Changes to Note

We will list major changes between Geo versions that developers might need to be aware of, in this part of the user manual.

Unless otherwise noted, everything here is referencing Addon's capabilities, but we will also mention changes to class names, function names, etc.

To 4.0

Here is all the highlights of what you might need to know about 4.0.

Changes

- Class names changed:
  - Addon = geoAddon <
  - Site = geoSite (the Site Class) <
  - AdminPageAutoload = geoAdmin <
  - product_configuration = geoPC <
  - Session = geoSession <

<

- Moving methods out of geoSite and into separate classes based on what the method does. For example, geoSite→number_deformat() is now geoNumber::deformat(). The end goal is to get rid of the site class all together, but that won't be for a while. See the table below for methods that have been moved already. <

geoSite *old* method	**Moved to**
`$site_class->get_category_name($db, $cat_id);`	`geoCategory::getName($cat_id)`
`$site_class->get_classified_data($db, $listing_id)`	`$site_class->get_classified_data($listing_id);` (first param, **$db** is removed)
`$site_class->deformat_number($number);`	`geoNumber::deformat($number);`
`$site_class->sendMail(...);`	`geoEmail::sendMail(...);`

- TODO: Document what methods are moved where. Any new method that is moved out of site class document it here. <

New in 4.0

Here are the highlights, if I've forgotten anything feel free to add it or let us know. For more information on anything here, please reference the new Example addon specific for Geo 4.0

- Ability to create **Payment Gateways**: In your addon, create a new directory *payment_gateways*. In this directory you can have as many different payment gateway files as you want, one for each different payment gateway. For a guide, see

classes/payment_gateways/_template.php (or _cc_template.php). This is sort of a "fill in the blank template" file, read the documentation provided in the PHP source for how to use. Any files starting with an underscore "_" will be ignored by the system. <

- Ability to create **Order Items**: In your addon, create a new directory named *order_items*. In this directory you can have as many different order item files as you want, one for each different order item. An order item is something in an order, for example "classified" is an order item for classified listings. You can also have "sub-items" which work the same way as normal ones, except that they are attached to a main order item (one example is the bolding order item). For a guide, see *classes/order_items/_template.php*. This is sort of a "fill in the blank template" file, read the documentation provided in the PHP source for how to use. Any files starting with an underscore "_" will be ignored by the system. <
- TONS of new *software hooks* aka *core events*, which are listed below. You can see all of them documented in the example addon for Geo version 4.0. <

New Core Events 4.0

Here is a list of new core events below. See the documentation in the new Example addon for documentation. This is NOT a complete list of core events, this is just the new ones between 3.1.9 and 4.0.

new core event	**Location Used**
overload_geoFilter_replaceDisallowedHtml	**classes/php5_classes/Filter.class.php** geoFilter::replaceDisallowedHtml()
filter_geoFilter_replaceDisallowedHtml	**classes/php5_classes/Filter.class.php** geoFilter::replaceDisallowedHtml()
overload_geoFilter_listingDescription	**classes/php5_classes/Filter.class.php** geoFilter::listingDescription()
filter_geoFilter_listingDescription	**classes/php5_classes/Filter.class.php** geoFilter::listingDescription()
overload_geoFilter_listingShortenDescription	**classes/php5_classes/Filter.class.php** geoFilter::listingShortenDescription()
filter_geoFilter_listingShortenDescription	**classes/php5_classes/Filter.class.php** geoFilter::listingShortenDescription()
overload_imagesOrderItem_processImages	**classes/order_items/images.php** imagesOrderItem::processImages()
filter_ssl_url_checks	**app_top.main.php**
overload_Search_classifieds_BuildResults	**classes/search_class.php** Search_classifieds::BuildResults()
notify_Display_ad_display_classified_after_vars_set	**classes/Browse_display_ad.php** Display_ad::display_classified()
notify_display_page	**classes/site_class.php** geoSite::display_page()
admin_update_insert_group	**admin/admin_group_management_class.php**
admin_home_templates_display	In admin
Browse_ads_display_browse_result_addRow	**classes/browse_ads.php** Browse_ads::browse_result()
Browse_ads_display_browse_result_addHeader	**classes/browse_ads.php** Browse_ads::browse_result()
Search_classifieds_search_form	**classes/search_class.php** Search_classifieds::search_form
Search_classifieds_Search_filterWhereClause	**classes/search_class.php** somewhere
Search_classifieds_BuildResults_addHeader	**classes/search_class.php** Find Waldo

new core event	Location Used
Search_classifieds_BuildResults_addRow	**classes/search_class.php** somewhere
my_account_home_add_box	**classes/user_management_home.php** I hope someone reads this,
User_management_information_display_user_data_plan_information	**classes/user_management_information.php** tracking down the new core
my_account_links_add_link	**modules/my_account_links.php** events were a lot of work

To 4.0.0RC11

Below are changes to new functionality between RC10 and RC11:

In RC10 and before	In RC11 and after
In order items, the required method **geoCart_cartDisplay()**	Changed to **getDisplayDetails($inCart)**. The method is now used in various places to display "itemized" information about orders and order items, therefore the name has changed since it is no longer exclusively used in the geoCart class. If $inCart is true, that means the cart object is available to use if needed and that the information is being displayed in the normal cart view, if it is false you must make sure the method does not attempt to use the geoCart class as that class may not be instantiated.

To 4.0.4

There are a few slight changes to how some of the existing core events work. All of these changes are for core events that have been added in 4.0 and just changed between 4.0.3 and 4.0.4.

As Geo development was updating the docs on the example addon, they saw some places that could use some improvement. Below are those improvements are fully documented so you can make the necessary changes, if you were already using any of them in your own custom addon.

Browse_ads_display_browse_result_addRow 4.0.4

Changed how the core event **Browse_ads_display_browse_result_addRow** works. If you use this core event in your addon, please note these changes:

Before (4.0.0 to 4.0.3): Addon returns additional text, *including* the surrounding <td></td> HTML tags.

After (4.0.4 and up): Addon only returns an array of text, each array entry text **without** the surrounding <td></td> HTML tags, one array entry for each column to be inserted. If the addon returns an boolean false, the columns for the addon will not be displayed. These changes allow more control for anyone who wishes to customize the category browsing results template.

Changes to System Template: See Template Changes

Browse_ads_display_browse_result_addHeader 4.0.4

Changed how the core event **Browse_ads_display_browse_result_addHeader** works. If you use this core event in your addon, please note these changes:

Before (4.0.0 to 4.0.3): Addon returns additional text, *including* the surrounding <td></td> HTML tags.

After (4.0.4 and up): Addon returns an array of associative arrays, like this:

```
return array (
    array ('cssClass' => 'css_class_to_use', 'text' => 'Heading Text')
);
```

Note that the 'cssClass' in the array is not required, if not specified then "photo_column_header" will be used (the default column class name for columns without their own class). If you return boolean false, it will not display that column.

The text does **not** contain the surrounding <td></td> HTML tags.

Changes to System Template: See Template Changes

Search_classifieds_BuildResults_addHeader 4.0.4

Changed how the core event **Search_classifieds_BuildResults_addHeader** works. If you use this core event in your addon, please note these changes:

Before (4.0.0 to 4.0.3): Addon returns additional text, *including* the surrounding <td></td> HTML tags.

After (4.0.4 and up): Addon returns an array of associative arrays, like this:

```
return array (
    array ('cssClass' => 'css_class_to_use', 'text' => 'Heading Text')
);
```

Note that the 'cssClass' in the array is not required, if not specified then "search_page_results_title_row" will be used (the default column class name for columns without their own class). If you return boolean false, it will not display that column.

The text does **not** contain the surrounding <td></td> HTML tags.

Changes to System Template: See Template Changes

Search_classifieds_BuildResults_addRow 4.0.4

Changed how the core event **Search_classifieds_BuildResults_addRow** works. If you use this core event in your addon, please note these changes:

Before (4.0.0 to 4.0.3): Addon returns additional text, *including* the surrounding <td></td> HTML tags.

After (4.0.4 and up): Addon only returns an array of text, each array entry text **without** the surrounding <td></td> HTML tags, one array entry for each column to be inserted. If the addon returns an boolean false, the columns for the addon will not be displayed. These changes allow more control for anyone who wishes to customize the category browsing results template.

Changes to System Template: See Template Changes

User_management_information_display_user_data_plan_information 4.0.4

Changed how the core event **User_management_information_display_user_data_plan_information** works. If you use this core event in your addon, please note these changes:

Before (4.0.0 to 4.0.3): Addon returns additional text, *including* the surrounding <tr><td></td></tr> HTML tags.

After (4.0.4 and up): Will be inserted "outside" of a table, so no longer need to "start" from the <tr> tag. So you can use divs, or stick to the table format to ensure it matches up exactly with rest of page.

Changes to System Template: See Template Changes

To 4.0.6

Added new functionality for Addons, to have a method in the addon's info class called enableCheck(), where if that method returns false, the addon will be temporarily disabled for that page load. Added for commercial addon developers, to give a common place to do license checks if desired. Geo addons now implement this as well.

To 4.0.7

Added new method:

```
geoAddon::disableAddon($addonName, $temporary = false)
```

This method allows an addon to be disabled for the rest of the page load, or even in the DB[1)], see the docs on the new method for more details.

Changed how custom error handling works and how addons are initialized

In 4.0.7 and above if an addon uses the **errorhandle core event**, and causes a trigger_error (or "normal" error such as PHP notice) to be triggered as a result of calling the core event method, **the error will display on the page**. This is actually built into how PHP works, if a trigger_error() is called while already inside of a custom error handle call, to prevent trigger_error recursion it will echo the

error (regardless of the display_errors setting) and will log the message if PHP logging turned on (regardless of the error reporting level).

We used to suppress such messages using *ob_* functions so that they are not displayed on the page. In 4.0.7 (and to a lesser degree, 4.0.8) we have changed how the custom error handler works[2], and part of that change is to no longer suppress those messages. In other words, **the errorhandle core event must not result in a PHP error/notice or trigger_error to be called** or it will be displayed on the page.

Another side effect of this that is not as obvious, we've seen custom addons with extra white-space after the *?>* in the info.php file cause cookies to not be saved, since the extra white-space is no longer suppressed during addon initialization.

To 4.0.9

Changed default behavior of the method *geoOrderItem::geoOrder_processStatusChange_emailItemInfo()*

In the **geoOrderItem** super class that all order items extend, changed the behavior of the default method **geoOrder_processStatusChange_emailItemInfo()** as follows:

The title is automatically generated, by calling $this→getDisplayDetails(false); and using the 'title' index from the returned array, unless a title is passed in to take it's place. Here's the new (condensed) code:

```
class geoOrderItem {
//.....

    /**
     * Use this to display info about each main item, in the e-mail sent
saying the
     * order has been approved.  To keep consistent, use this format:
     *
     * ITEM TITLE [STATUS] - $COST
     *
     * (cost including sub-items of this)
     *
     * @param string $overrideTitle Can be used by individual order item to
let
     *  super class do most of the work, but allow order item to specify the
title,
     *  in the order item would return
parent::geoOrder_processStatusChange_emailItemInfo('my title')
     * @return string
     */
    public function geoOrder_processStatusChange_emailItemInfo
($overrideTitle = '')
    {
        $details = $this->getDisplayDetails(false);
```

```
        //... condensed for brevity ...//
        $title = ($overrideTitle)? $overrideTitle : $details['title'];
        return "$title $status - ".geoString::displayPrice($cost);
    }
//....
}
```

New Core Events
New core events were added in part, at the request of a 3rd party developer. Official Example Addon documentation is in the Example addon released for Geo version 4.1.0.

new core event	Location Used
registration_add_field_display	**classes/register_class.php** Register::registration_form_1()
registration_add_field_update	**classes/register_class.php** Register::insert_user()
registration_add_variable	**classes/register_class.php** Register::save_variables()

To 4.1

Added a new core event(s)

new core event	Location Used
auth_admin_user_login	geoPC::verify_credentials() (See Example Addon 2.1.0)
registration_check_info	Register::check_info() (See Example Addon 2.1.0)

Added new methods for geoOrderItem 4.1

```
geoOrderItem::addParentTypeFor($childType, $parentType);
```

This new **static** method allows addons with a parent order item, to "adopt" any "child" order items they wish. This is particularly useful if an addon creates an alternate to the classified order item (for example), the addon's order item can "adopt" all the child items for the classified item type, such as the bolding or featured listings. So long as the child order items are coded to handle this type of thing, which most normal order items are. This is best called from geoOrderItem_loadTypes_adoptions (see below).

```
geoOrderItem::getParentTypesFor($itemType);
```

Use this new **static** method as the way to get parents for a given order item. It is not good practice to call the getParentTypes() method directly for the given order item as was done in 4.0.*. Although doing it that way will work and get the initial "default" parent types, none of the parent types added using the new **geoOrderItem::addParentTypeFor()** method will be returned.

```
geoOrderItem::unregisterItemType($itemType);
```

Use this new **static** method to unregister a specific order item type so that it is not used for the rest of the page load. Using this method to remove a specific order item type would have the same effect

as removing the file. This is best called from geoOrderItem_loadTypes_obituary (see below).. It is a good way to have an addon that creates a custom order item that is meant to replace another order item, the custom order item would unregister the order item it is replacing.

Added calls at end of geoOrderItem::loadTypes()

In the geoOrderItem class, after the item types have been loaded, it now makes 2 calls: geoOrderItem_loadTypes_adoptions and geoOrderItem_loadTypes_obituary:

geoOrderItem_loadTypes_adoptions: This call is to give items a common place to "adopt" child items, using *geoOrderItem::addParentTypeFor()*.

geoOrderItem_loadTypes_obituary: This call is to give items a common place to unregister other order item types, using *geoOrderItem::unregisterItemType()*.

The call to these 2 methods first calls adoptions, then obituary. I know it's a little backwards from RL, but doing it in this order allows an addon order item to first get, then adopt, all the children types of a particular item, then in obituary it can kill off that item, effectively the addon order item can totally replace a "built in" order item.

New methods in geoCart 4.1

The following methods are available to use in the geoCart object, see docs in **classes/php5_classes/Cart.class.php** for further documentation.

```
geoCart::isRecurringCart();
```

Will return true if the Cart is a recurring billing cart, or false otherwise. Note that the cart must already be to a certain point before this will return true, the order item needs to already be set up in it, and it must already have figured out it is a standalone cart, and geoCart::isRecurringPossible() must be true. If not Enterprise edition, automatically returns false.

```
geoCart::isRecurringPossible();
```

Will return true if recurring billing appears to be possible, false otherwise. Meant primarily for use in recurring order items, when they are deciding if they should behave like a recurring order item or not. This will return true if there is at least one payment gateway that is capable of processing recurring billings. If not Enterprise edition, automatically returns false.

```
geoCart::isInMiddleOfSomething();
```

Returns true or false: Whether or not the current cart is "in the middle of something", in other words, if you were to attempt to add something new to the cart in the current page load, would it be interrupting something? Be sure the Cart is init before calling this, even if only items are inited. Note that if it is a "standalone" cart, it will always be in the middle of something until cart checkout is finished.

New methods in geoOrder 4.1

```
geoOrder::getRecurringBilling();
```

Gets the recurring billing object "attached" to the order, or false if there is none attached. If not Enterprise edition, it automatically returns false.

```
geoOrder::setRecurringBilling($recurring);
```

Sets the recurring billing object for the order. It both lets the order know the recurring billing is attached, and calls the method in the recurring object to set the order. so that $recurring→setOrder() is not necessary. If not Enterprise edition, it returns false and does not "attach" the recurring billing object.

```
geoOrder::getItem('recurring');
```

Not a new method, just a *new trick* for an existing method. Pass in the string 'recurring' into getItem() and, if there is a recurring item in the order, and it appears that the order is set up to be a recurring billing order, it will return that recurring item, or false if none found. If not Enterprise edition, it returns false automatically.

Added Recurring Billing Capabilities 4.1

One of the new features in Enterprise editions of 4.1, is the ability to use recurring billing for subscriptions, and potentially other order items. See below for information on what new stuff is available for use specifically for the recurring billing feature.

geoRecurringBilling - New class, only available in Enterprise editions (files needed are not included in other software editions). See the file **classes/php5_classes/RecurringBilling.class.php** for full documentation. Note that if not Enterprise edition, the system will not call the recurring billing specific methods, so there is no need to check for enterprise edition before using the recurring billing class, in any of the recurring billing specific methods.

New methods in Payment Gateways

There are a number of new methods available to use in payment gateways (listed below). See the **classes/payment_gateways/_template.php** for documentation on each of these new methods:

- isRecurring() <
- getRecurringAgreement() <
- adminRecurringDisplay() <
- recurringUpdateStatus() <
- recurringCancel() <

New Methods in Order Items:

Any "parent" order item can now be billed as a recurring billing order item. Note that the order item must be able to handle being billed as 1 time fee, for when there are no enabled recurring billing gateways (or not in Enterprise edition). See below for a list of the new methods available for order

items, see **classes/order_items/_template.php** for documentation on each of these methods:

- isRecurring() <
- getRecurringInterval() <
- getRecurringPrice() <
- getRecurringDescription() <
- getRecurringStartDate() <
- recurringBilling_updateStatus($recurring) <
- recurringBilling_cancel($recurring) <

To 4.1.2

Added new static method, **geoFilter::badword($string)** that filters a given string by the badwords set in the admin panel. [3)]

To 5.0

Note the following changes from 4.1.* to 5.0.0.

JS/CSS File Locations Changed 5.0

All CSS and JS have been moved to respective sub-folders in **geo_templates/default/external/** , to allow them to be easily over-written by custom template sets. This **does not include** any JS files that are part of 3rd party libraries, such as **prototype.js**, such files are still located in the original locations.

Also, the file previously named **general.js** has been moved with the rest, and re-named to **main.js**.

New Fields to Use Usable by Addons 5.0

There is a new class, **geoFields** which now handles all the "fields to use" settings. Addons have the ability to "add to" the fields and/or display locations so that those fields and/or display locations are settable in the admin panel, and even the ability to use at the category and "viewing user group" level seamlessly. For these added abilities, see the updated Example addon, specifically the core events **geoFields_getDefaultLocations** and **geoFields_getDefaultFields**.

geoUtil::array2csv() moved 5.0

The method **geoUtil::array2csv()** has been moved to **geoArrayTools::toCSV()**.

Defunct methods removed from geoUtil 5.0

The methods **csv2array()** and **array_filter_assoc()** have been removed from the **geoUtil** class, as they are not used in the software and have not been used for a while.

New Methods in geoListing 5.0

The following methods are available to use in the geoListing class, found in the file **classes/php5_classes/Listing.class.php**.

```
geoListing::addDataSet($dataSet);
```

New **static** method, which allows you to pass in an array of results pulled from the database, from the classifieds table. This is useful to pre-cache listing data to later be used when using geoListing object for a listing, so that when *geoListing::getListing()* is used the data will already be available, and it doesn't have to query the database a second time for data that has already been retrieved.

```
geoListing::remove($listingId);
```

New **static** method, used to remove a listing from the database. Note that this removes all traces including images, bids, etc.

Also added new addon hook, **notify_geoListing_remove** called from inside the geoListing::remove() method.

New Methods in geoTemplate 5.0

The following method has been added to the geoTemplate class in 5.0.0.

```
geoTemplate::template404($tplFile, $longMsg);
```

Calling this static will display an error message stating that the given template file could not be found in any of the template sets currently loaded. It will also stop the rest of the page from loading. It is meant more for internal use by the geoTemplate class (when template files are not found), but is a public method available for use if needed.

```
geoTemplate::getFilePath($g_type, $g_resource, $template_filename,
$dieOnError);
```

This method has been altered, to add the new param **$dieOnError**. The parameter is true by default, and when true, if the template file being requested cannot be found, it makes a call to the new **geoTemplate::template404()** to display an error message. If false, the method will just return the "relative" path if it cannot be found, which is the previous behavior in versions before 5.0.0.

New Methods in geoAddon 5.0

The following method was added to allow addons to easily set the default template used for an addon's page. [4)]

```
geoAddon::setDefaultPageTemplate($addon, $page, $tplName, $tplContents);
```

This is a non-static method, it can be used by an addon's install or upgrade method(s) in the addon's setup class, but we recommend to instead use the var **$pages_info** set in the Info class for the addon, which you can see full documentation on in the Example addon for 5.0. For full documentation on this method, see the PHP docs that go with the geoAddon class in 5.0.

geoString::displayPrice() changes 5.0

Will be changing the default behavior of the method **geoString::displayPrice()**: If $pre and/or $post parameters are specified, the values will be run through **geoString::fromDB()** automatically. We found that when the pre and post are used in the displayPrice method, they are usually coming from the database and need to be un-formatted, the times that they do not need it are the exception rather than the norm.

Core events added and removed 5.0

There are a few new core events, along with a few that have been removed. First the ones that have been removed:

Core Event Removed	Reason
admin_home_templates_display	Admin page removed (no more DB-based templates)

There have been some new core events added as well:

New core event	Location
admin_display_page_attachments_edit_end	**admin/design.php** DesignManage::display_page_attachments_edit()

Changed for File-based templates 5.0

In 5.0, the "DB-based templates" have been totally removed, replaced by file-based templates. This means a lot of things have changed, related to displaying the page, displaying modules, etc. We won't list all the changes, but below we do list changes that affect core functionality.

Changes

- **geoCachePage::quotePage()** - method changed, it now only accepts a single input. It no longer "looks" for tags that look like (!TAG!) as those tags are not used any more. <

Removed Classes

- **geoCacheFont** – Class totally removed, as there is no more internal font management system in 5.0. <
- **geoCacheTemplate** – Class totally removed, as with the use of Smarty templates, there is no need to cache templates any more. <

Removed Methods

- **DataAccess::get_page_modules()** – Method removed, as it is no longer needed with no DB-based templates. <
- **DataAccess::preload_pages()** – Was geared specifically toward way things worked in DB-based templates. <
- **DataAccess::getBaseUrl()** – While we were snooping around for old DB-template geared methods, we ran across this method, which is not used anywhere in the code, and is not needed since there is *geoTemplate::getBaseUrl()*. <
- **geoAddon::replaceTag()** – Method removed, as it was the "DB-Template geared" duplicate of *geoAddon::runTag()* method, which is still used. <

Removed Tables – The tables listed below are removed from fresh installations, and references to them are removed from the software, including in the *geoTables* class.

- **geodesic_addon_pages** – No more DB-saved addon page template attachments. Template attachments stored in files now. <
- **geodesic_templates** <
- **geodesic_templates_history** <
- **geodesic_pages_fonts** – No more font-management, all CSS is located in CSS files. <
- **geodesic_pages_modules** – Module to page attachments do not apply in file-based templates, as modules are now attached to the templates themselves, and the attachments are saved in files, not in the DB. <
- **geodesic_pages_templates** <
- **geodesic_pages_templates_affiliates** <
- **geodesic_extra_pages_registry** – Extra page contents are now file sub-templates, set in attachments file for each extra page. <

To 5.0.1

Changes to geoString::removeAccents() 5.0.1

Changed how the removeAccents() method works, to use an associative array as a map for converting characters. Also made the method convert Cyrillic characters to the Latin equivalent.

New methods in geoAddon class 5.0.1

The following methods have been added to the geoAddon class in order to allow for a minor feature addition in 5.0.1.

```
$geoAddon->updateTemplates($addonName);
```

New method that does the automated addon template assignment in the default template set, for the given addon name. This is used automatically when installing a new addon or updating an existing addon, or when the re-scan template attachments tool is performed on the default template set. See the PHP docs in the example addon for documentation on this new method.

```
$geoAddon->getEnabledList();
```

New method that gets an array of enabled addons, data included is addon name, title, and the addon's info class. Used by the new functionality when re-scanning template attachments for the default template set. See the PHP docs in the example addon for documentation on this new method.

Changes to Register class 5.0.1

In all methods in the Register class, the $db var has been removed from parameters passed into methods.

Removed the following defunct methods:

- **Register::get_group_from_registration_id()** <
- **Register::insert_auction_price_plan()** <

To 5.0.2

The following has changed in 5.0.2.

Changes to geoString::isFilePath() 5.0.2

Added new static method, **geoString::isFilePath($filePath)** which checks to make sure there are no invalid characters in the given path.

Changes to geoFile::scandir() 5.0.2

Changed scandir() method, adding $sorting_order var as 5th optional variable, to allow changing it to sort backwards.

To 5.0.3

The class **geoCacheTemplate** was removed as it is no longer used, now that the use of DB-based templates has been removed. This class should have been removed in 5.0.0, same as the geoCacheFont class, we just missed it is all.[5)]

To 5.1

insert_category_questions() moved 5.1

The method **insert_category_questions()** that used to be located in the **tempSiteClass** class, has been moved to classes/order_items/_listing_placement_common.php as the static method **_listing_placement_commonOrderItem::addCatQuestions($listingId, $session_variables)**.

The method now returns "search text" to be inserted based on questions for listing, or false upon failure.

new geoBrowse::categoryBrowsing() method 5.1

Added new method, **geoBrowse::categoryBrowsing($text, $cacheNamePrefix)** that gets rid of some duplicated code across different browsing pages.

To 5.1.2

Added new hook, **admin_home_display_news** to allow addons to also display news feeds, below the main one on the admin home page. See new example addon for documentation.

To 5.2.0

We have started working on adding the ability for the admin user to create an order from the admin area. The feature was not able to be completed before 5.2.0 was released, so we commented out the new admin page, but all the functionality is still there. We recommend any addon developers that have order items in their addons, to "turn on" the beta "Add to cart" functionality, and make sure it works with your custom order item.

To try out (and test any custom order items): In the file **admin/php5_classes/Admin.class.php** find the lines:

```
            //menu_page::addPage('admin_cart','orders','Create New
Order','menu_trans.gif','cart.php','AdminCart');
                //menu_page::addPage('admin_cart_select_user', 'admin_cart',
'Select User', 'menu_trans.gif', 'cart.php', 'AdminCart');
```

Un-comment both of those lines, and upload the changed file. Then in your admin panel, go to the new page **Orders > Create New Order** to start the process of creating a new order for a specific user.

To also "try out" editing a listing from the admin panel[6]: In the file **admin/templates/listing_details/index.tpl** find the lines:

```
        {*  Add new order not completed...
        <div class='center'>
            <a class='mini_button' href='/support/geocore-
wiki/doku.php/id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
```

```
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-
wiki;doku.php;id,page,admin_cart;userId,{$userId};action,new;main_type,listing_edit;listing_id,{$listingId};;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;/'>Edit
Listing Details</a>
        </div>
        *}
```

And remove the Smarty comments, by **replacing with**:

```
        <div class='center'>
            <a class='mini_button' href='/support/geocore-
wiki/doku.php/id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-wiki;doku.php;id,support;geocore-
wiki;doku.php;id,support;geocore-
wiki;doku.php;id,page,admin_cart;userId,{$userId};action,new;main_type,listing_edit;listing_id,{$listingId};;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;/'>Edit
Listing Details</a>
        </div>
```

Typical changes needed in 3rd party developer created custom order items:

- Any URL referenced files, such as CSS or JS, need to have "../" added to the front when in the admin panel. If in a template file, something like this might work:

```
<script type="text/javascript" src="{if $in_admin}../{/if}{external
file='js/addons/my_addon/custom.js'}"></script>
```

<

- To get the "user ID", do NOT use **$session→getUserId()**. Instead use **$cart→user_info['id']**. Since the user ID in the session var will be the admin user, but that is not the user that the order is being created for. <

Known Issues [7)]

- Preview item doesn't work. <
- Various features specific to when an admin user is creating the order, were not able to be completed yet. <
- As any further issues are found, they will be added to this list. If you find a problem with this feature, using "stock" order items, and not on this list, please let us know via support ticket. <

To 6.0.0

Order item new required method getCostDetails 6.0.0

In the geoOrderItem abstract class, there is a new abstract method getCostDetails() that must be implemented by every order item starting in 6.0, or it will generate a fatal error. Most order items just need to have the getCostDetails() method copied and pasted from the _template.php file from 6.0.

This method is used to give payment gateways more information about exactly what is being purchased in the order, so if something goes against a particular gateway's policies, that payment gateway can remove itself from the payment selections. Going forward when we add more things that can be purchased through the cart, this method will be more important.

Note that we do not make drastic changes like this lightly, we do realize that this will affect all 3rd party order items. We are making this change to be consistent with the getDisplayDetails() that the getCostDetails() was modeled after, and more importantly so that the method would not get overlooked by developers going forward, since it will be very important to ensure payment gateways are able to adhere to their usage agreements.

order item method getDisplayDetails 6.0.0

In order items, in the method **getDisplayDetails()**, that method can now have one additional array index **canAdminEditPrice** which should be a bool value. If true, it will allow that item's price to be edited when the item is in a cart in the admin panel. See the **_template.php** template order item released with version 6.0 for example of the new array key in use.

geoPaymentGateway::_successFailurePage() 6.0.0

In **geoPaymentGateway::_successFailurePage()**, added new optional 4th param, for invoice

object. If invoice object is passed in, an invoice link will be displayed on success/failure page.

Smarty library updated 6.0.0

We have updated the Smarty library, to Smarty 3.1.5 which may require some template changes. In the PHP code:

- No more setting template vars using the syntax

  ```
  $tpl->tpl_var = 'value';
  ```

  Use the normal syntax instead:

  ```
  $tpl->assign('tpl_var','value');
  ```

  <

- A lot of methods used for custom Smarty plugins have been moved out of the geoTemplate class, and moved into their own plugin files. If you used any of those methods directly, you will need to update your code. <
- Calling $tpl→fetch() will no longer default to set tpl name to 'index', update your code to use $tpl→fetch('index') instead. <
- Scriptaculous is now always loaded, no need to use **$view→scriptaculous=1;** <
- Implemented using the [template_set]filename.tpl syntax on include tags, so can force using template from specific template set. <
- Addon (and module) tags are now loaded **on the fly**, and the tag functions are now passed $params and $smarty, the same vars that would be passed if using a custom smarty plugin function.
  - Because of this, addon tags can no longer add content to the {header_html}. To still make this possible, we have added new functionality to the tags class: if there is a method named the same as the tag, but appended with **_auto_add_head**, that method will be called at the top, so that method would be able to add content to be displayed by {header_html}. This only affects addon tags, since addon pages the page method is not called "on the fly". <
  - Long story short, you need to place the addon tag in one of the **main_page** templates used to render the current page for the **_auto_add_head** method to get called[8)]. For times when you may want to use an addon tag inside system/module/addon sub-template: Also add the addon tag to one of the **main_page** templates, with added parameter **headOnly=1** in the tag. If that parameter is present, it will only parse _auto_add_head() method, the addon tag's contents will not be displayed. <
  - Module to template attachments have no effect anymore, no need to re-scan attachments, and modules/addon tags can be attached to any template (not just main_page templates). See above note about using addon tags that need to add something to the header though. <

<

DataAccess->addBrowsingWhereClause() Deprecated 6.0.0

The method **addBrowsingWhereClause()** has been deprecated, and will be removed in future release. More importantly, if the optional 4th parameter, **$additionalTable** is passed in, it will **print an error and exit**. This is because the "new" way of doing the same thing, requires additional info in order to add a table to the query that gets listings.

DataAccess->isBrowsingWhereClause() Deprecated 6.0.0

The method **isBrowsingWhereClause()** has been deprecated, and will always **return false**. This method will be removed in a future release.

Instead of the old:

```
if ($db->isBrowsingWhereClause()) {
//...
}
```

Use this instead (using new methods in 6.0.0 described below):

```
if ($db->getTableSelect(DataAccess::SELECT_BROWSE)->hasWhere()) {
//...
}
```

DataAccess->getBrowsingWhereClause() Removed 6.0.0

The method **getBrowsingWhereClause()** has been **removed**. Unlike the addBrowsingWhereClause() counterpart, there was not a way to make this "old way" work with the "new way", so we were not able to simply deprecate the function[9)].

The replacement for this, is getting a **geoTableSelect** object, by calling **DataAccess→getTableSelect(DataAccess::SELECT_BROWSE, true)**, and using that geoTableSelect to generate the SQL needed.

DataAccess->getTableSelect() method added 6.0.0

To replace **DataAccess→addBrowsingWhereClause()** is a new method, **DataAccess→getTableSelect()** which returns the **geoTableSelect object** that is used to generate the SQL used to retrieve listings. See the 6.0.0 PHP docs for the new **geoTableSelect** class for more information.

Here is a quick & dirty overview:

Adding a filter:

To add a filter you would do something like this:

```
$db->getTableSelect(DataAccess::SELECT_BROWSE)
    ->where("table_name.table_column = 'table_value'")
    //can add multiple where statements, that will be added together using
AND later...
    //Note:  This next line illustrates making use of new quoteInto()
method, see PHP docs for 6.0 for more info
    ->where($db->quoteInto("table_name.table_column = ?", 'table_value'));
```

If you are displaying a bunch of listings, and want to take any browsing filters into effect, you could do something like the below:

```
//pass in true for second param, to work with a "copy", so that
//any ->where() calls we make do not actually add browsing filters for the
rest of the page...
$query = $db->getTableSelect(DataAccess::SELECT_BROWSE, true);
//only get live listings
$query->where("geodesic_classifieds.live = 1",'live')
    //only get classifieds
    ->where("geodesic_classifieds.item_type = 1");

//Note that it is important to always specify the table name along with
//any column name.  There can sometimes be multiple tables being queried,
//so it is important to specify the table to avoid problems when the two
//tables have matching named columns.
//In other words, use geodesic_classifieds.live, NOT just live by itself.

//When you make the $query object treated as a string, it is converted into
SQL.
$result = $db->Execute(''.$query);

//get the count of how many results there are
$count = $db->GetOne(''.$query->getCountQuery());
//VERY IMPORTANT:  Once finished with query, unset it to free up memory!
Otherwise, will have a
//bunch of geoTableSelect object copies floating around when they are no
longer needed, taking up memory unnecessarily...
unset($query);
```

Search_classifieds::like_clause() removed 6.0.0

We have removed the function **$Search_classifieds→like_clause()** as it is no longer being used by the base software, as it is no longer needed with the new geoTableSelect class.

Removed Core Event Search_classifieds_Search_filterWhereClause 6.0.0

The core event Search_classifieds_Search_filterWhereClause has been removed, as the search class now uses the geoTableSelect to generate the query, it is not possible to filter the where clause

directly. Must now do so by using new core event **Search_classifieds_generate_query**.

Added core event Search_classifieds_generate_query 6.0.0

To replace the removed core event for manipulating search results, we have added a new core event, **Search_classifieds_generate_query**. See the docs in the example addon for more info.

user.register remote API success return value changed

Changed the success registration to return the following:

```
array ('success'=>1, 'user_id'=>123);
```

Also added 2 new options, to also include the {body_html} contents and/or the full page contents of the normal registration successful page. See the sample API client for the call for details.

Rename of geoListing::getBids()

The badly named method geoListing::getBids() has been renamed to geoListing::**bidCount**() to make it clear what about the bids it is getting without having to look at docs on the method. In 6.0.0 the old method *getBids* will remain, but is **deprecated**, and will be removed in a future release.

geoSite::get_number_of_bids() is deprecated

The function *geoSite::get_number_of_bids()* is deprecated, and will be removed in future release. Use the static method **geoListing::bidCount()** instead.

SellerBuyer call generatePaymentLink renamed

The SellerBuyer call **generatePaymentLink** did not make much sense for where it happens, so it has been re-named to **listingClosed**.

geoSite::get_category_configuration removed 6.0.0

The get_category_configuration() method has been removed from geoSite.

In its place, use the static method geoCategory::getCategoryConfig()

API Transports

Changed the remote API so that different "transport methods"[10)] can be used other than the built in XMLRPC. This should be "transparent" to API calls, but if there are any custom API calls that make use

of any IXR_* classes or functions (such as for returning an error), those will need to be changed to use only the built-in methods in the geoAPI class in version 6.0.0. See the documentation on the geoAPI class for 6.0.0.

Also, if you have a need to use something other than XMLRPC, check out the main built-in transport at **classes/api/_transports/xmlrpc.php**. For addons, can add transports by putting them in a folder within the addon directory named **api/_transports/transport_name.php**.

To 6.0.4

New method listing_renew_upgrade::checkNoDowngrade() (affects listing extras) 6.0.4

See bug 346. Changes have been made to stop the user from "down-grading" an ad if the ad is still live, to prevent an extra from being removed from a listing prematurely. Any 3rd party addons that add listing extras, in the order item's class, in the methods **getDisplayDetails** and **geoCart_other_detailsDisplay** should add few lines of code to enforce the new "no downgrade" ability for custom extras. See the bolding order item to use as an example of how to use the new method.

geoFilter::listingDescription() changes 6.0.4

There is a new parameter on said method, **$forceHtml** which when true, will force to always strip HTML from the description. If set to false (default), and the admin has configured the setting to not filter HTML when browsing, it will not automatically remove all HTML tags from the description. This is changed behavior, before it would always remove all HTML from the description, that behavior can no longer be assumed.

To 6.0.6

geoOrderItem::getRecurringPrice() 6.0.6

For "parent" recurring order items, in order for the new recurring tax to apply correctly to the recurring cost, changes are needed. Need to change it to get entire order total, not just total for items less than 100. Instead of:

```
    public function getRecurringPrice()
    {
        //use getOrderTotal to get recurring amount, so that special
recurring children can alter
        //recurring price as long as item process order is less than 100
        return $this->getOrder()->getOrderTotal(100);
    }
```

Use something like this[11]:

```
    public function getRecurringPrice()
    {
        //use getOrderTotal to get recurring amount, so that special
recurring children can alter
        //recurring price
        return $this->getOrder()->getOrderTotal();
    }
```

To 7.0.0

Version 7 includes significant changes to the Geographic Regions. In particular, the State and Country fields associated with users and listings are now referenced entirely differently.

Accessor functions are provided to make getting values the new way easier:

```
//as of 7.0.0
geoRegion::getStateNameForListing($listing_id)
geoRegion::getCountryNameForListing($listing_id)
//as of 7.0.2
geoRegion::getStateNameForUser($user_id)
geoRegion::getCountryNameForUser($user_id)
```

To 7.1.0

New list/gallery/grid views 7.1

Note that the following core events may need slight adjustments, in order to use a "field label" on browsing views, for new "list" and "gallery" views, as per bug 441.

Both core events **Browse_ads_display_browse_result_addHeader** and **Browse_module_display_browse_result_addHeader**:

Each array entry that had 'text' and 'css' before, now has an additional entry, 'label' which should be set to the label text to use for list and gallery views when browsing. See the updated example addon.

Support for $info->php5_mode officially dropped 7.1

(see bug 618) We've been keeping around the ability to use info variable $php5_mode to allow addons to have PHP5 version of the various files in a "php5_classes" sub-folder, for backwards compatibility with addons built for Geo version 3.1 and earlier. That variable was deprecated in version 4.0 when the software first started required PHP 5.2, and is now being completely removed. Now the software will load the main files from the base folder, ignoring if the info class happens to have the old php5_mode set or not.

Use of lightUpBox replaced 7.1

(See bug 145) Technically this goes in "design" since it deals with **javascript** as is part of the template sets, but it does flow over into "development" some so we are mentioning here as well.

As part of bug 145, we have removed the old JS object lightUpBox that used to allow opening things in a lightbox. It has been replaced by a jQuery plugin **gjLightbox**.

Note that use of CSS class **lightUpLink** and **lightUpImg** will still work perfectly fine, as the jQuery version of the lightbox will do the same thing as the old Prototype did. The change is that if you have anything that calls upon the lightUpBox object directly in your custom javascript, that will need to be re-coded. Below are a few examples of how to use it now:

Description	Old call	New call
Use AJAX to display URL contents in lightbox	`lightUpBox.lightUpLinkManual('http://example.com');`	`jQuery(document).gjLightbox('get','http://example.com');`
Display contents in a lightbox	`lightUpBox.openBox('<div>Contents to show</div>');`	`jQuery(document).gjLightbox('open','<div>Contents to show</div>');`

geoPC::get_hashed_password() Changes 7.1.0

The password hashing done in the software is now able to be extended by addons, to add additional "hash algorithms". This includes the ability to use a salt if desired/needed. The primary reason for this change, is to allow addons to validate passwords their own way, to allow importing users from other software platforms that save the password in a different way. Since it allows for a salt, "built in", it will work even with password generating that makes use of random generated salt. See the example addon and bug 713 for more details on what is possible.

For existing addons however, the main thing is that the function **$geoPC→get_hashed_password()** has changed, it now accepts 4 possible parameters[12]:

- String **$username** <
- String **$password** The password, in plain text. <
- string **$hash_type** *CHANGED* : This is either 'core:sha1', 'core:plain', or 'core:auctions_old'[13], OR can be a type added by an addon. See the example addon for more info. <
- string **$salt** *NEW* (optional) The salt, if the hash type uses a salt value. If matching against password already saved, pass in the salt value as well (saved in the geodesic_logins table). Or if generating a new password, leave blank and it will generate a salt on it's own. If the hash does not use a salt, this can be left out or a blank string. <
- **Returns** *CHANGED*: string|array|bool: If the hash type does not need to use salt, returns string of the hashed password. If it does use salt, returns array in form **array('password' ⇒ 'Hashed Password Value', 'salt' ⇒ 'Salt value')**. Or can return false on error (for instance, if you pass in blank username or password). <

The main changes are that the 3rd parameter is a string (as described) instead of a number, you can now optionally pass in a salt value as the 4th parameter, and the return value "could be" an array. Below is a sample of what might need to be changed in existing code.

```
//Set up the vars
$pc = geoPC::getInstance();
```

```
$db = DataAccess::getInstance();
$username='some_user';
$password='some pass in plaintext';

//OLD WAY

//hash it using setting for what hash to use for clients...
$hashed_password = $pc->get_hashed_password($username, $password,
$db->get_site_setting('client_hash_pass');
//NO salt in the "old way"

//...
//Do something with the hashed password here...
//...

//NEW WAY

$salt='';//If we have a salt value already, set it here
//Note that salt is optional, and not used for the built in hash as it uses
the username for the salt

$hashed_password = $pc->get_hashed_password($username, $password,
$db->get_site_setting('client_hash_pass'), $salt);
if (is_array($hashed_password)) {
    $salt = $hashed_password['salt'];
    $hashed_password = $hashed_password['password'];
}
//...
//Do something with the hashed password / salt here...
//...
```

In addition, there have been changes to the **geodesic_logins** table:

- Addition of **hash_type** column, which is normal to be blank when updating. It is set when the password is changed, so that when it is validating a password, it knows to ONLY check the "hash type" set in this column. If the column is blank for a specific entry, when checking the password, it will go through and check all of the password hash types. <
- Addition of **salt** column. This column will be blank unless the site is using an addon that implements it's own hash type, that makes use of a salt. The example addon has fully working md5 hash algorithm for demonstration purposes. <

New geoRecurringBilling::processPayment() method and order item hook 7.1.0

In order to "process" all "payment signals" made for recurring billing payments, we created a new method in **geoRecurringBilling** class called **processPayment()**. In any custom payment gateways that make use of recurring_process to process a payment signal, it is advised that at the point it is adding the transaction to the recurring billing object, instead call $recurring→processPayment($transaction) to let the system do a few things for you. See that method to see exactly what it does for you, and make sure your own payment gateway does not do those

things in duplicate, and for further documentation on it's use.

To go along with the new method, is a new order item hook **RecurringBilling_processPayment** which is called from the above mentioned method. The purpose is to mimic the use of Order_processStatusChange, but instead of using for when order's status is changing, instead notify when a new payment is received. Or in other words, it lets an order item get triggered when a new payment is received for something, this is ideal for use by order items meant to track payments for **3rd party affiliate software**. Check out the updated Post Affiliate Pro addon's use of the new hook for an example of use.

New addon text sections 7.1.0

See bug 687, and updated example addon. The edit text page for addons has changed, and can now be divided into "sections", ideal for addons with a ton of text entries.

Started using jQuery 7.1.0

As per bug 535, the software now uses jQuery "built in". There are a ton of places that still use prototype, we will be converting those places over to use jquery instead, but that process will span several versions. Until it is complete, prototype and Scriptaculous will continue to be loaded "along side" jQuery, make sure if you use jQuery that you do not use the $(...) shortcut as it will be interpreted as Prototype. Instead use the full jQuery(...) for jquery.

7.1.1

geoImage::displayImageBlockLargeLink() and displayImageBlockLarge() removed 7.1.1

The methods displayImageBlockLargeLink() and displayImageBlockLarge() in the geoImage class have been removed, as they are no longer used, and would not function on new installations anyways due to missing templates.

To 7.2.0

geoOrderItem::geoCart_initSteps() - REQUIRED param change 7.2.0

For bug 156, changes are required to any 3rd party order items, to the static method **geoCart_initSteps**. First, need to add **$allPossible=false** to the method so that it looks like:

```
public static function geoCart_initSteps ($allPossible=false)
```

<tip c w>**Warning:** Failure to make this change will cause a **Fatal Error** if running versions above 7.2. If nothing else, just make the change to add **$allPossible=false** to the method

geoCart_initSteps() to make sure it does not cause a fatal error. Note that this can safely be added to order items that work on previous versions as well, so having $allPossible=false will work on previous versions, but NOT having it in 7.2 will cause fatal error.</tip>

Now, in the actual function, if it is empty you don't need to do anything else. If you do have steps added however, and this is for the auction or listing process: Make sure that if you do any checks based on user info, including user's price plan settings or user group settings, or anything else like that... If $allPossible is true, **always** add any steps that have a chance of showing. This does not apply to steps that are only added based on site-wide settings, for instance if a step is only added when auctions is turned on. But any checks that are not on "site wide" settings, in other words for one person the step could be added, another the step not added, make sure if $allPossible is true that such steps are added anyways.

Also be aware that if $allPossible is true, that means it is in the admin, and is not part of the normal cart process. So if you use this method for other things besides simply adding steps[14], you will need to bypass any custom code like that if $allPossible is true.

In other words, if $allPossible is true, the method should initiate any possible steps, even ones that may only be added in certain circumstances, and should be made to not do anything else outside of adding steps.

Within the same method, if you call "children" order items to let them initialize steps, make sure to pass in the value of $allPossible instead of null for the second parameter to geoOrderItem::callUpdate. In other words the line should look something like this:

```
geoOrderItem::callUpdate('geoCart_initSteps',$allPossible,$children);
```

geoOrderItem::geoCart_canCombineSteps() - new method 7.2.0

For bug 156, added a new method called by the system for parent order items, **geoCart_canCombineSteps()**. If you have a custom order item, that has multiple steps, and wish to make use of the new options in 7.2 for combining steps into single page, can define this **static** method in your custom order item, and have it return true. When you do so, it will show in the new page in the admin at **Listing Setup > Listing Placement Steps** where the admin can choose to combine steps.

To 7.3

Change constant GEO_TEMPLATE_URL 7.3

Replacing the old config constant **GEO_TEMPLATE_URL** with the new **GEO_TEMPLATE_LOCAL_DIR** - which now works more like the ADMIN_LOCAL_DIR setting in config.php. See bug 899 for details on this and other changes to how the design system works.

Use head_html instead of header_html 7.3

In a move to be a lot more consistent(see bug 939), especially now that <header> is a valid tag in HTML 5 and is totally different from <head>... Places that used to use header_html for inserting things in the <head> section, have been changed to use head_html instead. The most notable are:

- **{header_html}** to **{head_html}** - Note that the old {header_html} will still work for backwards compatibility with older versions. <
- **header_html** in Payment Gateway **admin_display_payment_gateways()** - In payment gateways, the function admin_display_payment_gateways() has one possible entry in the array that is returned, for "header_html", that has been changed to "head_html". If you need a payment gateway to work with older and newer versions, set both head_html AND header_html in the return array (set to the same thing). <
- **geoView→setHeaderVar()** renamed to **geoView→setHeadVar()** - the old function is deprecated. <
- **geoView→setHeaderTpl()** renamed to **geoView→setHeadTpl()** - the old function is deprecated. <
- **geoView→getAssignedHeaderVars()** renamed to **geoView→getAssignedHeadVars()** - the old function is deprecated. <

New {footer_html} and {add_footer_html} tags 7.3

As part of bug 899, we've added a new tag {footer_html} which will display "delayed JS" on the page. It is only used if the admin has turned on the setting in **design > settings** to move certain JS to the {footer_html}. Now calls to **geoView→addJScript()**, and **geoView→addTop()** - the former has new parameters to "force" the JS to only load in the head. It defaults such that it will move the JS file to be loaded in the footer_html automatically (if set to use {footer_html}).

We've also added {add_footer_html} ... {/add_footer_html} smarty block function which serves to move the contents to the {footer_html} if the admin has turned on using the {footer_html} tag.

Things to be aware of:

- Any JS needs to be done so that it works whether it is loaded in the top or the bottom. If you are using jQuery() to delay the JS until the page is loaded, you are already doing this. <
- If you have embedded JS inside templates (which is fine in certain circumstances), that uses anything "built in" - use the new block tags **{add_footer_html}** and **{/add_footer_html}** before and after the JS. See {add_footer_html}..{/add_footer_html} for more details of usage. <
- Calls to **geoView→addTop()** will automatically be loaded in footer_html if it "looks like" Javascript being added. If the stuff you are adding to the top "needs" to be loaded in the top, use the new parameter to force it to be in the top not the footer. <

Changed User_management_information_display_user_data 7.3

Added ability to return array of arrays, to allow each order item OR addon that uses the hook, to return multiple rows to be displayed on the user information page.

To 7.4

Category saved differently for listings 7.4

Inserting / Updating Listings

This is likely to break any addons that insert or update listings, that set the category. Instead of setting the category column in geodesic_classifieds table, in 7.4 instead call **geoCategory::setListingCategory($listing_id, $category_id)** and that method will insert the category for the given listing ID. Or see that method's source code for how to insert it in the database yourself[15] Note that without the changes, it may "seem" to work at first glance on software updates, but new installations will not have the category column so will cause an error when inserting / updating. Also, even on updates, just setting the category column will have no effect if not also inserted the "correct" way for 7.4 and up, that listing will be treated just as if the category is not set at all.

If it only sets the session_variables and lets the "built in" order item(s) insert the listing, no changes are needed as you would still just set the $session_variables['category'] to the ID and the built-in order item will insert the categories correctly.

Browsing / Filtering by Category

If browsing / filtering by category, no longer use the "in_statement". If your custom addon uses "in_statement" it will need to be updated. The recommended method is to use the **geoTableSelect** for "building" the query, and using the new method **geoBrowse→whereCategory()** to make the adjustments to filter by specified category. If doing it "by hand", the idea is to check if the listing is in the category using a simple query:

```
SELECT * FROM `geodesic_listing_categories` WHERE `category`=#
```

Where # is the category ID. We would recommend to use that as a "sub query" to your main query, similar to this:

```
SELECT * FROM `geodesic_classifieds` WHERE `live`=1 ... AND EXISTS(SELECT *
FROM `geodesic_listing_categories` WHERE
`geodesic_listing_categories`.`listing`=`geodesic_classifieds`.`id` AND
`geodesic_listing_categories`.`category`=#)
```

Where # is the category ID... The query is simplified for brevity, obviously having ... is not a valid query.

Category Name / Description / Image in Language Table 7.4

In past versions, the category name and description were duplicated, set in the main geodesic_categories table and also in the languages table (now called geodesic_categories_languages as noted in next change). Now both are only set in **geodesic_categories_languages** so as to avoid

confusion, and to be consistent with other areas that have language tables.

Also note that the image has moved to the geodesic_categories_languages table as well.

geodesic_classifieds_categories_languages renamed to geodesic_categories_languages 7.4

As the title says, the table **geodesic_classifieds_categories_languages** is now named **geodesic_categories_languages**. Note that if you used the built-in **geoTables::categories_languages_table** or **$db→geoTables→categories_languages_table** those have been updated to reflect the new name, so you may not need to make any changes in your own code depending on how you reference the table.

geoCategory::getInStatement() Removed 7.4

As part of bug 1082, the method geoCategory::getInStatement() has been removed. Simply put, there is no more in_statement for categories.

geoCategory::updateInStatement() Removed 7.4

As part of bug 1082, the method geoCategory::updateInStatement() has been removed. Simply put, there is no more in_statement for categories.

geoSite::get_sql_in_statement() Remove 7.4

As part of bug 1082, the method geoSite::get_sql_in_statement() has been removed. Simply put, there is no more in_statement for categories.

New listing "field type" of "yesno" for listings 7.4

As part of bug 1128, there are 2 new fields for classifieds (and auctions), "show_contact_seller" and "show_other_ads", both of which are set to either "yes" (default value) or "no". And the part that is important here, there is a new "type of field" for those listings, "yesno" which indicates to the validation that the value is either going to be "yes" or "no". Any other value would be evaluated as "no".

JS function geoUtil.autoSubmitForm() moved 7.4

In the ongoing task to convert existing functionality to use jQuery instead of Prototype, the old function geoUtil.autoSubmitForm() has been converted to jQuery and moved to gjUtil.autoSubmitForm() as part of bug 1163

1)

by passing in true for $temporary
2)

Changes were needed to make things work correctly in the new PHP 5.2.10. Although that was not the original reason for the changes, since PHP 5.2.10 was released *after* Geo 4.0.7, but that has become the new reason we made this change as it is a better reason. 
3)

Old location at $geoSite→check_for_badwords($text)
4)

Note that in 5.0, there is only file-based templates, and the "default" template set can always assume to be the default that is shipped with the software. If the default template set is altered, that is not supported.
5)

We mention it's removal in 5.0 notes but it still exists until 5.0.3, it is just not used anywhere in the system in 5.0 onward.
6)

Note that the above changes must be done first, or the mods below will not work.
7)

These known issues, along with additional features we wanted to add as part of this feature, are the reason this feature is commented out in 5.2.0
8)

The pre-parser that allows _auto_add_head methods to be called, is not capable of following into system/module/addon sub-templates, there is not enough info at time pre-parsing takes place to do this.
9)

like we normally try to do, to give developers a little time to update their code
10)

What we are calling the layer that processes api input and sends the output back to the client.
11)

The important part being, do not pass in a value for getOrderTotal, with it passing in 100 the new recurring tax added in 6.0.6 is bypassed.
12)

Note that we do not normally document code in the wiki, but currently the PHP Docs that we used in the past is not working, so we are not able to provide the updated PHPDocs like we have provided for past versions.
13)

This is really old and not too common, used for compatibility with version 1 auctions platform
14)

For instance if you use it as a hook to add something new to the cart
15)

Note that using the method to insert is recommended as it will probably be more "update friendly".