EXT: Improved overlays

Extension Key: overlays
Language: en
Keywords: forDevelopers, forBeginners

Table of Contents

Introduction

What does it do?

This extension makes it easier to get a set of records from the TYPO3 database, properly overlaid for sites with multiple languages. Furthermore it provides in a central place a set of tools related to database calls that are sometimes spread over several classes in the TYPO3 architecture. Versions and workspaces are also supported, as well as language overlays in foreign tables.

It also aims to reduce the number of database calls, at least for language overlays. In a scenario where many records are fetched and overlaid the number of requests to the database will be much smaller, albeit at the cost of increased PHP processing (there's no free lunch). For version overlays, extension "overlays" still relies on the TYPO3 Core method `t3lib_page::versionOL()`.

There are limitations to what can be achieved with a simple library. Please read the "Limitations" chapter carefully.

Questions and support

If you have any questions about this extension, please ask them in the TYPO3 English mailing list, so that others can benefit from the answers. Please use the bug tracker on forge.typo3.org to report problem or suggest features (http://forge.typo3.org/projects/extension-overlays/issues).

Credits

Most of the overlay code itself comes from the TYPO3 core and was not developed by myself. It is encapsulated in various methods for easier use.

Keeping the developer happy

Every encouragement keeps the developer ticking, so don't hesitate to send thanks or share your enthusiasm about the extension.

If you appreciate this work and want to show some support, please check http://www.monpetitcoin.com/en/francois/support-me/.

Installation

All you need is to install the extension and you can start using it in your own developments right away. The class uses the autoloading mechanism, so – as of TYPO3 4.3 – you don't even need to manually include it in your code.

Compatibility

As of version 2.1.0, TYPO3 4.5 or more is required.

Upgrading to version 2.0

Version 2.0 introduced support for versioned records (a.k.a workspaces). **The existing API was unchanged**, so no code relying on "overlays" should break. There are some new methods and the signature of some others changed, but all new parameters have default values, so nothing is broken by switching to this new version.

A single method was deprecated: `getWorkspaceCondition()` was renamed to `getVersioningCondition()`. `getWorkspaceCondition()` still exists, but you will get deprecation warnings if you use it.

Parameters or methods existing as of version 2.0 are marked as such in the API description below.

Limitations

TYPO3's overlay architecture makes it very difficult to provide a full-fledged API for some operations without building a complete SQL parser on top (which is essentially what the "dataquery" extension of the Tesseract project does).

Table joins

In `t3lib_db::exec_SELECTquery()` it is perfectly okay to make joins by setting several table names in the `$from_table` parameter. Example:

```
$result = t3lib_db::exec_SELECTquery('*', 'pages, tt_content', 'tt_content.pid = pages.uid');
```

This will not work correctly for overlays. Indeed tables must be overlaid individually. So in order to have joined tables overlaid, it would be necessary to separate the result of the query into two tables again, overlay each part and join them again.

The suggestion that can be made here is to select records from both tables separately using `tx_overlays::getAllRecordsForTable()` and perform the join in your PHP code.

Text search

Imagine trying to select records based on a LIKE condition place on a text field. Imagine the following records:

uid	l18n_parent	sys_language_uid	title	...	
1	0	0	Weather forecast for Geneva	...	
2	1	3	Wetterbericht für Genf	...	

Now let's say we want to make a text-based search on the title. We might write something like:

```
$records = tx_overlays::getAllRecordsForTable('uid, title', 'tt_news', "title LIKE '%" . $searchWord . "%'");
```

($searchWords would have been properly escaped first, of course). Since extension "overlays" aims to follow the TYPO3-way of doing overlays it will automatically select only records in the default language and overlay them afterwards. This means that the above query will work fine if you search for "geneva", but will not work if you search for "genf", because that word does not appear in the default language. The result of the query is thus wrong: there is a record that would match the condition, but it won't be found because it's not in the default language.

On the other hand if you tried to select directly in the right language (using `t3lib_db::exec_SELECTquery()`), you will miss some information because some fields in "tt_news" are not editable in the translations.

There's no simple solution to this conundrum. One might be to select records in the requested language first, use these records' "parent_id" information to get the records in the default language, and then overlay the latter with the former. This is obviously a lot of overhead. Not to mention what a nightmare it becomes if you add the dimension of version overlays.

A better solution is probably to avoid such text-based searches at all and use search engines in that case (Apache Solr, Google, etc.), although that will probably not work for workspace preview, except if you can manage to index workspace versions. Then again it's probably unnecessary to be able to correctly search in workspaces in most cases.

All that is described above is not an issue if you test numeric fields, such as simple flags, dates, etc.

Developer's manual

Introduction

Traditionally getting a set of records from the database in a TYPO3 application (typically a FE plugin) is done by calling a method like `t3lib_db::exec_SELECTquery()`. Then – if your site uses multiple languages and your records are localized using standard TYPO3 mechanisms – you have to overlay each record with its translation by calling `t3lib_page::getRecordOverlay()`. It is also up to you to check whether the translation succeeded or not. Here is how you code may look like, taking "tt_news" as an example and doing a simple rendering with a bullet list:

```
      // Restrict records to default language
$languageCondition = $GLOBALS['TCA']['tt_news']['ctrl']['languageField'] . ' IN (0, -1)';
      // Also take records that exist only in chosen language
if ($GLOBALS['TSFE']->sys_language_content > 0) {
      $languageCondition .= ' OR (tt_news.' . $GLOBALS['TCA']['tt_news']['ctrl']['languageField'] . " =
'" . $GLOBALS['TSFE']->sys_language_content . "' AND tt_news." . $GLOBALS['TCA']['tt_news']['ctrl']
['transOrigPointerField'] . " = '0')";
}
      // Add enable fields check
$where = '(' . $languageCondition . ') ' . $GLOBALS['TSFE']->sys_page->enableFields('tt_news');
$news = $GLOBALS['TYPO3_DB']->exec_SELECTgetRows('uid, pid, title', 'tt_news', $where);

$content = '<ul>';
foreach ($news as $item) {
            // Perform overlay on each record
      $overlay = $GLOBALS['TSFE']->sys_page->getRecordOverlay('tt_news', $item, $GLOBALS['TSFE']-
>sys_language_content, $GLOBALS['TSFE']->sys_language_contentOL);
            // Check if record was not unset (in translation strict mode)
      if (isset($overlay)) {
            $content .= '<li>' . $overlay['title'] . '</li>';
      }
}
$content .= '</ul>';
```

Getting overlaid records in a single call

Using the "overlays" library the above code is reduced to the following:

```
$news = tx_overlays::getAllRecordsForTable('title', 'tt_news');
$content .= '<ul>';
foreach ($news as $item) {
      $content .= '<li>' . $item['title'] . '</li>';
}
$content .= '</ul>';
```

The `tx_overlays::getAllRecordsForTable()` method takes the same arguments as `t3lib_db::exec_SELECTquery()`, but takes care of all the enable fields and all the translation process.

On top of this the "overlays" library is so structured that all translation overlays are gotten in a single database request. The normal TYPO3 process generates one query per record to overlay. Thus "overlays" plays much nicer to your database. This comes at the price of some additional PHP processing.

The full API

On top of the method demonstrated above, class `tx_overlays` provides a full toolbox related to getting overlaid records in the TYPO3 frontend. This API is described below.

The extension comes with an autoloader declaration, so there's no need to include the `tx_overlays` class if you're using TYPO3 4.3 or above.

`tx_overlays` is a fully static class. As such it cannot be XCLASSed.

Method:	Parameters:	Description:
getAllRecordsForTable	**$selectFields** (string) Comma-separated list of fields to select from the table. **$fromTable** (string) Table from which to select. **$whereClause** (string) Additional WHERE clause for the query. Enable fields are handled automatically. Optional, defaults to empty string. **$groupBy** (string) GROUP BY field(s). Optional, defaults to empty string. **$orderBy** (string) ORDER BY field(s). Optional, defaults to empty string. **$limit** (string) LIMIT value ([begin,]max). Optional, defaults to empty string. **$indexField** (string) Optional, name of a field to use as an index for the result array. *Must be included in the list of select fields.* (since v2.2.0)	This method returns an array with all records properly overlaid with their translations and filtered according to the table's enable fields. NOTE: this will not work properly using implicit joins (i.e. `SELECT * FROM foo,bar...`) as neither enable fields, nor translations, nor versions will be cleanly applied to both tables.
getSingleRecordForTable	**$selectFields** (string) Comma-separated list of fields to select from the table. **$fromTable** (string) Table from which to select. **$whereClause** (string) Additional WHERE clause for the query. Enable fields are handled automatically. Optional, defaults to empty string. **$groupBy** (string) GROUP BY field(s). Optional, defaults to empty string. **$orderBy** (string) ORDER BY field(s). Optional, defaults to empty string.	Similar to `getAllRecordsForTable()`, but returns a single record by automatically applying a limit of 1 to the underlying SQL query.
getEnabledFieldsCondition	**$table** (string) Name of the table to build the condition for. **$showHidden** (boolean) Set to TRUE to force the display of hidden records. **$ignoreArray** (array) Defines which enable fields to ignore. Use keys like "disabled", "starttime", "endtime", "fe_group" (i.e. keys from "enablecolumns" in TCA) and set values to TRUE to exclude corresponding conditions from WHERE clause.	This method returns the complete SQL condition needed to respect all enable fields as defined in the TCA of the table. Essentially this method is a wrapper around t3lib_page::enableFields(). The only difference is that it removes the " AND " at the beginning of the condition, so that it's not necessary to prepare for such condition by adding things like "1=1" to your SQL statements. For more details look at what happens inside t3lib_page::enableFields().
getLanguageCondition	**$table** (string) Name of the table to build the condition for. **$alias** (string) *(v2.0)* Alias of the table to be used in the SQL condition.	This method returns the SQL condition necessary to select the correct records with records to their language, based on the table's TCA and the currently selected language in the FE.
getWorkspaceCondition	**$table** (string) Name of the table to build the condition for.	**Deprecated**. Use getVersioningCondition() instead.
getVersioningCondition *(v2.0)*	**$table** (string) Name of the table to build the condition for. **$alias** (string) Alias of the table to be used in the SQL condition. **$getOverlaysDirectly** (boolean) Set to TRUE to directly get the overlays, instead of the placeholders to overlay (check the method's comments in the code for a deeper discussion). Leave the default value of FALSE if unsure.	This method assembles the SQL condition necessary to select the correct records for the live web site or for the current workspace.
getAllFieldsForTable *(v2.0)*	**$table** (string) Name of the table to get the fields for.	This method returns the list of all fields for a given table (based on querying the database, not analyzing the TCA).
selectBaseFields *(v2.0)*	**$table** (string) Table from which to select. **$selectFields** (string) List of fields to select from the table.	This method makes sure that the query will include a number of base fields necessary for the overlay process. This means the uid and pid fields.
selectOverlaysFields	**$table** (string) Table from which to select. **$selectFields** (string) List of fields to select from the table.	This method makes sure that the query will include all necessary fields for the **language** overlay process. It returns the original list of selected fields plus any other necessary fields that were missing.

Method:	Parameters:	Description:
selectOverlayFieldsArray *(v2.0)*	**$table** (string) Table from which to select. **$selectFields** (string) List of fields to select from the table.	This method returns an array containing the list of all fields that should be added to the current list of selected fields, in order to have all the necessary information for the **language** overlay process.
selectVersioningFields *(v2.0)*	**$table** (string) Table from which to select. **$selectFields** (string) List of fields to select from the table.	This method makes sure that the query will include all necessary fields for the **version** overlay process. It returns the original list of selected fields plus any other necessary fields that were missing.
selectVersioningFieldsArray *(v2.0)*	**$table** (string) Table from which to select. **$selectFields** (string) List of fields to select from the table.	This method returns an array containing the list of all fields that should be added to the current list of selected fields, in order to have all the necessary information for the **version** overlay process.
overlayRecordSet	**$table** (string) Table name **$recordset** (array) Full recordset to overlay. Must containt uid, pid and $TCA[$table]['ctrl'] ['languageField']. **$currentLanguage** (integer) Uid of the currently selected language in the FE. **$overlayMode** (string) Overlay mode. If "hideNonTranslated" then records without translation will not be returned un-translated but removed instead. **$doVersioning** (boolean) *(v2.0)* Set to TRUE if version overlays must be performed.	This method performs both the translation and versioning overlays on the given recordset, according to the relevant TCA settings. NOTE: if overlay mode is "hideNonTranslated", the overlaid recordset may contain less records than the original one, if some records were missing a translation.
getOverlayRecords	**$table** (string) Name of the table for which to fetch the records. **$uids** (array) Array of all uid's of the original records for which to fetch the translation. **$currentLanguage** (integer) uid of the system language to translate to. **$doVersioning** (boolean) *(v2.0)* Set to TRUE if version overlays must be performed.	This method is a wrapper around tx_overlays::getLocalOverlayRecords() and tx_overlays::getForeignOverlayRecords(). It will call the appropriate method depending on the translation structure (i.e. translations in same table or in foreign table). It returns the same result as these two methods.
getLocalOverlayRecords	**$table** (string) Name of the table for which to fetch the records. **$uids** (array) Array of all uid's of the original records for which to fetch the translation. **$currentLanguage** (integer) uid of the system language to translate to. **$doVersioning** (boolean) *(v2.0)* Set to TRUE if version overlays must be performed.	This method gets all translation overlay records for the chosen language and for all records indicated by the list of uid's. It will work for tables that contain their own translations. If workspace preview is active, the translation overlays will have been properly version-overlaid. It returns all overlay records organized by uid and pid so that the overlay process can take place properly afterward.
getForeignOverlayRecords	**$table** (string) Name of the table for which to fetch the records. **$uids** (array) Array of all uid's of the original records for which to fetch the translation. **$currentLanguage** (integer) uid of the system language to translate to. **$doVersioning** (boolean) *(v2.0)* Set to TRUE if version overlays must be performed.	This method gets all translation overlay records for the chosen language and for all records indicated by the list of uid's. It will work for tables that use a foreign table to store their translations. If workspace preview is active, the translation overlays will have been properly version-overlaid. It returns all overlay records organized by uid and pid so that the overlay process can take place properly afterward.
overlaySingleRecord	**$table** (string) Name of the table for which the operation is taking place **$record** (array) Record to overlay **$overlay** (array) Overlay of the record	This method actually performs the translation overlay of a single record taking into account all fine-grained translation settings at field level (i.e. "l10n_mode"). It returns the properly overlaid record.

Known problems

None to date.

To-Do list

Nothing planned for now.

For other feature requests, please use the bug tracker on forge.typo3.org (http://forge.typo3.org/projects/extension-overlays/issues).