm`. If shipping aggregation is used, it allows you to override the weight in which aggregation occurs. If not set, aggregation occurs at 150.