# EXT: Static Info Tables

Extension Key: **static_info_tables**

The content of this document is related to TYPO3
- a GNU/GPL CMS/Framework available from www.typo3.com

## Table of Contents

# Introduction

## What does it do?

Static Info Tables is a collection of database tables which provides data on:

- territories
- countries
- country zones (states, local government areas)
- languages
- currencies
- taxes
- markets

These tables are static. They should be used as reference only, and are useful for for applications like shop plugins and other extensions that handle country-related data.

The records can be found in the backend in the root level of the page tree (globe).

## Requirements

Versions of extension Static Info Tables (static_info_tables) > 2.0.0 require TYPO3 4.0.
If you want to use the multiple language variant extensions like static_info_tables_de or others, then you should install the extensions lib and div for version 2.0.1. They are needed for an error free data processing to fetch the translated text fields.

## Standards

Standards relevant to the Static Info Tables are:

- for country codes: ISO 3166-1 (see http://www.iso.org/iso/en/prods-services/iso3166ma/index.html );
- for country subdivisions: ISO 3166-2;
- for currencies: ISO 4217;
- for languages: ISO 639-1 and RFC 3066 (see http://www.faqs.org/rfcs/rfc3066.html );
- for postal address formats: UPU S42-1 (see http://www.upu.int/post_code/en/postal_addressing_systems_member_countries.shtml ).

A major source of reference data is provided by the Locale Data Repository (CLDR) Project http://www.unicode.org/cldr/.

## Credits

Thanks to René Fritz, Eckhard Zemp, Stanislas Rolland, David Brühlmeier, Franz Holzinger and Martin Kutschker for their contributions.

... and thanks to the translators.

## Support

You can get support for this extension at http://jambage.com/index.php?id=162 .

# Users manual

This manual provides information for developers.

## Tables Structure

The main concept of the tables are the ISO codes.

It is highly recommended not to use  the uid of the records as reference! They may change with the next update. Use the ISO codes instead.

You can choose between following code types:

- alpha 2 codes: GB, DE
- alpha 3 codes: GBR, DEU
- number codes: 826, 276

Most of the time, there are references to other tables with all three code types.

All fields have a prefix like cn_ for the table static_countries. This is useful if you join tables, so you have unique field names for xx_iso_2 for example.

### static_territories

This table provide some data of territories of the world. The data is hierarchical organized which means the field tr_parent_iso_nr defines the parent of the current entry.

The static_countries table have a reference to this table.

```
CREATE TABLE static_territories (
  uid int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  pid int(11) unsigned DEFAULT '0' NOT NULL,
  tr_iso_nr int(11) unsigned DEFAULT '0' NOT NULL,
  tr_parent_iso_nr int(11) unsigned DEFAULT '0' NOT NULL,
  tr_name_en varchar(50) DEFAULT '' NOT NULL,
  PRIMARY KEY (uid),
  UNIQUE uid (uid)
);
```

### static_countries

This table provide some data of all countries.

```
CREATE TABLE static_countries (
  uid int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  pid int(11) unsigned DEFAULT '0' NOT NULL,
  cn_iso_2 char(2) DEFAULT '' NOT NULL,
  cn_iso_3 char(3) DEFAULT '' NOT NULL,
  cn_iso_nr int(11) unsigned DEFAULT '0' NOT NULL,
  cn_parent_tr_iso_nr int(11) unsigned DEFAULT '0' NOT NULL,
  cn_official_name_local varchar(128) DEFAULT '' NOT NULL,
  cn_official_name_en varchar(128) DEFAULT '' NOT NULL,
  cn_capital varchar(45) DEFAULT '' NOT NULL,
  cn_tldomain char(2) DEFAULT '' NOT NULL,
  cn_currency_iso_3 char(3) DEFAULT '' NOT NULL,
  cn_currency_iso_nr int(10) unsigned DEFAULT '0' NOT NULL,
  cn_phone int(10) unsigned DEFAULT '0' NOT NULL,
  cn_eu_member tinyint(3) unsigned DEFAULT '0' NOT NULL,
  cn_uno_member tinyint(3) unsigned DEFAULT '0' NOT NULL,
  cn_address_format tinyint(3) unsigned DEFAULT '0' NOT NULL,
  cn_zone_flag tinyint(4) DEFAULT '0' NOT NULL,
  cn_short_local varchar(70) DEFAULT '' NOT NULL,
  cn_short_en varchar(50) DEFAULT '' NOT NULL,
  PRIMARY KEY (uid),
  UNIQUE uid (uid)
);
```

Local names are in the official languages of the countries. Appropriate Unicode fonts are required to display all entries on a client computer.

Address formats are described in the section Address Formats of the configuration

### static_country_zones

This table provides data about country subdivisions, provinces, departments or states.

In the static_countries table, there is a flag `cn_zone_flag` which indicates if country zones exist for a given country.

```
CREATE TABLE static_country_zones (
  uid int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  pid int(11) unsigned DEFAULT '0' NOT NULL,
  zn_country_iso_2 char(2) DEFAULT '' NOT NULL,
  zn_country_iso_3 char(3) DEFAULT '' NOT NULL,
  zn_country_iso_nr int(11) unsigned DEFAULT '0' NOT NULL,
  zn_code varchar(45) DEFAULT '' NOT NULL,
  zn_name_local varchar(128) DEFAULT '' NOT NULL,
  zn_name_en varchar(50) DEFAULT '' NOT NULL,
  PRIMARY KEY (uid),
  UNIQUE uid (uid)
);
```

### static_currencies

```
CREATE TABLE static_currencies (
  uid int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  pid int(11) unsigned DEFAULT '0' NOT NULL,
  cu_iso_3 char(3) DEFAULT '' NOT NULL,
  cu_iso_nr int(11) unsigned DEFAULT '0' NOT NULL,
  cu_name_en varchar(50) DEFAULT '' NOT NULL,
  cu_symbol_left varchar(12) DEFAULT '' NOT NULL,
  cu_symbol_right varchar(12) DEFAULT '' NOT NULL,
  cu_thousands_point char(1) DEFAULT '' NOT NULL,
  cu_decimal_point char(1) DEFAULT '' NOT NULL,
  cu_decimal_digits tinyint(3) unsigned DEFAULT '0' NOT NULL,
  cu_sub_name_en varchar(20) DEFAULT '' NOT NULL,
  cu_sub_divisor int(11) DEFAULT '1' NOT NULL,
  cu_sub_symbol_left varchar(12) DEFAULT '' NOT NULL,
  cu_sub_symbol_right varchar(12) DEFAULT '' NOT NULL,
  PRIMARY KEY (uid),
  UNIQUE uid (uid),
  KEY parent (pid)
);
```

### static_languages

Local name of a language is in the language itself. Appropriate Unicode fonts are required to display all entries on a client computer.

```
CREATE TABLE static_languages (
  uid int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  pid int(11) unsigned DEFAULT '0' NOT NULL,
  lg_iso_2 char(2) DEFAULT '' NOT NULL,
  lg_name_local varchar(99) DEFAULT '' NOT NULL,
  lg_name_en varchar(50) DEFAULT '' NOT NULL,
  lg_typo3 char(2) DEFAULT '' NOT NULL,
  lg_country_iso_2 char(2) DEFAULT '' NOT NULL,
  lg_collate_locale varchar(5) DEFAULT '' NOT NULL,
  lg_sacred tinyint(3) unsigned DEFAULT '0' NOT NULL,
  lg_constructed tinyint(3) unsigned DEFAULT '0' NOT NULL,
  PRIMARY KEY (uid),
  UNIQUE uid (uid),
  KEY parent (pid)
);
```

### static_taxes

```
CREATE TABLE static_taxes (
  uid int(11) unsigned DEFAULT '0' NOT NULL auto_increment,
  pid int(11) unsigned DEFAULT '0' NOT NULL,
  crdate int(11) unsigned DEFAULT '0' NOT NULL,
  deleted tinyint(4) unsigned DEFAULT '0' NOT NULL,
  hidden tinyint(4) unsigned DEFAULT '0' NOT NULL,
  starttime int(11) unsigned DEFAULT '0' NOT NULL,
  endtime int(11) unsigned DEFAULT '0' NOT NULL,
  tx_country_iso_nr int(11) unsigned DEFAULT '0' NOT NULL,
  tx_country_iso_2 char(2) DEFAULT '' NOT NULL,
  tx_country_iso_3 char(3) DEFAULT '' NOT NULL,
  tx_zn_code varchar(45) DEFAULT '' NOT NULL,
  tx_name_en varchar(255) DEFAULT '' NOT NULL,
  tx_scope tinyint(3) unsigned DEFAULT '0' NOT NULL,
  tx_code varchar(5) DEFAULT '' NOT NULL,
  tx_class tinyint(3) unsigned DEFAULT '0' NOT NULL,
  tx_rate varchar(20) DEFAULT '' NOT NULL,
  tx_priority tinyint(3) unsigned DEFAULT '0' NOT NULL,
  PRIMARY KEY (uid),
  KEY parent (pid)
);
```

### static_markets

This table provides international market places following the ISO-Standard 10383.

```
CREATE TABLE static_markets (
  uid int(11) auto_increment,
  pid int(11) default '0',
  country int(11) default '0',
  mic varchar(4) default '',
  institution_description varchar(90) default '',
  acronym varchar(30) default '',
  city varchar(30) default '',
  url varchar(90) default '',
  PRIMARY KEY (uid),
  KEY parent (pid)
);
```

# Backend Features

Class class tx_staticInfoTables_div offers a few functions mainly used in TCA.

## sys_language table

As example, the TCA definition of the `static_lang_isocode` field of the `sys_language` table is modified in order to modify the behavior of the ISO code select field in the website language record.

```
t3lib_div::loadTCA('sys_language');
$TCA['sys_language']['columns']['static_lang_isocode']['config'] = array (
      'type' => 'select',
      'items' => Array (
        Array('',0),
      ),
      // uid definition
      // 'foreign_table' => 'static_languages',
      // 'foreign_table_where' => 'AND static_languages.pid=0 ORDER BY static_languages.lg_name_en',
      'itemsProcFunc' => 'tx_staticinfotables_div->selectItemsTCA',
      'itemsProcFunc_config' => array (
        'table' => 'static_languages',
        'where' => 'lg_iso_2 IN ('EN', 'NL', 'FR', 'DE')',
        'indexField' => 'uid',
        'prependHotlist' => 1,
        //  defaults:
        //'hotlistLimit' => 8,
        //'hotlistSort' => 1,
        //'hotlistOnly' => 0,
        //'hotlistApp' => TYPO3_MODE,
      ),
      'size' => 1,
      'minitems' => 0,
      'maxitems' => 1,
      );
```

The function tries to return values in the right language. Additionally you can use an ISO code field as index instead of the uid.

## TCEmain post processing

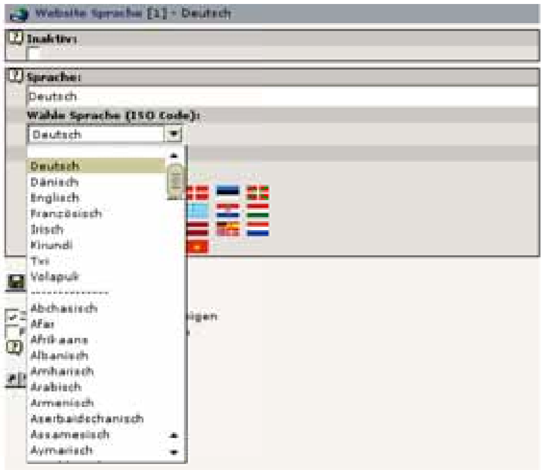
This following code registers a TCEmain post processing function.

```
$TYPO3_CONF_VARS['SC_OPTIONS']['t3lib/class.t3lib_tcemain.php']['processDatamapClass'][] =
'EXT:static_info_tables/class.tx_staticinfotables_syslanguage.php:&tx_staticinfotables_syslanguage';
```

The class looks like this:

```
class tx_staticInfoTables_syslanguage {
  function processDatamap_postProcessFieldArray($status, $table, $id, $fieldArray, $parent) {
    if($table == 'sys_language' AND $fieldArray['static_lang_isocode']) {
        // the hotlist will be updated only if the field changed, because only then it's in the
$fieldArray
      tx_staticinfotables_div::updateHotlist ('static_languages', $fieldArray['static_lang_isocode'],
'uid');
    }
  }
}
```

What does it do? When a `sys_language` record is saved and the field `static_lang_isocode` was changed, the hotlist for the table `static_languages` will be updated. The hotlist will be used by the function `tx_staticinfotables_div::selectItemsTCA` to prepend the most used entries to a selector box.



Website Sprache [1] - Deutsch

Inaktiv: □

Sprache: Deutsch

Wähle Sprache (ISO Code): Deutsch

# Frontend API

Class tx_staticinfotables_pi1 may be used in any Frontend Typo3 extension.

Some default values may be configured using the TS template constant editor. See the Configuration section.

## Class inclusion

Use the following statement to include it in your script:

```
require_once(t3lib_extMgm::extPath('static_info_tables').'pi1/class.tx_staticinfotables_pi1.php');
```

## Class initialization

The following example statements will instantiate and initialize the class:

```
$this->staticInfo = t3lib_div::makeInstance('tx_staticinfotables_pi1');
$this->staticInfo->init();
```

Init() will initialize:

- the language of display: using the TypoScript template config.language property;
- the default currency used to format amounts: using the configured currencyCode;
- defaults for selected values in drop-down lists: using  the configured countryCode, countryZoneCode, currencyCode and languageCode.

## Getting a localized name

Each of the following example statements will return the name of some entity given its code. The name will be in the language specified by the TypoScript config.language property.

```
$this->staticInfo->getStaticInfoName('COUNTRIES', $countryCode);
$this->staticInfo->getStaticInfoName('SUBDIVISIONS', $zoneCode, $countryCode);
$this->staticInfo->getStaticInfoName('CURRENCIES', $currencyCode);
$this->staticInfo->getStaticInfoName('LANGUAGES', $languageCode, '', '', $self);
```

In the case of LANGUAGES, if $self is set to 1, the name of the language will be returned in the language itself.

See the source code for a description of the parameters.

## Creating a localized drop-down selector

Each of the following example statements will return a string of <select> and <option> HTML tags to display a drop-down selector of countries, country subdivisions, currencies or languages. The drop-down selector will be produced in the language specified by the TS config.language property.

```
$this->staticInfo->buildStaticInfoSelector('COUNTRIES', $fieldName, $CSSClassName, $selectedCountryCode,
'', $submitFlag, $id, $title, $where, $lang, $local);
$this->staticInfo->buildStaticInfoSelector('SUBDIVISIONS',  $fieldName, $CSSClassName,
$selectedZoneCode, $countryCode, $submitFlag, $id, $title);
$this->staticInfo->buildStaticInfoSelector('CURRENCIES', $fieldName, $CSSClassName,
$selectedCurrencyCode, '', $submitFlag, $id, $title);
$this->staticInfo->buildStaticInfoSelector('LANGUAGES', $fieldName, $CSSClassName,
$selectedLanguageCode, '', $submitFlag, $id, $title);
```

See the source code for a description of these and other parameters.

Note that the entries of the drop-down selector are sorted using the current locale. If the front end page is using utf-8 encoding (config.metaCharset = utf-8), then the locale should be fully qualified; for example: config.locale_all = fr_CA.UTF-8.

### Formating an amount

The following statement will return the amount $amount formated for display in the default currency or in the currency specified in the previously called loadCurrencyInfo method.

```
$formatedAmount = $this->staticInfo->formatAmount($amount);
```

### Loading an alternate currency

The currency to be used to format amounts may be overridden using method loadCurrencyInfo

```
$this->staticInfo->loadCurrencyInfo($currencyCode);
```

where $currencyCode is the ISO alpha-3 code (cu_iso_3) of some currency.

The loaded currency will be effective until the same method is called again with a different currency code.

Init() loads the default currency.

### Formating an address

The following statement will return an address formated for display according to the address format specified for the country in the static_countries table (cn_address_format).

```
$formatedAddress = $this->staticInfo->formatAddress($delim, $streetAddress, $city, $zip,
$subdivisionCode, $countryCode);
```

where $delim will often be chr(10), $subdivisionCode is country subdivision code, and $countryCode is a ISO alpha-3 country code.

### Applying consumer taxes

The following statement will return an array of taxes applicable to a sale of a specified amount to a final consumer while taking into account import-export and internal market rules (EU and Canada).

```
$appliedTaxesArray = applyConsumerTaxes($amount, $taxClass, $shopCountryCode,
$shopCountrySubdivisionCode, $buyerCountryCode, $buyerCountrySubdivisionCode, $EUThreshold) ;
```

where $taxClass takes values 0 – Non-taxable, 1 - Taxable goods, or 2 – Taxable services and where setting $EUThreshold indicates that the shop has sales to the buyer's country beyong the regulatory threshold (meaningful only within EU internal market).

The returned array will be an array of 4-tuples of applied taxes: ('tx_name', 'tx_rate', 'tx_amount', 'tx_priority').

### Getting country info

You can get now the info for different countries by applying the parameters you have obtained from a FE edit field.

```
$countryArray = tx_staticinfotables_div::fetchCountries($input['country'], $input['countryISO2'],
$input['countryISO3'], $input['countryISONr']);
```

where the first parameter may be the text of the country in English or the local language and only parts of it. The fitting languages will be guessed by the input text.

# Configuration

## Installation

### Language Packs

The current version of the static tables do not include any other language labels than English. Other languages are provided by extra extensions and extend the current entries.

### Character encoding

The tables are encoded with UTF-8. This means that if you use another encoding in your installation you have to convert the encoding. The extension provide a conversion tool in the extension manager. Select "UPDATE!" in the main menu in the details view of the extension.

Each language pack has the same tool which must be used when importing the localized data.

### Managing the data

Extension Static Info Tables Manager (extension key: cc_infotablesmgm) should be used to manage the data. It is also used to generate the language packs.

## Frontend Configuration

### Static Template

When the frontend API is used, static template "Static Info Tables" should be included in the TS template being used.

### TypoScript Reference

Use the Constant Editor template tool to set these properties used by frontend API:

| Property: | Data type: | Description: | Default: |
|---|---|---|---|
| currencyCode | string | The ISO alpha-3 code of the currency to be used to format amounts. | EUR |
| countryCode | string | ISO alpha-3 code of the default selected country in drop-down selector. | |
| countryZoneCode | string | Code of the default selected country zone in drop-down selector. | |
| languageCode | string | ISO alpha-2 code of the default selected language in drop-down selector.<br>May also be of the form LG_CN where LG is an ISO alpha-2 language code and CN an ISO alpha-2 country code. | |
| onChangeAttribute | string | Value of onchange attribute of drop-down selectors, when present. | javascript:this.form.submit(); |

### Address Formats

Nine(9) address formats are defined in TS setup of static template "Static Info Tables" as follows:

```
 addressFormat {
## See www.upu.int
## Semi-colon (;)-separated address lines
## Examples of address format 1: Austria, Denmark, France, Germany, Russia
  1 = %street;%zip  %city;%countryName
## Examples of address format 2: India
  2 = %street;%city %zip;%countryName
## Examples of address format 3: Australia, USA
  3 = %street;%city %countrySubdivisionCode  %zip;%countryName
## Example of address format 4: Canada
  4 = %street;%city (%countrySubdivisionName) %zip;%countryName
## Example of address format 5: Great Britain
```

```
    5 = %street;%city;%zip%countryName
## Example of address format 6: Mexico
    6 = %street;%zip %city, %countrySubdivisionCode;%countryName
## Example of address format 7: Italy
    7 = %street;%zip %city %countrySubdivisionCode;%countryName
## Example of address format 8: Spain
    8 = %street;%zip %city (%countrySubdivisionName);%countryName
## Example of address format 9: Brazil
    9 = %street;%city - %countrySubdivisionCode;%zip;%countryName
  }
```

# Change Log

| Version: | Changes: |
|---|---|
| 1.3.0 | Provide a userFunc for TCA select fields with "hotlist" support.<br>Changed the language selector in sys_language table to use new userFunc.<br>Added the functionality to automatically set label fields to the BE users language – if possible.<br>Additional language names in french, spanish, dutch and german.<br>Additional zone data.<br>New address types.<br>Updates for french and spanish names.<br>Some updates in the currencies table.<br>Renamed static_currencies.cu_name to  static_currencies.cu_name_en<br>Changed the TCA definition to make translations possible. |
| 1.4.0 | New functions in tx_staticinfotables_div: getIsoCodeField(), getTitleFromIsoCode() |
| 1.8.0 | New table static_territories<br>Imported the main data from Locale Data Repository (CLDR) Project http://www.unicode.org/cldr/<br>Split other languages than english into own extensions (language packs).<br>Some bugfixes. See doc/changelog.txt.<br>Changed encoding to UTF-8.<br>Provide encoding conversion in in EM (choose "UPDATE!").<br>Other languages than English are provided by extra language pack extensions.<br>Some bugfixes.<br>Extended static_country_zones.<br>Published the cc_infotablesmgm extension for managing the data. |
| 2.0.0 | Merge fields and data as provided by sr_static_info.<br>Integrate frontend API from sr_static_info.<br>Add indicator of UN membership in table static_countries.<br>Update official and local names of countries.<br>Enable localization of country subdivision names.<br>Enable country variants of languages such as Brazilian Portuguese.<br>Convert locallang files to xml format. |
| 2.0.1 | New parameters to the function API containing WHERE clauses to filter the elements for select boxes in extensions using it.<br>Install the extensions div and lib to make this feature possible. |
| 2.0.2 | new countries Serbia, Montenegro and Guernsey |
| 2.0.3 | New table static_markets for international market places following the ISO-Standard 10383. |
| 2.0.4 | support for custom where clauses with static_info_tables for TCA in the backend |
| 2.0.5 | New function fetchCountries to get a list of countries by specific parameters or parts of names of countries in different languages. Parameters might be left empty. |
| 2.0.6 | Multiple countries can be selected now at once.<br>new Galician language<br>Store the charset used with the import in the EM settings by yourself! |