punkt.de

# pt_extlist documentation

punkt.de

# pt_extlist documentation

publisher: punkt.de

publication: 2010

print-out: 1

person responsible: Daniel Lienert, Michael Knoll

Copyright, brands and trademarks : The author will make every endeavor to consider in all publications copyrights of the used illustrations, sounds, video sequences and texts, to use illustrations, sounds, video sequences produced by himself, or to fall back on license-free illustrations, sounds, video sequences and texts. All brands and trade marks, mentioned within the internet offer, which may be registered and protected by third parties are unrestricted subject to the regulations of the respective valid laws and to the rights of the registered owners. However, due to the bare mention of an brand or trademark, one can not jump to the conclusion, that brand names are not protected by rights of third parties! The copyright for published objects, produced by the author himself, remains only with the author of the pages. A duplication or a use of such illustrations, sounds, video sequences and texts in other electronic or printed publications without the strict agreement of the author is not permitted.

disclaimer: The author does not take over any guarantee for the topicality, the correctness, completeness or quality of the information, made available. Liability claims against the author, concerning damage of idealistic or of material kind, which was caused by the use or not use of the presented information and/or by the use of incorrect and incomplete information, are in principle impossible, so far as not a deliberate or roughly negligent fault can be proved on the part of the author. The documents and graphics on this Web site can be affected by technical inaccuracies or misprints, for which we don't assume any liability. punkt.de any time and without announcement can carry out technical amendments or improvements at the products which don't have to be documented absolutely on this Web site. Therefore punkt.de doesn't take any guarantee for the correctness of the details on this Web site. A legally binding contract on no account takes place alone by the information given here. Please consult us before you make use of the information given here for your application. The author expressly reserves itself the right, to change, to supplement or to delete parts of the pages or the entire offer or occasionally or finally to stop the publication without separate announcement.

Reading this ..... iv
1. Introduction ..... 1
1. What does it do? ..... 1
2. xxx ..... 3
1. Examples ..... 3
1.1. Setting up a demo list based on static_countries table ..... 3
3. Set up and Configuration ..... 10
1. Setting up Lists ..... 10
1.1. Widgets overview ..... 10
1.2. TypoScript configuration ..... 10
1.3. Setting up widgets as content elements ..... 17
4. Architecture ..... 18
1. Architecture ..... 18
1.1. Who should read this? ..... 18
1.2. Overview ..... 18
1.3. Basic Architecture ..... 19
1.4. Configuration ..... 19
1.5. Handling State ..... 20
1.6. The Data Backend ..... 21
1.7. The domain model ..... 23
1.8. List Rendering ..... 23
5. xxx ..... 24
1. API ..... 24
1.1. Extending pt_extlist ..... 24
2. Cookbook ..... 27
2.1. Testaufgabe ..... 27
6. TSREf ..... 28
1. pt_extlist ..... 28
7. Developer Guide ..... 103
1. Extending pt_extlist ..... 103
1.1. RenderChain ..... 103

# Reading this

| | |
|---|---|
| author | Michael Knoll, Daniel Lienert |
| version | 1.0 |
| status | draft |
| confidentiality | internal |
| document type | extension documentation |
| start | 18.10.2010 |
| last date of editing | 03.02.2011 |
| last date of printing | - |

**Table 1. Information about the document**

A documentation for users, administrators and developers.

The users will learn how to install and setup a demo list. The administrators will learn setting up a list and plugin configuration. The developers will learn what pt_extlist is about and how they can manipulate pt_extlist.

The users and administrators need the basic knowledge about programming.

# Chapter 1. Introduction

## 1. What does it do?

This extension is intended to generate all sorts of lists. The data sources for the list can be a database or an extbase repository or anything you write a data-backend for (SOAP, CSV, XML, ...).

There are different steps you have to do if you want to set up a list. For a detailed example see section "Instruction" ###TODO insert link here###.

Here are some screenshots to give you an impression of how it looks like.

| Country name (local country name) | Capital | ISO | Phone | Continent |
|---|---|---|---|---|
| Andorra Details | Andorra la Vella | AD | 376 | Europe |
| Afghanistan (افغانستان) Details | Kabul | AF | 93 | Asia |
| Antigua and Barbuda Details | St John's | AG | 1268 | Americas |
| Anguilla Details | The Valley | AI | 1264 | Americas |
| Albania (Shqipëria) Details | Tirana | AL | 355 | Europe |
| Netherlands Antilles (Nederlandse Antillen) Details | Willemstad | AN | 599 | Americas |
| Angola Details | Luanda | AO | 244 | Africa |
| Argentina Details | Buenos Aires | AR | 54 | Americas |
| American Samoa (Amerika Samoa) Details | Pago Pago | AS | 685 | Oceania |
| Austria (Österreich) Details | Vienna | AT | 43 | Europe |

**Figure 1.1. List rendered from static_countries table**

Besides the list itself, there are some more widgets that can be created by pt_extlist:

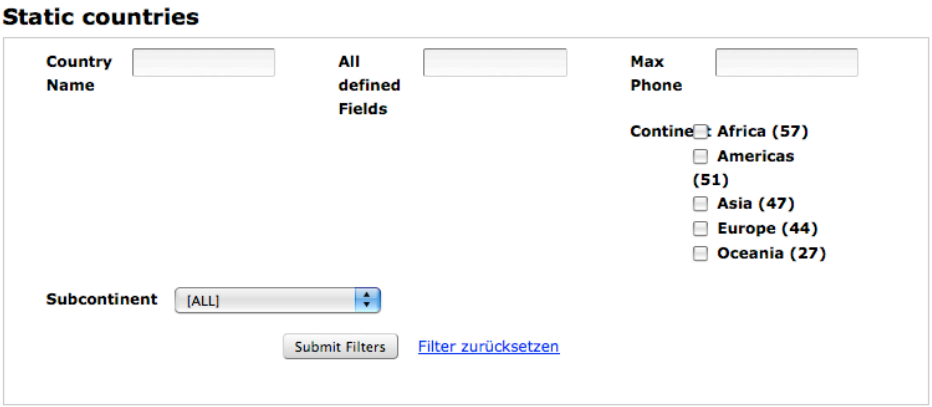


**Figure 1.2. Filters for static_countries table**

Zeige Element 1 bis 10 von 226

<< < 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 > >>

**Figure 1.3. Pager for static_countries table**

The plugin's flexform lets you insert a pt_extlist plugin as a content element where you can configure your plugin's appearance:

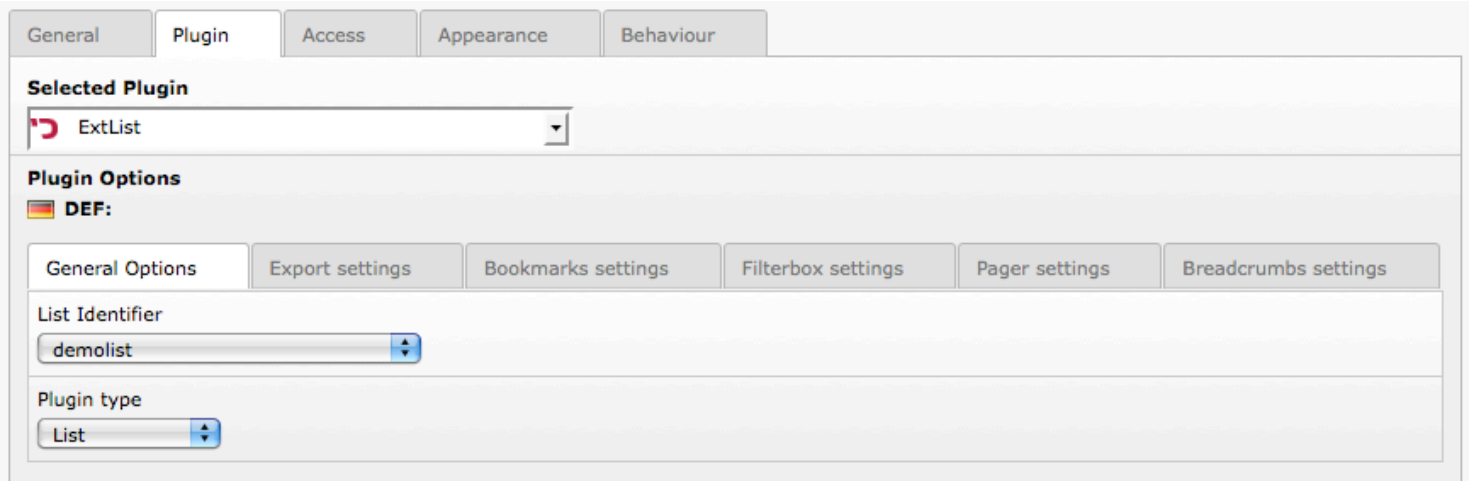


**Figure 1.4. Flexform for inserting plugin**

You can put several content elements on a page for setting up the layout and appearance of your widgets:

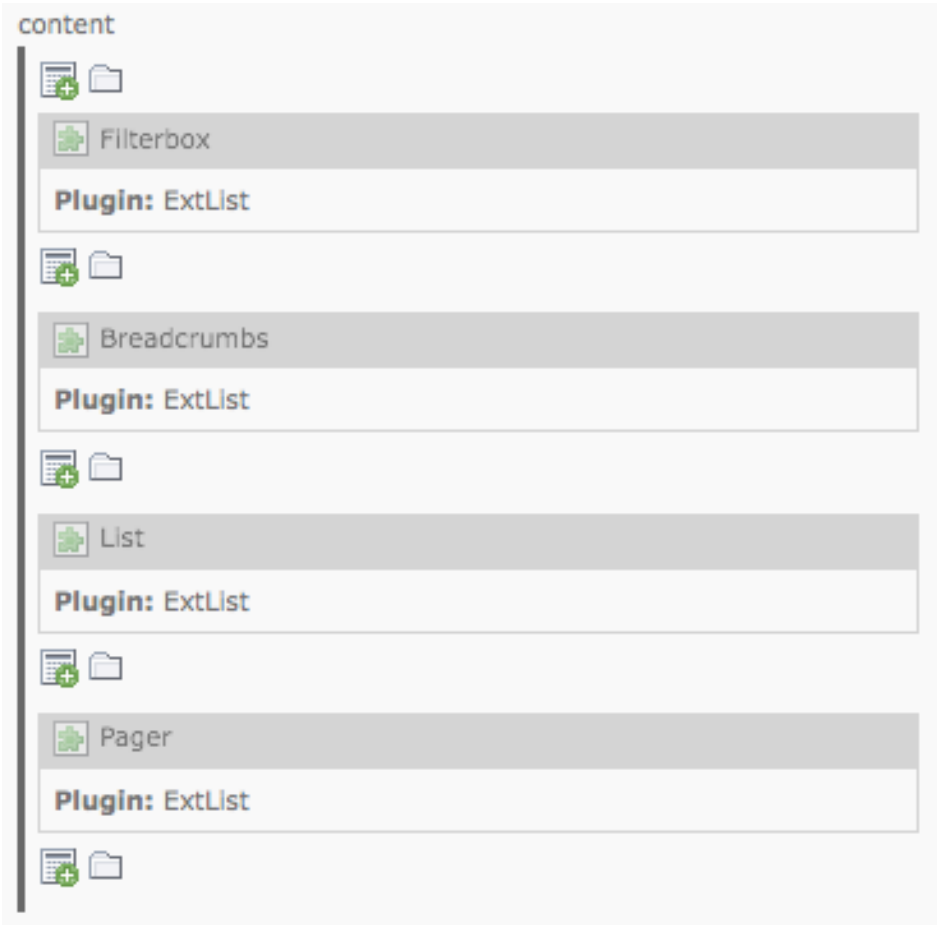


**Figure 1.5. Content elements for pt_extlist**

# Chapter 2. xxx

## 1. Examples

In this section, some examples will be provided to describe the functionality of pt_extlist. Before you can start, make sure, that pt_extlist is installed and loaded using Extension Manager.

### 1.1. Setting up a demo list based on static_countries table

In this example, you will learn how to create a list by using a TYPO3 table as data source. We will use static_countries table as it is available on all TYPO3 installations.

We will set up a page showing filters, list and pager for static countries.

1. Create a new page inside your page tree and open the template module. Open Template module and create new extension template:

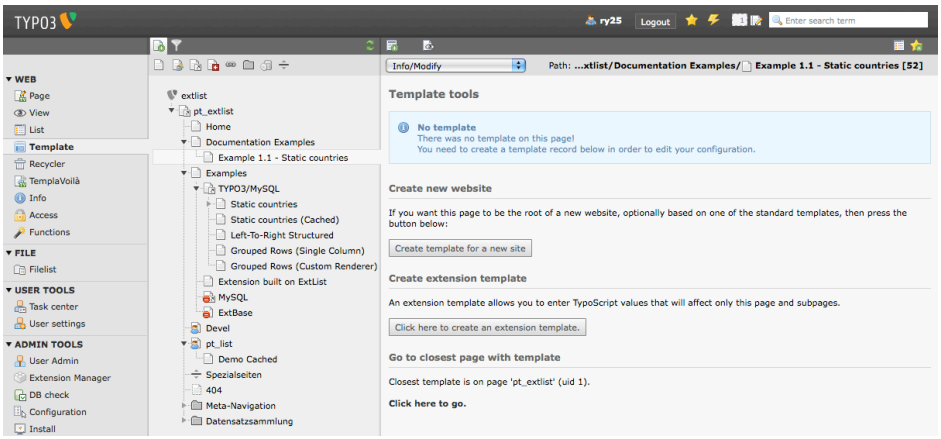


**Figure 2.1. Create new extension template ###TODO### insert 1,2,3 for showing what to do in image**

2. Give your extension template a proper name:

**Figure 2.2. Give your extension template a proper name**

3. Switch to the "Includes" Tab and select the following templates:

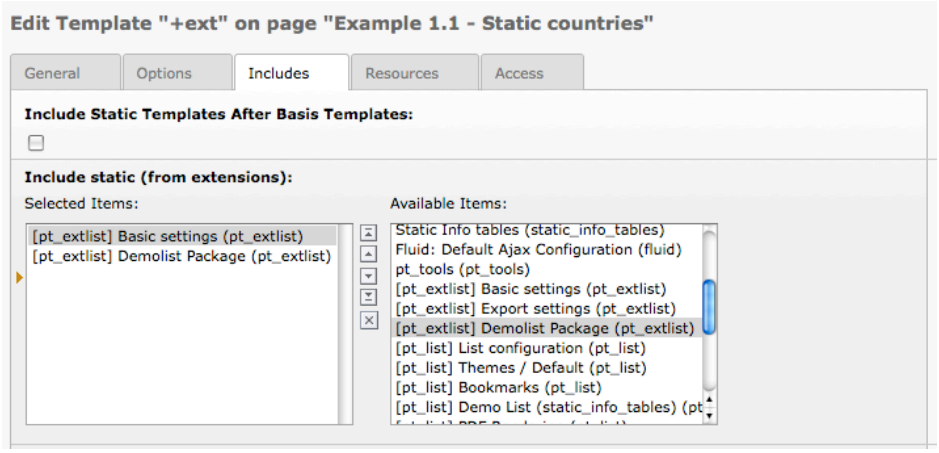


**Figure 2.3. Select Basic settings and demolist package as static templates**

4. Save your template and switch to the page module.

5. Select the page you just created and insert a new content element of type "plugin":

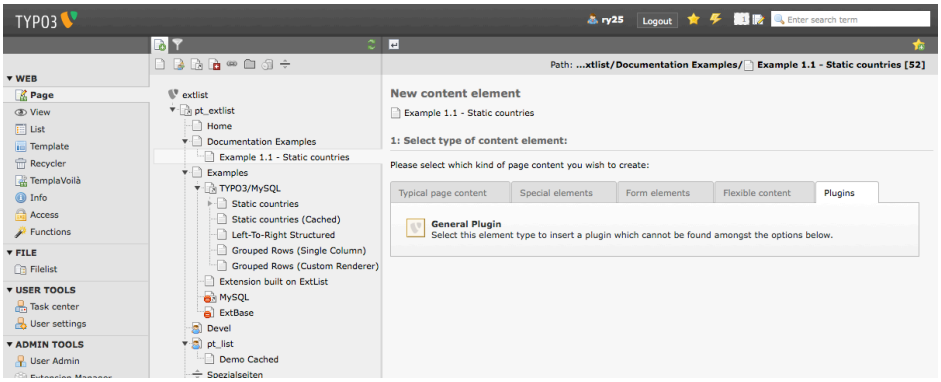


**Figure 2.4. Insert plugin as content element**

6. Select ExtList from the page content's "Selected Plugin" list:

**Figure 2.5. Select ExtList as content type**

7. In the flexform for ExtList select "demolist"

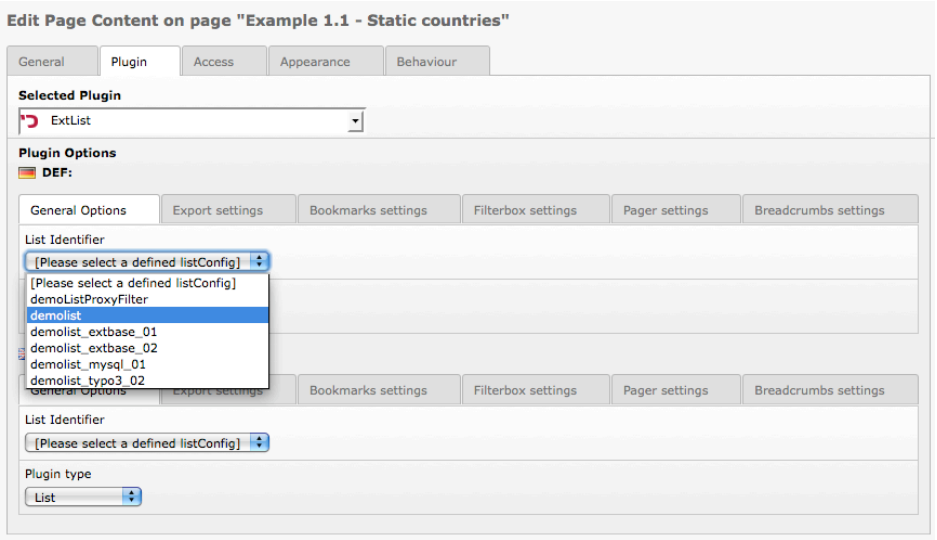


**Figure 2.6. Select "demolist" as list identifier**

8. As plugin type select "Filterbox":

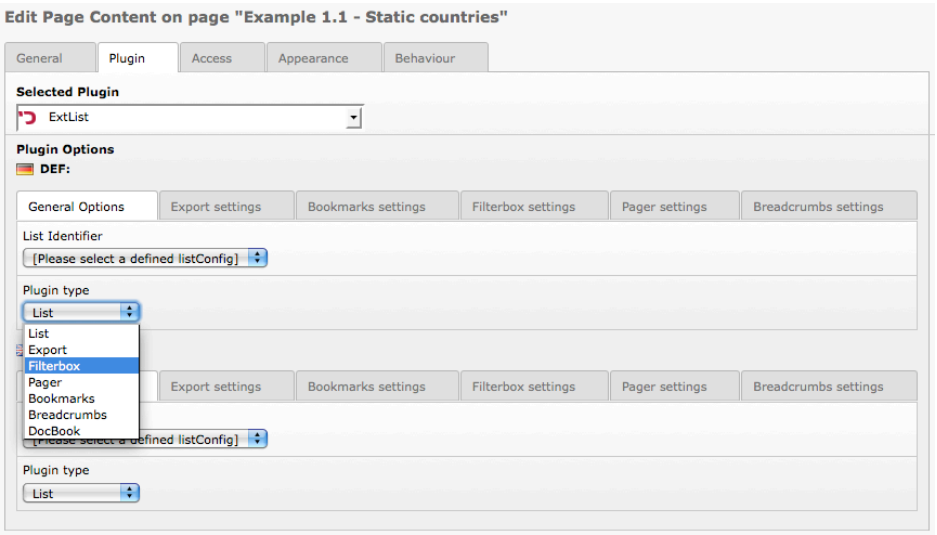


**Figure 2.7. Select "Filterbox" as Plugin Type**

9. Switch to the "Filterbox settings" Tab and input "filterbox1" as Filterbox Identifier:

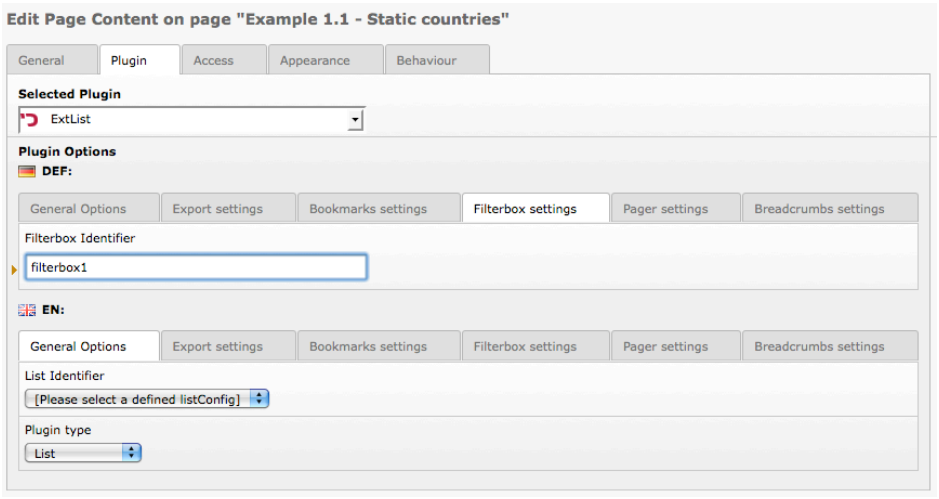


**Figure 2.8. Setting the filterbox identifier**

10. Save your content element and create another one just below. Select "Plugin" as content type and "ExtList" as plugin type just as you did before. Again select "demolist" as list identifier (steps 5 - 7 above), but this time select "list" as plugin type:

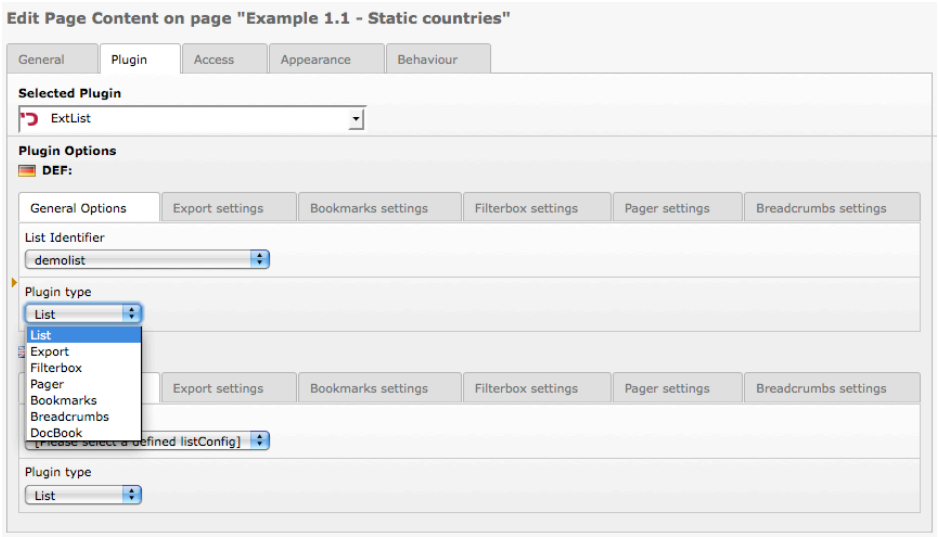


**Figure 2.9. Select "List" as Plugin Type**

11. Save and create a third content element. Repeat steps 5 - 7 from above, the select "demolist" as List Identifier and select "Pager" as Plugin Type:

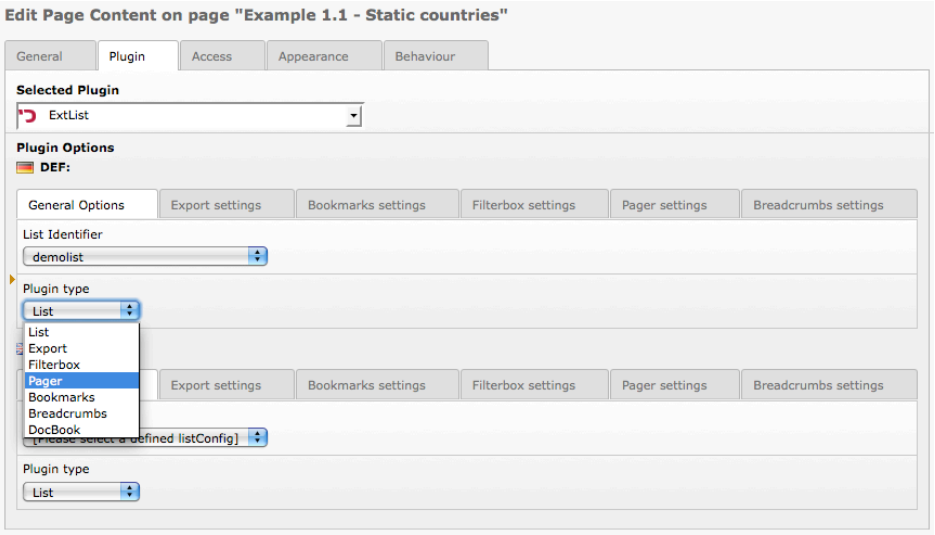


**Figure 2.10. Select "Pager" as Plugin Type**

12. Save content element and take a look at the page in the Frontend. Depending on your CSS Styles, it should look somehow like that:

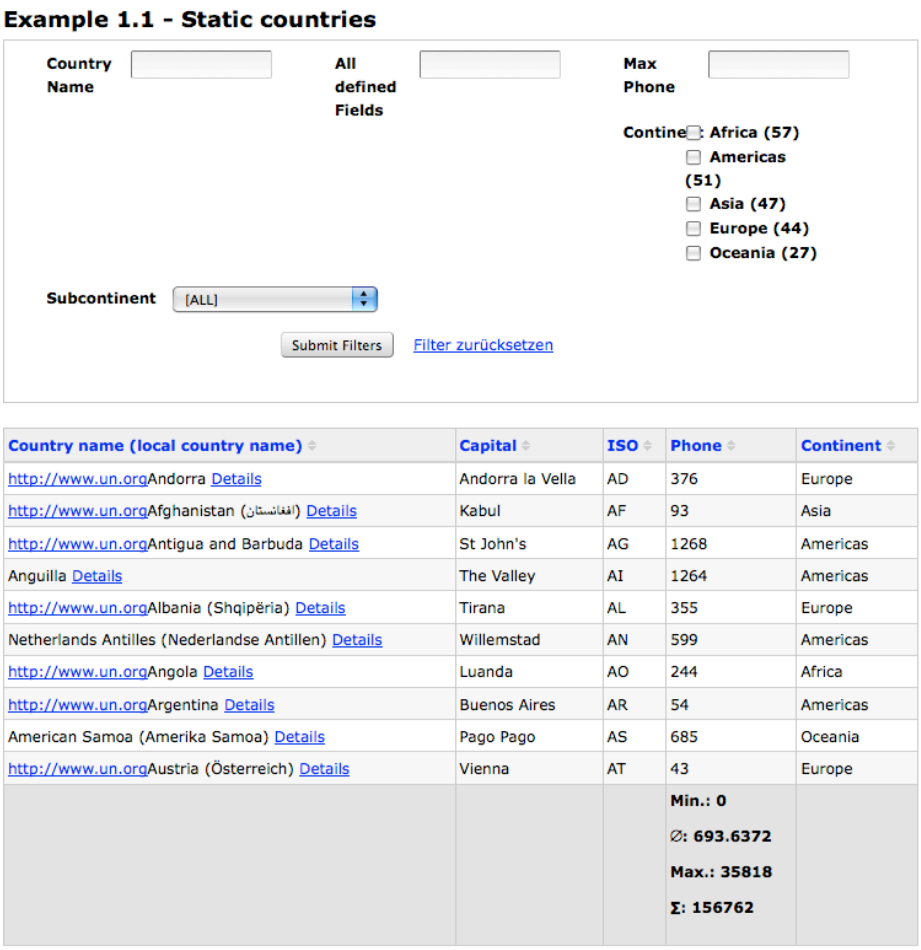


**Figure 2.11. Frontend view of ExtList widgets**

13 Now let's do a little more advanced stuff and change the number of records shown per page. Therefore switch to the Template module and select the page where you added the content elements above. Write the following line of code into your setup field:

```
plugin.tx_ptextlist.settings.listConfig.demolist.pager.itemsPerPage = 4
```

Now reload your page in the Frontend and look what's happening - there should be only 4 records per page anymore:

| Country name (local country name) | Capital | ISO | Phone | Continent |
|---|---|---|---|---|
| http://www.un.orgAndorra Details | Andorra la Vella | AD | 376 | Europe |
| http://www.un.orgAfghanistan (افغانستان) Details | Kabul | AF | 93 | Asia |
| http://www.un.orgAntigua and Barbuda Details | St John's | AG | 1268 | Americas |
| Anguilla Details | The Valley | AI | 1264 | Americas |
| | | | **Min.: 0**<br>**∅: 693.6372**<br>**Max.: 35818**<br>**Σ: 156762** | |

**Figure 2.12. List after changing items per page**

So that's it - you just set up your first list! Feel free to test the other sample configurations shipping with pt_extlist to see some more features.

# Chapter 3. Set up and Configuration

## 1. Setting up Lists

In this section you will learn how to set up lists using pt_extlist. We will guide you step by step through the TypoScript configuration and show you how to use pt_extlist's widgets as page content.

### 1.1. Widgets overview

Get an overview of what the individual widgets are doing and how they look like in the frontend. All widgets depend on a list identifier set up in TypoScript and selected within the FlexForm of your plugin.

#### 1.1.1. List widget

Renders a list of data set up by configuration. Can use headers for sorting list data by certain columns.

#### 1.1.2. Filter widgets

Renders filterboxes containing multiple filters defined by configuration.

#### 1.1.3. Pager widget

Renders a pager as configured by configuration. Pager limits rows of list to configured amount per page.

#### 1.1.4. Breadcrumbs widget

Breadcrumbs show which filters are activated and which values they have.

#### 1.1.5. Bookmarks widget

Bookmarks enable user to save certain list settings like filters, pager, sortings and reload them again afterwards.

### 1.2. TypoScript configuration

#### 1.2.1. List Identifier and TypoScript namespace

Each list has its own identifier.

#### 1.2.2. Sample Configuration

The following listing shows a sample configuration as it ships with pt_extlist.

```
plugin.tx_ptextlist.settings {

 _LOCAL_LANG.default.emptyList = empty list

 listConfig.demolist < plugin.tx_ptextlist.prototype.listConfig.default
 listConfig.demolist {
```

```
 backendConfig < plugin.tx_ptextlist.prototype.backend.typo3
 backendConfig {

  datasource {
   # no configuration required here
  }

  tables (
   static_countries,
   static_territories st_continent,
   static_territories st_subcontinent
  )

  baseFromClause (
   static_countries
   LEFT JOIN static_territories AS st_subcontinent ON
(static_countries.cn_parent_tr_iso_nr = st_subcontinent.tr_iso_nr)
   LEFT JOIN static_territories AS st_continent ON (st_subcontinent.tr_parent_iso_nr =
st_continent.tr_iso_nr)
  )

  baseWhereClause (
   st_continent.tr_name_en <> ""
   AND st_subcontinent.tr_name_en <> ""
  )
 }

 fields {
  name_local {
   table = static_countries
   field = cn_short_local
   isSortable = 1
  }

  name_en {
   table = static_countries
   field = cn_short_en
  }

  uno_member {
   table = static_countries
   field = cn_uno_member
  }

  capital {
   table = static_countries
   field = cn_capital
  }

  iso2 {
   table = static_countries
   field = cn_iso_2
   isSortable = 0
  }

  phone {
   table = static_countries
   field = cn_phone
  }

  isoNo {
   table = static_countries
   field = cn_currency_iso_nr
  }
```

```
   continent {
    table = st_continent
    field = tr_name_en
   }

   subcontinent {
    table = st_subcontinent
    field = tr_name_en
   }

   countryuid {
    table = static_countries
    field = uid
   }
  }

  pager {
   pagerConfigs {
    second {
     enabled = 1
     pagerClassName = Tx_PtExtlist_Domain_Model_Pager_DefaultPager
     templatePath = EXT:pt_extlist/Resources/Private/Templates/Pager/second.html

     showNextLink = 1
     showPreviousLink = 1
     showFirstLink = 0
     showLastLink = 0
    }
   }
  }

  columns {

   10 {
    columnIdentifier = nameColumn
    label = LLL:EXT:pt_extlist/Configuration/TypoScript/Demolist/
locallang.xml:column_nameColumn

    fieldIdentifier = name_local, name_en, countryuid, uno_member
    isSortable = 1
    sorting = name_local

    renderObj = COA
    renderObj {
     5 = IMAGE
     5.if {
      value.data = field:uno_member
      equals = 1
     }
     5.file = EXT:pt_list/typoscript/static/demolist/un.gif
     5.stdWrap.typolink.parameter = http://www.un.org
     5.stdWrap.typolink.ATagParams = class="un-link"

     10 = TEXT
     10.data = field:name_en
     10.append = TEXT
     10.append {
      data = field:name_local
      if {
       value.data = field:name_local
       equals.data = field:name_en
       negate = 1
      }
     }
     10.append.noTrimWrap = | (|)|
```

```
     10.wrap3 = | 

     20 = TEXT
     20.value = Details
     20.typolink.parameter = 1
     20.typolink.additionalParams.dataWrap =
 &tx_unseretolleextension_controller_details[countryuid]={field:countryuid}
    }
   }

   11 {
    label = Capital
    columnIdentifier = capital
    fieldIdentifier = capital
    cellCSSClass {
     renderObj = TEXT
     renderObj.dataWrap = {field:capital}
    }
   }

   20 {
    label = LLL:EXT:pt_extlist/Configuration/TypoScript/Demolist/
locallang.xml:column_isoNoColumn
    columnIdentifier = isoNoColumn
    fieldIdentifier = iso2
    isSortable = 1
   }

   30 {
    label = Phone
    columnIdentifier = phoneColumn
    fieldIdentifier = phone
   }

   40 {
    label = Continent
    columnIdentifier = continent
    fieldIdentifier = continent
   }

   50 {
    label = Subcontinent
    columnIdentifier = subcontinent
    fieldIdentifier = subcontinent
    accessGroups = 3
   }
  }

  aggregateData {
   sumPhone {
    fieldIdentifier = phone
    method = sum
   }
   avgPhone {
    fieldIdentifier = phone
    method = avg
   }
   maxPhone {
    fieldIdentifier = phone
    method = max
   }
   minPhone {
    fieldIdentifier = phone
    method = min
   }
```

```
  }

  aggregateRows {
   10 {
    phoneColumn {
     aggregateDataIdentifier = sumPhone, avgPhone, maxPhone, minPhone
     renderObj = TEXT
     renderObj.dataWrap (
      Min.: <b>{field:minPhone}</b><br />
      &empty;: <b>{field:avgPhone}</b><br />
      Max.: <b>{field:maxPhone}</b><br />
      &sum;: <b>{field:sumPhone}</b><br />
     )
    }
   }
  }

  filters {
   filterbox1 {
    filterConfigs {
     10 < plugin.tx_ptextlist.prototype.filter.string
     10 {
      filterIdentifier = filter1
      label = LLL:EXT:pt_extlist/Configuration/TypoScript/Demolist/
locallang.xml:filter_nameField
      fieldIdentifier = name_local
     }

     11 < plugin.tx_ptextlist.prototype.filter.string
     11 {
      filterIdentifier = allFields
      label = All defined Fields
      fieldIdentifier = *
     }

     15 < plugin.tx_ptextlist.prototype.filter.max
     15 {
      filterIdentifier = filter15
      label = Max Phone
      fieldIdentifier = phone
      accessGroups = 3
     }

     20 < plugin.tx_ptextlist.prototype.filter.checkbox
     20 {
      filterIdentifier = filter2
      label = Continent
      fieldIdentifier = continent
      filterField = continent
      displayFields = continent
      showRowCount = 1
      submitOnChange = 0
      invert = 0
      invertable = 0

      excludeFilters = filterbox1.filter3
     }

     30 < plugin.tx_ptextlist.prototype.filter.select
     30 {
      filterIdentifier = filter3
      label = Subcontinent
      fieldIdentifier = subcontinent
      filterField = subcontinent
      displayFields = continent, subcontinent
```

```
      multiple = 0
      showRowCount = 1
      submitOnChange = 0
      inactiveOption = [ALL]
      invert = 0
      invertable = 0
     }

    }
   }

   filterbox2 {
    showSubmit = 0
    showReset = 0
    filterConfigs {
     10 < plugin.tx_ptextlist.prototype.filter.firstLetter
     10 {
      filterIdentifier = filter4
      label = Capital
      fieldIdentifier = capital
     }
    }
   }
  }
 }
}


plugin.tx_ptextlist.settings.listConfig.demoListProxyFilter {
 backendConfig < plugin.tx_ptextlist.prototype.backend.typo3
 backendConfig {
  tables (
   static_territories
  )

  continent {
   table = static_territories
   field = tr_name_en
  }
 }

 fields {
  continent {
   table = static_territories
   field = tr_name_en
  }
 }

 filters {
  filterbox1 {
   filterConfigs {
    10 < plugin.tx_ptextlist.prototype.filter.select
    10 {
     filterIdentifier = continent
     label = Subcontinent
     fieldIdentifier = continent
     filterField = continent
     displayFields = continent
     showRowCount = 0
     multiple = 0
     inactiveOption = [ALL]

     renderObj = TEXT
     renderObj {
      dataWrap = {field:allDisplayFields}
```

```
     }
    }
   }
  }
 }
}

##################################
# Localization Override
##################################
plugin.tx_ptextlist._LOCAL_LANG{
 default {
  emptyList = List is empty.
 }
 de {
  emptyList = Liste ist leer.
 }
}
```

1.2.3. backendConfig section

1.2.3.1. Setting up a MySQL backend

1.2.3.2. Setting up a TYPO3 backend

1.2.3.3. Setting up a Extbase backend

1.2.4. fields section

1.2.4.1. Setting up fields for database backends

1.2.4.2. Setting up fields for Extbase domain objects

1.2.5. columns section

1.2.6. filters section

1.2.7. pager section

1.2.8. aggregateData section

1.2.9. aggregateRow section

1.2.10. Localization override

1.3. Setting up widgets as content elements

# Chapter 4. Architecture

This section gives you an overview of the architecture of pt_exlist. It shows its design principles and gives you a hint on where to make your hands dirty if you want to extend pt_extlist or use it in third party extensions.

## 1. Architecture

### 1.1. Who should read this?

If you are just interested in how pt_extlist works and want to see how things are put together, the following chapter could be a nice thing to read. If you are a developer and want to extend pt_extlist or want to use it as an API inside your extension, it's necessary to read this!

### 1.2. Overview

Take a look at the following figure to get a first impression of pt_extlist's architecture:

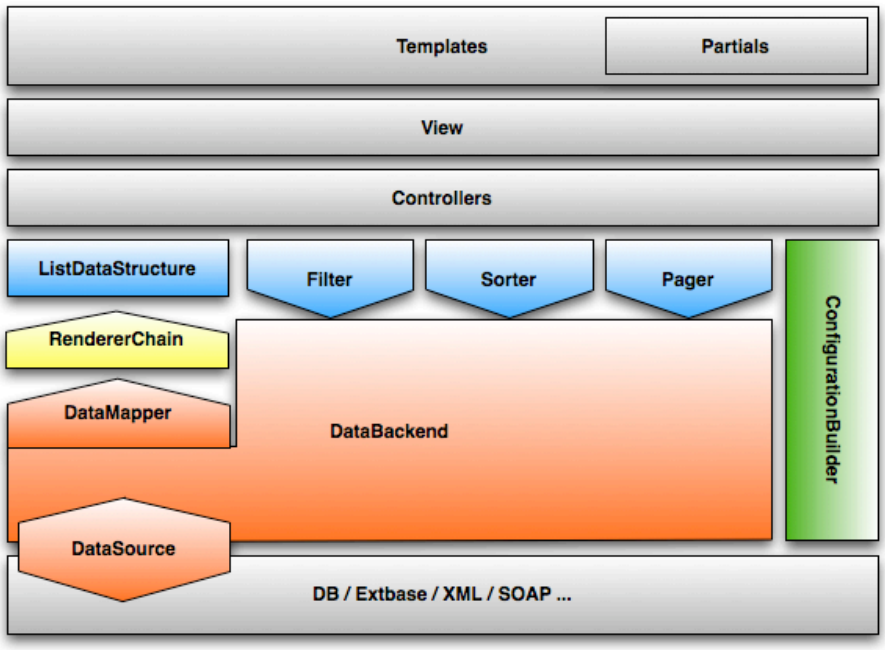


**Figure 4.1. Basic Architecture of pt_extlist**

The red elements stand for data-related stuff. DataSource gets our data from whatever source we use. Sources can be TYPO3 databases as well as arbitrary MySQL databases and Extbase repositories.

The green elements are configuration-specific which means that they pull our configuration from TypoScript or Flexforms and do various merging on them to generate some nice-to-handle objects that are used for all configuration inside pt_extlist.

The blue elements are what we would like to call our Domain Models as they handle things like lists, filters, pagers etc. which is the core domain of our extension.

Finally the yellow elements stand for all objects that handle the rendering of our list data. As we will see, this can get quite complex.

Our extension architecture is surrounded by some out-of-the box framework stuff provided mainly by Extbase and TYPO3.

## 1.3. Basic Architecture

### 1.3.1. Modell-View-Controller (MVC)

### 1.3.2. Independent Components

### 1.3.3. pt_extlist Lifecycle

## 1.4. Configuration

Regarding our extension in a bottom-up fashion, we can easily start with configuration as its the basic thing to have before we can set up anything else. As you might have suggested, most of our configuration is using in TypoScript. Besides there are some settings coming from Flexform or surrounding extensions. We covered this problem inside pt_extlist using what we call a configurationBuilder. Roughly speaking, it is an object that handles the creation of all configuration stuff we need. As we did not like fiddling around with arrays, we started to implement so called configuration objects for all the configuration that we needed. The main ascpects of those objects is to make sure that the required settings are there and correct. Configuration objects are passed to all other objects that require some kind of configuration.

Take a look at this diagram to get an idea of how everything works together:

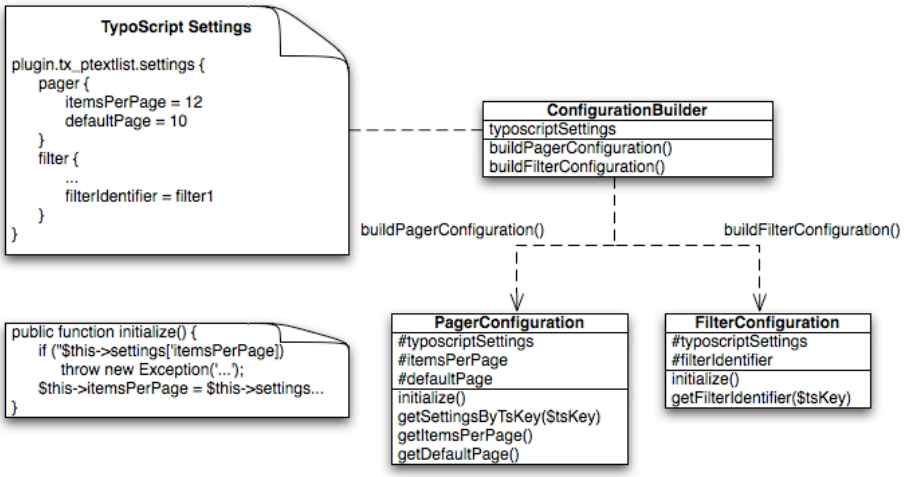


**Figure 4.2. Configuration Builder scheme**

So what's the basic idea of our so-called configuration objects?

1. TypoScript is a mighty tool for setting up configuration stuff. Unfortunatly there is no way to check for existence and correctness of settings. Our configuration objects jump in the gap here and present a single point of checking configuration.

2. Whenever you need a configuration set in TypoScript in your code, you have to check whether it is set or not. This probably has to be done several times and has to be changed several times whenever your configuration needs to be changed. Configuration objects enable you to keep your TS and your program logic synchronized by only changing code in one single class.
3. Configurations objects provide you with an object-oriented style of accessing configuration settings. You no longer have to fiddle around with arrays when you want to access configuration, now there are Getters and Setters and things like code-completion.

Before we had what we call the configuration builder, objects had to handle their configuration by themselves which made it necessary to implement huge init()-methods for setting up all configuration stuff. Whenever an object was not a singleton, the configuration stuff had to be run time and time again for every instantiation of an object.

As the configuration builder is a extension-wide singleton class which caches configuration objects once they are created, we now can simply grab a configuration from it, every time we need it and do not have to cope with checking the configuratin within our domain objects which makes them a lot simplier to implement and understand.

## 1.5. Handling State

Since Extbase has been around, handling with persistence of domain objects isn't that a big problem anymore. We use our repositories for finding, updating and deleting objects and that's it. But what about session persistence. Let's say, a user filters a list with some complicated criterias, selects one single record of it to look at and then comes back to his filtered list. Shouldn't it show the state it had before he opened the single record?

Such functionality is handled in pt_extlist with what we called a "Session-Persistence-Manager". Roughly speaking it is a container to which you can register objects that get a portion of data from the session when the object is created and can store data back to the session when the lifecycle ends.

Here is a sketch of how session persistence is working within pt_extlist:

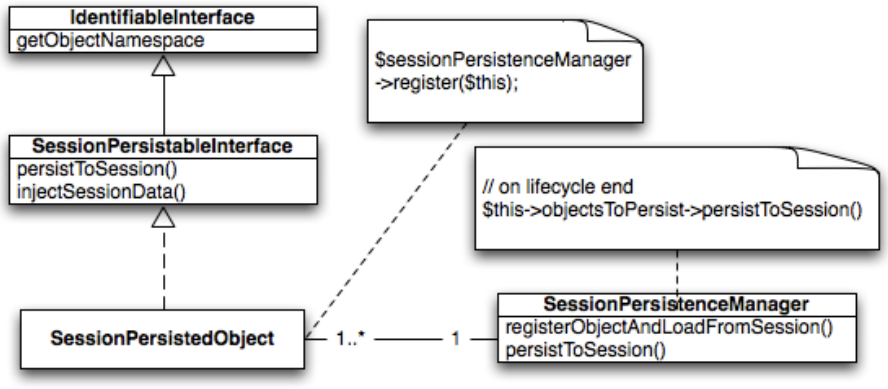


**Figure 4.3. Session Persistence in pt_extlist**

Please mind, that an object creates its own namespace within the session using the IdentifiableInterface. As we cannot automatically create a namespace for an object this method is implemented individually in each object that wants to use session persistence.

Another place where objects can gather state from are GET and POST parameters. Coming back to our filter example from above, a user could change a previously session-persisted filter so the data coming from the POST vars should overwrite the session settings.

Therefore, we introduced a GET/POST-Var adapter that works somehow similar as the session persistence manager, of course without being able to write back data to it as this would not make any sense.

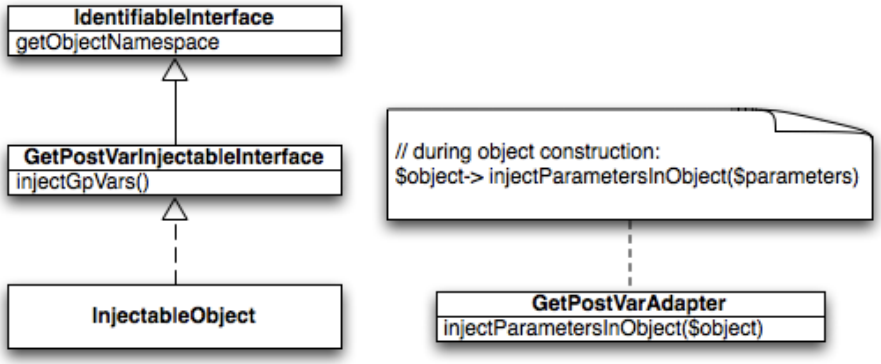


**Figure 4.4. GET/POST Var Adapter**

The GET/POST-Var adapter also handles the extension and instance namespace. Think about the following situation: you have two instances of pt_extlist plugin on the same page. Each instance comes with its own filters, pagers etc. and the should not influence each other. As an example think about a shopping page, where you have a list of articles on the left and a overview of your shopping cart as a second list on the right.

What the GP-Var adapter does is adding the list identifier (e.g. 'articleList' and 'shoppingCartList') to the namespace of the objects to differentiate between the two instances of the plugin.

## 1.6. The Data Backend

You can talk about the Data Backend as the heart of the pt_extlist extension. Somehow almost everything comes together here. Most of the pt_extlist components like filters, lists, pagers and breadcrumbs etc. influence the data backend or are themselves influenced or created by the data backend.

Besides being the "glue" holding all components of the extension together, the data backend has the taks of communicating with the "outer-world". This means, that whatever data is displayed within the extension is gathered by the backend from whatever datasource it is working on. The data backend also knows how to translate generic queries and constraints created from the components before they are passed to the data source.

###TODO### insert diagram for data backend

### 1.6.1. The Query Object

Former implementations of pt_extlist where bound to SQL-databases as a databackend. All queries created were SQL queries so you could directly create SQL snippets within all classes that manipulated the query. E.g. a filter class created some WHERE-clauses, a pager created a LIMIT-clause and a column header created an ORDER BY-clause for sorting.

One big change that comes with pt_extlist was being able to use whatever data source you like as a data backend. We therefore hat to generalize all queries that are created within our classes.

For this purpose, we introduced a so-called query object that can handle common SQL-query like functionality in an object oriented manner. Queries can take constraints (WHERE-clauses), limitations (LIMIT-clausses), sortings (ORDER BY-clauses) and some other stuff. But it's kept in a form that enables us to transform the query to whatever backend we want to use. At the moment there is a TYPO3 backend, using the actual TYPO3-database as a datasource and an Extbase backend that uses Repositories as data sources.

In order to send a query to the corresponding datasource, we have to translate it to a language that can be handled by the backend. We therefore introduced interpreters. Compared to the Extbase query object, those interpreters are independent of the query object itself, so that you can implement your own interpreters for whatever backend you want to support (e.g. XML via XPath or XQuery, SOAP, REST...).

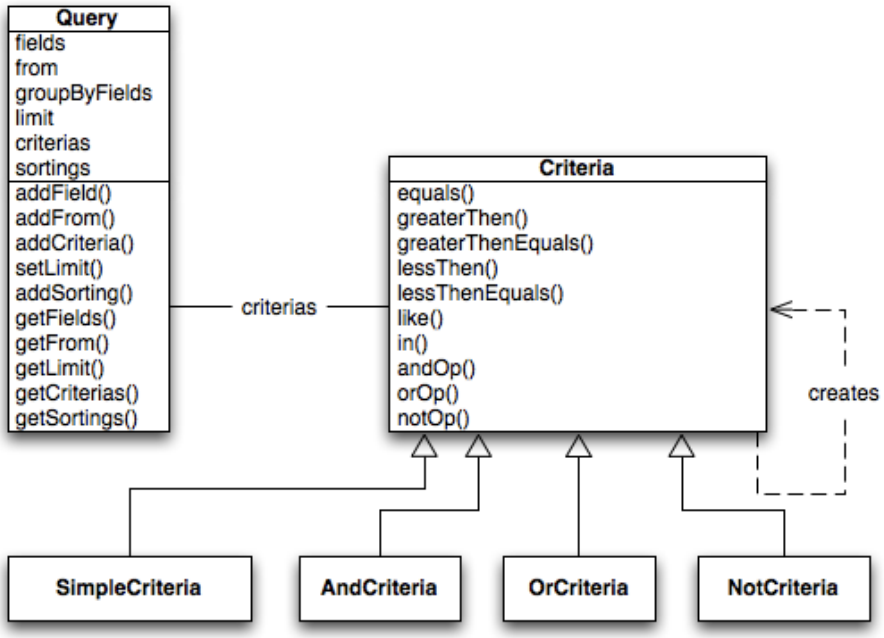


**Figure 4.5. Query Object**

The translation is handled by Interpreters shipping with a data-backend. Not every data-backend requires its own Interpreter, for example the MySQL backend and the TYPO3 backend share a common Interpreter as they both use SQL.

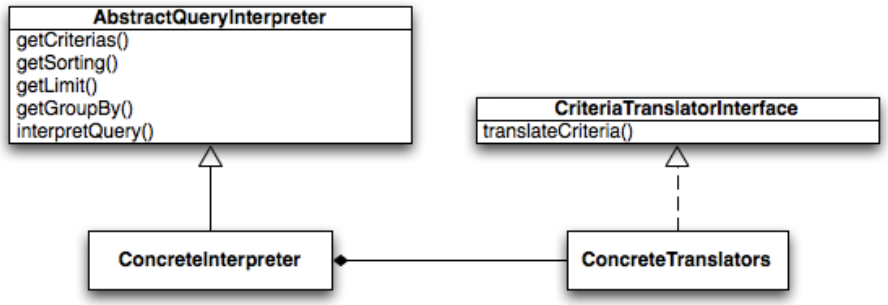


Figure 4.6.

### 1.6.2. Data Sources

### 1.6.3. Data Mappers

## 1.7. The domain model

### 1.7.1. The List object

### 1.7.2. Filters

### 1.7.3. Pager

### 1.7.4. BreadCrumbs

### 1.7.5. Bookmarks

## 1.8. List Rendering

### 1.8.1. The Renderer Chain

# Chapter 5. xxx

## 1. API

In this section you will learn how to use pt_extlist's API to extend the extension or use it within third-party extensions.

### 1.1. Extending pt_extlist

Pt_extlist can be extended in multiple ways. Many of its classes are configured via TypoScript so you can easily exchange them with your own classes to fit your needs. Common types of extensions are changing Data-Backends or writing your own filter classes. We will start with the latter one.

#### 1.1.1. Writing your own filter classes

Recapitulating what has been told about filters in the Architecture chapter, we reintroduce the following class diagram to understand what filters are actually doing:

**Figure 5.1. Filter Interface**

By taking a look at the Interface for filters, you see that there are mainly three main purposes:

1. Configuration and State-related stuff
2. Returning a filter query that determines what the filter is actually filtering on the data
3. Creating a filter breadcrumb information

Keeping in mind that there are some helpers - namely abstract classes - that do a lot of work for us we do not have to implement much logic when creating a new filter class:

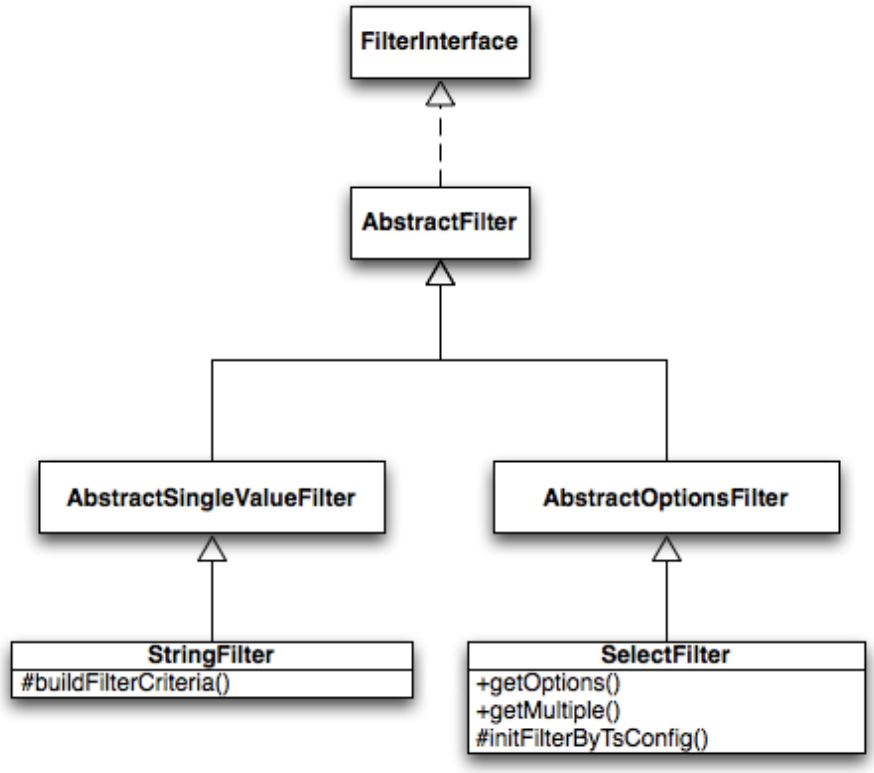


Figure 5.2. Abstract Filter Classes

So as you can see - all that's left for you to implement in your concrete filter class is a method that creates the actual filter criteria.

#### 1.1.1.1. String Filter Example

One of the most simple filters shipping with pt_extlist is the String filter. It can filter a string value based on a user input which is also a string. You can find the String-Filter class in the Classes/Domain/Model/Filter/StringFilter.php file.

Here is the PHP source code:

```
class Tx_PtExtlist_Domain_Model_Filter_StringFilter extends
 Tx_PtExtlist_Domain_Model_Filter_AbstractSingleValueFilter {

 /**
  * Creates filter query from filter value and settings
  *
  * @return Tx_PtExtlist_Domain_QueryObject_Criteria Criteria for current filter value
 (null, if empty)
  */
 protected function
 buildFilterCriteria(Tx_PtExtlist_Domain_Configuration_Data_Fields_FieldConfig
 $fieldIdentifier) {

  if ($this->filterValue == '') {
   return NULL;
  }
```

```
  $fieldName = Tx_PtExtlist_Utility_DbUtils::getSelectPartByFieldConfig($fieldIdentifier);
  $filterValue = '%'.$this->filterValue.'%';

  $criteria = Tx_PtExtlist_Domain_QueryObject_Criteria::like($fieldName, $filterValue);

  return $criteria;
 }
}
```

The most important function is *buildFilterCriteria()* where the filter creates a constraint on how the data filtered by this filter should look like. We use our generic query criteria *Tx_PtExtlist_Domain_QueryObject_SimpleCriteria* with an operator *like* here to implement a string filter that uses a LIKE-comparison in its built criteria. *Tx_PtExtlist_Domain_QueryObject_Criteria::like($fieldName, $filterValue)* is nothing more but a factory method that returns a criteria object.

As we mentioned above, a lot of functionality is given to us by our abstract classes, so to get some more information about what the String-Filter does and how it is configured, ake a look at its TypoScript prototype located in Configuration/TypoScript/BaseConfig/Prototype/Filter.txt:

```
string {
 filterClassName = Tx_PtExtlist_Domain_Model_Filter_StringFilter
 partialPath = Filter/String/StringFilter
 defaultValue =
 accessGroups =

 breadCrumbString = TEXT
 breadCrumbString {
  # Fields that can be used are "label" and "value"
  dataWrap = {field:label} equals {field:value}
 }
}
```

You find a lot more configuration possibilities here than you would assume after looking at the filter class above. First of all, there is a filterClassName, that determines which filter class to instantiate in order to create a string filter object. The partial path leeds us to the HTML template that is used for the filter's user interface. defaultValue lets us set a predefined value when the filter is shown for the first time and accessGroups restricts the filter to certain fe_groups that are allowed to see the filter.

breadCrumbString enables us to create a TS template for rendering the breadcrumb text of the filter.

The last thing we have to know, when we want to implement our own filter class is how to actually configure them within our list configuration. Therefore you should take a look at one of the demolists' filterbox configurations. There we find something like this:

```
filters {
 filterbox1 {
  filterConfigs {
   10 < plugin.tx_ptextlist.prototype.filter.string
   10 {
    filterIdentifier = filter1
    label = LLL:EXT:pt_extlist/Configuration/TypoScript/Demolist/
locallang.xml:filter_nameField
    fieldIdentifier = name_local
   }
  }
 }
```

```
}
```

All the filters of a list configuration are configured in the *filters* section of your configuration. Within this section you have to set up a arbitrary key for the name of your filterbox. In the example above, this is *filterbox1*. For each filterbox, you have to set up a list of filters within *filterConfigs* and in there we finally have our String-Filter. The basic settings are copied from the prototype above, then we have to change the settings that are unique for our usasge of the filter like filterIdentifier, label and the fieldIdentifier we want to let our filter operate on.

# 2. Cookbook

This chapter is a collection of recipes for common tasks in pt_extlist.

## 2.1. Testaufgabe

Beschribung der Aufgabe

**Figure 5.3.**

- 

1.

# Chapter 6. TSREf

## 1. pt_extlist

TypoSript Reference

### plugin.tx_ptextlist.settings

plugin.tx_ptextlist.settings

#### TS-Key:

plugin.tx_ptextlist.settings

#### Description

Main TS-key for all pt_extlist settings.
**CObject:** NO

#### Child elements

listConfig,

### listConfig

listConfig

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig

#### Description

Holds configuration for all list identifiers configured by array key.
**Datatype:** Associative Array (listIdentifier => listConfiguration)
**CObject:** NO

#### Example

```
plugin.tx_ptextlist.settings {

 listConfig {
  listIdentifier1 {

   # Here goes the configuration for your first list
   ...
  }

  listIdentifier2 {
```

```
   # Here goes the configuration for your second list
   ...
  }

  # Feel free to add more lists...
  ...
 }
}
```

#### Child elements

base, [yourListId],

## base

base

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.base

#### Description

Basic settings for pt_extlist
**CObject:** NO

#### Child elements

uncachedSessionStorageAdapter, cachedSessionStorageAdapter,

## uncachedSessionStorageAdapter

uncachedSessionStorageAdapter

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.base.uncachedSessionStorageAdapter

#### Description

Define the session storage adapter that should be used in uncached mode. Posible values are the classes described below or any other class that implements the interface Tx_PtExtbase_State_Session_Storage_AdapterInterface.
**Datatype:** String
**Posible values:** Tx_PtExtbase_State_Session_Storage_SessionAdapter: Default browser session storage adapter. Tx_PtExtbase_State_Session_Storage_FeUserSessionAdapter: TYPO3 fe_user bases session storage.
**Default:** Tx_PtExtbase_State_Session_Storage_SessionAdapter
**CObject:** NO

## cachedSessionStorageAdapter

cachedSessionStorageAdapter

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.base.cachedSessionStorageAdapter

#### Description

Define the session storage adapter that should be used in cached mode. Posible values are the classes described below or any other class that implements the interface Tx_PtExtbase_State_Session_Storage_AdapterInterface.
**Datatype:** String
**Posible values:** Tx_PtExtlist_Domain_StateAdapter_Storage_NullStorageAdapter: Hold every session variable as GET parameters in the URL. Use this adapter if you want to use realURL. Tx_PtExtlist_Domain_StateAdapter_Storage_DBStorageAdapter: Save the session variables into the database and add an identifier for this db-entry to the GET variables. Use this adapter if you want to handle many parameters and keep the URL length constant.
**Default:** Tx_PtExtlist_Domain_StateAdapter_Storage_NullStorageAdapter
**CObject:** NO

## [yourListId]

[yourListId]

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId]

#### Description

Holds configuration for a single list identifier. This is where the main part of your list configuration has to be set up.
**CObject:** NO

#### Example

```
plugin.tx_ptextlist.settings {

 listConfig {
  demolist_mysql_01 {

   backendConfig < plugin.tx_ptextlist.prototype.backend.mysql
   backendConfig {
    ...
   }

   fields {
```

```
    ...
   }

   columns {
    ...
   }

   filters {
    ...
   }

   pager {
    ...
   }
  }
 }
}
```

**Child elements**
default, backendConfig, fields, columns, rendererChain, aggregateData, aggregateRows, pager, filters, headerPartial, bodyPartial, agregateRowsPartial,

## default

default

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].default

**Description**

List default values.
**CObject:** NO

**Child elements**
sortingColumn,

## sortingColumn

sortingColumn

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].default.sortingColumn

**Description**

The default sorting column while no other sortng ist set. Direction can be set with DESC or ASC after the column identifier.

**Datatype:** String
**Posible values:** Any column identifier
**CObject:** NO

#### Example

```
// use capital as default sorting column in ascending order
sortingColumn = capital

// use capital as default sorting column in descending order
sortingColumn = capital DESC
```

## backendConfig

backendConfig

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig

#### Description

Holds the configuration for the data backend to be used with this configuration. Configurable values depend on individual backend, e.g. TYPO3-Backend, MySQL-Backen, ExtBase-Backend etc.
**CObject:** NO

#### Variants

backendConfig.mysql, backendConfig.typo3, backendConfig.extbase,

#### Variants of backendConfig:

## backendConfig.mysql

backendConfig.mysql

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql

#### Description

Backend configuration for connecting to native MySQL databases. DSN needs to be configured.
**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.backend.mysql

#### Example

```
backendConfig {

 dataSource {
  username = t3ry21ptldev
  password = xxxxxx
  databaseName = t3ry21ptldev
 }

 tables (
  City,
  Country
 )

 baseFromClause (
  Country
  INNER JOIN City ON City.CountryCode = Country.Code
  INNER JOIN CountryLanguage ON CountryLanguage.CountryCode = Country.Code
 )

 baseWhereClause (
 )

 baseGroupByClause (
  Country.Code, City.ID
 )
}
```

#### Child elements

datasource, baseFromClause, baseWhereClause, baseGroupByClause,

## datasource

datasource

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.datasource

#### Description

Defines datasource for (MySQL) data backend.
**CObject:** NO

#### Example

```
dataSource {
 host = mysql.db.hoster.tld
 username = funnyrabbit
 password = xxxxxxx
 databaseName = rabbitsdatabase
}
```

#### Child elements

host, port, databaseName, username, password,

## host

host

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.datasource.host

#### Description

Defines hostname of mysql database server.
**Datatype:** String
**Posible values:** All valid addresses for servers (IP-Addresses and URLs)
**Default:** localhost
**CObject:** NO

## port

port

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.datasource.port

#### Description

Port number on which MySQL database server is listening.
**Datatype:** Int
**Posible values:** Any valid port number
**Default:** 3306
**CObject:** NO

## databaseName

databaseName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.datasource.databaseName

#### Description

Name of database to be used for connection.
**Datatype:** String
**Posible values:** Any valid database name

**CObject:** NO

## username

username

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.datasource.username

#### Description

Username to be used for database connection.
**Datatype:** String
**Posible values:** Valid username for MySQL connection.
**CObject:** NO

## password

password

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.datasource.password

#### Description

Password to be used for database connection.
**Datatype:** String
**Posible values:** Any valid password.
**CObject:** NO

## baseFromClause

baseFromClause

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.baseFromClause

#### Description

Holds the FROM clause of a mysql query without the FROM keyword.
**Datatype:** String
**CObject:** YES

#### Example

```
static_countries
LEFT JOIN static_territories AS st_subcontinent ON (static_countries.cn_parent_tr_iso_nr =
 st_subcontinent.tr_iso_nr)
```

## baseWhereClause

baseWhereClause

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.baseWhereClause

#### Description

Holds the WHERE clause of a mysql query without the WHERE keyword.
**CObject:** YES

#### Example

```
# Example using cObject
baseWhereClause.cObject = COA
baseWhereClause.cObject {
    10 = TEXT
    10 {
        data = TSFE:fe_user|user|username
        noTrimWrap = | ticket.queue_id IN (SELECT queue_id FROM personal_queues WHERE
 user_id = (SELECT id FROM users WHERE login = "|")) |
    }
    20 = TEXT
    20 {
        data = TSFE:fe_user|user|username
        noTrimWrap = | AND ticket.user_id IN (1, (SELECT id FROM users WHERE login = "|"))
 |
    }
    30 = TEXT
    30 {
        data = TSFE:fe_user|user|username
        noTrimWrap = | OR ticket.user_id = (SELECT id FROM users WHERE login = "|") |
    }
}

# Example using  no cObject
baseWhereClause = fe_users.uid <> 0
```

## baseGroupByClause

baseGroupByClause

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.mysql.baseGroupByClause

#### Description

Holds the GROUP BY clause of a mysql query without the GROUP BY keyword.
**Datatype:** String
**CObject:** YES

#### Example

```
st_continent
```

## backendConfig.typo3

backendConfig.typo3

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.typo3

#### Description

TYPO3 Backend. Inherits the Features from the MySql Backend but the datasource is already configured. The TYPO3 Backend handles enableFields automatically in where clauses.
**CObject:** NO

#### Example

```
backendConfig < plugin.tx_ptextlist.prototype.backend.typo3
backendConfig {
 tables (
  static_countries,
  static_territories st_continent,
  static_territories st_subcontinent
 )

 baseFromClause (
  static_countries
  LEFT JOIN static_territories AS st_subcontinent ON (static_countries.cn_parent_tr_iso_nr
 = st_subcontinent.tr_iso_nr)
  LEFT JOIN static_territories AS st_continent ON (st_subcontinent.tr_parent_iso_nr =
 st_continent.tr_iso_nr)
 )

 baseWhereClause (
  st_continent.tr_name_en <> ''
  AND st_subcontinent.tr_name_en <> ''
 )
}
```

#### Child elements

tables, useEnableFields,

## tables

tables

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.typo3.tables

#### Description

Define tables to be used with enable fields. TYPO3 special fields (deleted, hidden) are automatically applied to query.
**Datatype:** Comma seperated list of table names and aliases (optional).
**CObject:** NO

#### Example

```
static_countries,
static_territories st_continent
```

## useEnableFields

useEnableFields

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.typo3.useEnableFields

#### Description

Use enableFields for alle tables that are defined with key tables.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

#### Example

```

```

## backendConfig.extbase

backendConfig.extbase

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].backendConfig.backendConfig.extbase

#### Description

Extbase Repository Backend.
**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.backend.extbase

## fields

fields

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].fields

#### Description

Defines raw datasource fields, wich can than combined and processed in to table fields.
**Datatype:** Associative array
**CObject:** NO

#### Child elements

[yourFieldId],

## [yourFieldId]

[yourFieldId]

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].fields.[yourFieldId]

#### Description

Named definition of a single data field.
**Datatype:** String
**CObject:** NO

#### Example

```
<![CDATA[
name_local {
 table = static_countries
 field = cn_short_local
 isSortable = 1
}
```

**Child elements**
table, field, special, isSortable, expandGroupRows,

## table

table

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].fields.[yourFieldId].table

**Description**

The table, this field belongs to.
**Datatype:** String
**CObject:** NO

## field

field

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].fields.[yourFieldId].field

**Description**

Name of the tables field.
**CObject:** NO

## special

special

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].fields.[yourFieldId].special

**Description**

Insert a individual SQL snippet.
**CObject:** NO

## isSortable

isSortable

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].fields.[yourFieldId].isSortable

**Description**

**CObject:** NO

## expandGroupRows

expandGroupRows

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].fields.[yourFieldId].expandGroupRows

**Description**

**CObject:** NO

## columns

columns

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns

**Description**

Holds the tables column definitions.
**Datatype:** Array
**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.column.default

**Example**

```
10 {
 label = LLL:EXT:pt_extlist/Configuration/TypoScript/Demolist/
locallang.xml:cn_short_localColumn
 columnIdentifier =  cn_short_localColumn
 fieldIdentifier =  cn_short_local
 isSortable = 1
}
```

**Child elements**
10,20,30,

## 10,20,30

10,20,30

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30

#### Description

**CObject:** NO

#### Child elements

fieldIdentifier, label, renderUserFunctions, renderTemplate, renderObj, sorting, sortingImageAsc, sortingImageDesc, sortingImageDefault, accessGroups, cellCSSClass,

## fieldIdentifier

fieldIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.fieldIdentifier

#### Description

Either a single identifier of a defined field, a comma separated list of identifiers or '*' to use all defined fields.
**Datatype:** fieldIdentifier or *
**CObject:** NO

#### Example

```
// use a single field
fieldIdentifier = cn_short_local

// use a list of fields
fieldIdentifier = cn_short_local, name_local

// use every defined field
fieldIdentifier = *
```

## label

label

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.label

#### Description

A label for this element.
**Datatype:** String
**CObject:** YES

#### Example

```
// as string
label = country

// as referenc to language file entry
label = LLL:EXT:pt_extlist/Configuration/TypoScript/Demolist/
locallang.xml:cn_short_localColumn
```

## renderUserFunctions

renderUserFunctions

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.renderUserFunctions

#### Description

A list of userfunctions to render the field value.
**CObject:** NO

#### Example

```
renderUserFunctions {
 10 = EXT:pt_extlist/Resources/Private/UserFunctions/
class.tx_ptextlist_demolist_renderer.php:tx_ptextlist_demolist_renderer->iso2CodeRenderer
}
```

## renderTemplate

renderTemplate

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.renderTemplate

#### Description

**CObject:** NO

## renderObj

renderObj

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.renderObj

#### Description

**CObject:** NO

## sorting

sorting

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.sorting

#### Description

Define sorting for this column. A single sorting definition consists either of a fieldidentifier or a fieldidentifier followed by direction. The direction can be fixed with adding a '!' in front of the direction.
**Datatype:** Seperated list of fieldIdentifiers.
**CObject:** NO

#### Example

```
// Order by name_local ASC, name_en ASC
sorting = name_local, name_en

// Order by name_local DESC, name_en ASC
sorting = name_local DESC, name_en

// Order by name_local DESC, name_en ASC
// Sorting by name_local is fixed to DESC with the "!" and does not toggle
sorting = name_local !DESC, name_en
```

## sortingImageAsc

sortingImageAsc

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.sortingImageAsc

#### Description

Image for ascending ordered column header.

**Default:** EXT:pt_extlist/Resources/Public/List/icon_table_sort_asc.png
**CObject:** NO

## sortingImageDesc

sortingImageDesc

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.sortingImageDesc

#### Description

Image for descending ordered column header.
**Default:** EXT:pt_extlist/Resources/Public/List/icon_table_sort_desc.png
**CObject:** NO

## sortingImageDefault

sortingImageDefault

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.sortingImageDefault

#### Description

Image for unordered column header.
**Default:** EXT:pt_extlist/Resources/Public/List/icon_table_sort_default.png
**CObject:** NO

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.accessGroups

#### Description

**CObject:** NO

## cellCSSClass

cellCSSClass

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].columns.10,20,30.cellCSSClass

#### Description

**CObject:** NO

## rendererChain

rendererChain

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].rendererChain

#### Description

Holds the renderer configuration.
**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.rendererChain

#### Example

```
rendererChain {
 rendererConfigs {
  110 < plugin.tx_ptextlist.prototype.renderer.columnGroup
  110 {
   columnIdentifier = ContinentColumn
   showRowCount = 1
  }
 }
}
```

#### Child elements

enabled, rendererConfigs,

## enabled

enabled

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].rendererChain.enabled

#### Description

**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

## rendererConfigs

rendererConfigs

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].rendererChain.rendererConfigs

#### Description

A list of chained renderer classes that work on list data structures. The default renderer class uses the column configuration, to render the list of field data in a list of rows and columns. By default this renderer iscalled at position 100. All defined renderer before 100 work on a field data list, while renderer after 100 work on a column list data structure.
**Datatype:** Array (10,20,30)
**CObject:** NO

#### Example

#### Child elements

[yourNumericRendererId],

## [yourNumericRendererId]

[yourNumericRendererId]

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].rendererChain.rendererConfigs.[yourNumericRendererId]

#### Description

Configuration of a single renderer.
**Posible values:** 10,20,30
**CObject:** NO

#### Child elements

renderClassName,

## renderClassName

renderClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].rendererChain.rendererConfigs.[yourNumericRendererId].renderClassNar

#### Description

The class name of the renderers php class.
**Datatype:** String
**CObject:** NO

## aggregateData

aggregateData

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].aggregateData

#### Description

Defines aggregates of data fields.
**Datatype:** Associative array
**CObject:** NO

#### Example

```
aggregateData {
 sumPhone {
  fieldIdentifier = phone
  method = sum
 }
}
```

#### Child elements

[yourAggregateFieldId],

## [yourAggregateFieldId]

[yourAggregateFieldId]

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].aggregateData.[yourAggregateFieldId]

#### Description

Named definition of a single data field.
**Datatype:** Associative array
**CObject:** NO

#### Example

```
sumPhone {
 fieldIdentifier = phone
 method = sum
 scope = query
}
```

#### Child elements

method, scope,

## method

method

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].aggregateData.[yourAggregateFieldId].method

#### Description

Defined aggregate methods.
**Datatype:** String
**Posible values:** min,max,sum,avg
**CObject:** NO

## scope

scope

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].aggregateData.[yourAggregateFieldId].scope

#### Description

The scope for the aggregation can be either set to the current page or to the whole query. Aggregates for the current page are calculated internally without an additonal database query.
**Datatype:** String
**Posible values:** page,query
**CObject:** NO

## aggregateRows

aggregateRows

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].aggregateRows

#### Description

Holds the aggregates columns definitions.
**Datatype:** Associative array
**Posible values:** All columnIdentifiers
**CObject:** NO

#### Example

```
aggregateRows {
 10 {
  phoneColumn {
   aggregateDataIdentifier = sumPhone
   renderObj = TEXT
   renderObj.dataWrap (
     &sum;: <b>{field:sumPhone}</b>
   )
  }
 }
}
```

#### Child elements

[yourColumnId],

## [yourColumnId]

[yourColumnId]

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].aggregateRows.[yourColumnId]

#### Description

**CObject:** NO

#### Child elements

aggregateDataIdentifier,

## aggregateDataIdentifier

aggregateDataIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].aggregateRows.[yourColumnId].aggregateDataIdentifier

#### Description

**Datatype:** String
**CObject:** NO

## pager

pager

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager

#### Description

Holds all pager configuration
**Datatype:** Associative array
**CObject:** NO

#### Child elements

itemsPerPage, pagerConfigs,

## itemsPerPage

itemsPerPage

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager.itemsPerPage

#### Description

Sets the items per page. A value of 0 means to show all entrys.
**Datatype:** Integer
**CObject:** NO

## pagerConfigs

pagerConfigs

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager.pagerConfigs

**Description**

Holds an array of all defined pagers
**Datatype:** Associative array
**CObject:** NO

**Child elements**

[yourPagerIdentifier],

## [yourPagerIdentifier]

[yourPagerIdentifier]

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager.pagerConfigs.[yourPagerIdentifier]

**Description**

Identifier of a pager
**Datatype:** String
**CObject:** NO

**Variants**

default, delta, minimal,

**Variants of [yourPagerIdentifier]:**

## default

default

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager.pagerConfigs.[yourPagerIdentifier].default

**Description**

Configuration for default pager
**CObject:** NO

## delta

delta

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager.pagerConfigs.[yourPagerIdentifier].delta

#### Description

Configuration for delta pager. Shows only a configured amount of pagenumbers around the current page.
**CObject:** NO

## minimal

minimal

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager.pagerConfigs.[yourPagerIdentifier].minimal

#### Description

Configuration for minimal pager. Shows only Forward and backward links without the page numbers.
**CObject:** NO

#### Child elements

pagerClassName, templatePath,

## pagerClassName

pagerClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager.pagerConfigs.[yourPagerIdentifier].pagerClassName

#### Description

Name of the PHP Class of this pager.
**Datatype:** String
**CObject:** NO

#### Example

```
pagerClassName = Tx_PtExtlist_Domain_Model_Pager_DefaultPager
```

## templatePath

templatePath

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].pager.pagerConfigs.[yourPagerIdentifier].templatePath

#### Description

Name of the template class for this pager.
**Datatype:** String Path
**CObject:** NO

#### Example

```
templatePath = EXT:pt_extlist/Resources/Private/Templates/Pager/Default.html
```

## filters

filters

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters

#### Description

Holds all filterbox configurations.
**Datatype:** Associative array
**CObject:** NO

#### Child elements

[yourFilterBoxId],

## [yourFilterBoxId]

[yourFilterBoxId]

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId]

#### Description

**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.filterBox

#### Child elements

showReset, showSubmit, filterConfigs,

## showReset

showReset

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].showReset

#### Description

Show a reset link for all filters of this filterBox.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

## showSubmit

showSubmit

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].showSubmit

#### Description

Show a submit button for this filterBox.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

## filterConfigs

filterConfigs

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs

#### Description

Holds the configuration of the filters of this filter box.
**Datatype:** Array
**Posible values:** 10,20,30...
**CObject:** NO

#### Child elements
[yourNumericFilterId],

## [yourNumericFilterId]

[yourNumericFilterId]

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId]

#### Description

**CObject:** NO

#### Variants
String, Select, Checkbox, Button, radiobutton, firstletter, tagcloud, Proxy,

#### Variants of [yourNumericFilterId]:

## String

String

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String

#### Description

Shows a string filter.
**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.filter.string

#### Example

```
10 < plugin.tx_ptextlist.prototype.filter.string
10 {
 filterIdentifier = firstNameSearch
 label = Firstname
 fieldIdentifier = firstName
}
```

#### Child elements
filterIdentifier, label, defaultValue, accessGroups, filterClassName, partialPath, invert, invertable,

## filterIdentifier

filterIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String.filterIdenti

#### Description

The unique identifier of this filter.
**Datatype:** String
**CObject:** NO

## label

label

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String.label

#### Description

The label wich is displayed beside the filter.
**Datatype:** String
**CObject:** NO

## defaultValue

defaultValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String.defaultVal

#### Description

The default value which is shown or selected by default.
**Datatype:** String
**CObject:** NO

#### Example

```
// Selects key 3 as default Value
defaultValue = 3

// Selects multiple values in a multi value filter (3 and 18)
```

```
defaultValue {
 10 = 3
 20 = 18
}

// Selects current user as default in an user select list
defaultValue.cObject = TEXT
defaultValue.cObject.data = TSFE:fe_user|user|username
```

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String.accessGr

#### Description

Comma separated list of user groups wich have access to this filter.
**Datatype:** Comma eparated list.
**Posible values:** TYPO3 Group Ids
**CObject:** NO

## filterClassName

filterClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String.filterClass

#### Description

Name of the PHP Class of this filter.
**Datatype:** String
**CObject:** NO

#### Example

```
Tx_PtExtlist_Domain_Model_Filter_StringFilter
```

## partialPath

partialPath

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String.partialPath

#### Description

**CObject:** NO

## invert

invert

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String.invert

#### Description

Invert the constraint of this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## invertable

invertable

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].String.invertable

#### Description

Show a controle to invert this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## Select

Select

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select

#### Description

**CObject:** NO

**Prototype:** plugin.tx_ptextlist.prototype.filter.select

#### Child elements

filterIdentifier, fieldIdentifier, label, partialPath, filterClassName, defaultValue, accessGroups, invert, invertable, displayFields, multiple, excludeFilters, showRowCount, submitOnChange, inactiveOption, inactiveValue, options,

## filterIdentifier

filterIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.filterIdenti

#### Description

The unique identifier of this filter.
**Datatype:** String
**CObject:** NO

## fieldIdentifier

fieldIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.fieldIdenti

#### Description

WARNING: Which one to choose??? Identifier of a defined data field. The filter affects on this data field. If more than on fieldIdentifier is given, the resulting query is linked with or. Either a single identifier of a defined field, a comma separated list of identifiers or '*' to use all defined fields.
**Datatype:** fieldIdentifier or *
**CObject:** NO

#### Example

```
// use a single field
fieldIdentifier = cn_short_local

// use a list of fields
fieldIdentifier = cn_short_local, name_local

// use every defined field
fieldIdentifier = *
```

## label

label

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.label

#### Description

The label wich is displayed beside the filter.
**Datatype:** String
**CObject:** NO

## partialPath

partialPath

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.partialPat

#### Description

**CObject:** NO

## filterClassName

filterClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.filterClass

#### Description

Name of the PHP Class of this filter.
**Datatype:** String
**CObject:** NO

#### Example

```
Tx_PtExtlist_Domain_Model_Filter_StringFilter
```

## defaultValue

defaultValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.defaultVa

#### Description

The default value which is shown or selected by default.
**Datatype:** String
**CObject:** NO

#### Example

```
// Selects key 3 as default Value
defaultValue = 3

// Selects multiple values in a multi value filter (3 and 18)
defaultValue {
 10 = 3
 20 = 18
}

// Selects current user as default in an user select list
defaultValue.cObject = TEXT
defaultValue.cObject.data = TSFE:fe_user|user|username
```

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.accessGr

#### Description

Comma separated list of user groups wich have access to this filter.
**Datatype:** Comma eparated list.
**Posible values:** TYPO3 Group Ids
**CObject:** NO

## invert

invert

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.invert

#### Description

Invert the constraint of this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## invertable

invertable

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.invertable

#### Description

Show a controle to invert this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## displayFields

displayFields

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.displayFie

#### Description

One or multiple identifiers of previously defined data fields.
**Datatype:** String / Comma separated list
**Posible values:** Previously defined field indentifier.
**Default:** If not set, the field defined in fieldIdentifier is used.
**CObject:** NO

## multiple

multiple

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.multiple

#### Description

**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## excludeFilters

excludeFilters

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.excludeFi

#### Description

List of filters that are not considered for the group query of this filter.
**Datatype:** String / Comma separated list of filterIdentifiers.
**CObject:** NO

## showRowCount

showRowCount

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.showRow

#### Description

Show the rowcount if the select filter is filled with grouped data.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

## submitOnChange

submitOnChange

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.submitOn

#### Description

Instant submit filter if a value is selected.

**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## inactiveOption

inactiveOption

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.inactiveO

#### Description

Label of an option that is added to the select list. If this option is selected, the filter is inactive.
**Datatype:** String
**CObject:** NO

## inactiveValue

inactiveValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.inactiveVa

#### Description

The submitted value for the inactiveOption described above.
**Datatype:** String
**CObject:** NO

## options

options

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Select.options

#### Description

Defines explicit options to display in a data filter. If options are set, no data is fetched from the database.
There are to variants to define the data. See the example for details.
**Datatype:** array
**CObject:** NO

#### Example

```
# There are two variants to define the data

# Variant 1: Simple key / value
options {
 checkedin = Someone Checked in
 noonecheckedin = Noone Checked In
}

# Variant 2: Use integer keys to define option order and subKeys key / value to define the
 options
10 {
    key = 1
    value = LLL:EXT:my_ext/locallang.xml:theLabelOfTheOption
}

20 {
    ...
}
```

## Checkbox

Checkbox

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox

#### Description

**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.filter.checkbox

#### Child elements

filterIdentifier, fieldIdentifier, label, partialPath, filterClassName, defaultValue, accessGroups, invert, invertable, displayFields, multiple, excludeFilters, showRowCount, submitOnChange, inactiveOption, inactiveValue, options,

## filterIdentifier

filterIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.filterId

#### Description

The unique identifier of this filter.

**Datatype:** String
**CObject:** NO

## fieldIdentifier

fieldIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.fieldId

#### Description

WARNING: Which one to choose??? Identifier of a defined data field. The filter affects on this data field. If more than on fieldIdentifier is given, the resulting query is linked with or. Either a single identifier of a defined field, a comma separated list of identifiers or '*' to use all defined fields.
**Datatype:** fieldIdentifier or *
**CObject:** NO

#### Example

```
// use a single field
fieldIdentifier = cn_short_local

// use a list of fields
fieldIdentifier = cn_short_local, name_local

// use every defined field
fieldIdentifier = *
```

## label

label

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.label

#### Description

The label wich is displayed beside the filter.
**Datatype:** String
**CObject:** NO

## partialPath

partialPath

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.partial

#### Description

**CObject:** NO

## filterClassName

filterClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.filterC

#### Description

Name of the PHP Class of this filter.
**Datatype:** String
**CObject:** NO

#### Example

```
Tx_PtExtlist_Domain_Model_Filter_StringFilter
```

## defaultValue

defaultValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.defau

#### Description

The default value which is shown or selected by default.
**Datatype:** String
**CObject:** NO

#### Example

```

// Selects key 3 as default Value
defaultValue = 3

// Selects multiple values in a multi value filter (3 and 18)
defaultValue {
 10 = 3
 20 = 18
```

```
}

// Selects current user as default in an user select list
defaultValue.cObject = TEXT
defaultValue.cObject.data = TSFE:fe_user|user|username
```

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.acces

#### Description

Comma separated list of user groups wich have access to this filter.
**Datatype:** Comma eparated list.
**Posible values:** TYPO3 Group Ids
**CObject:** NO

## invert

invert

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.invert

#### Description

Invert the constraint of this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## invertable

invertable

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.inverta

#### Description

Show a controle to invert this filter.

**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## displayFields

displayFields

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.displa

#### Description

One or multiple identifiers of previously defined data fields.
**Datatype:** String / Comma separated list
**Posible values:** Previously defined field indentifier.
**Default:** If not set, the field defined in fieldIdentifier is used.
**CObject:** NO

## multiple

multiple

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.multip

#### Description

**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## excludeFilters

excludeFilters

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.exclu

#### Description

List of filters that are not considered for the group query of this filter.

**Datatype:** String / Comma separated list of filterIdentifiers.
**CObject:** NO

## showRowCount

showRowCount

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.showR

#### Description

Show the rowcount if the select filter is filled with grouped data.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

## submitOnChange

submitOnChange

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.submi

#### Description

Instant submit filter if a value is selected.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## inactiveOption

inactiveOption

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.inactiv

#### Description

Label of an option that is added to the select list. If this option is selected, the filter is inactive.
**Datatype:** String

**CObject:** NO

## inactiveValue

inactiveValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.inactiv

#### Description

The submitted value for the inactiveOption described above.
**Datatype:** String
**CObject:** NO

## options

options

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Checkbox.option

#### Description

Defines explicit options to display in a data filter. If options are set, no data is fetched from the database. There are to variants to define the data. See the example for details.
**Datatype:** array
**CObject:** NO

#### Example

```
# There are two variants to define the data

# Variant 1: Simple key / value
options {
 checkedin = Someone Checked in
 noonecheckedin = Noone Checked In
}

# Variant 2: Use integer keys to define option order and subKeys key / value to define the
 options
10 {
    key = 1
    value = LLL:EXT:my_ext/locallang.xml:theLabelOfTheOption
}

20 {
    ...
}
```

## Button

Button

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button

#### Description

Renders a filter with buttons.
**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.filter.button

#### Child elements

filterIdentifier, fieldIdentifier, label, partialPath, filterClassName, defaultValue, accessGroups, invert, invertable, displayFields, excludeFilters, showRowCount, submitOnChange, inactiveOption, inactiveValue, options,

## filterIdentifier

filterIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.filterIdent

#### Description

The unique identifier of this filter.
**Datatype:** String
**CObject:** NO

## fieldIdentifier

fieldIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.fieldIdent

#### Description

WARNING: Which one to choose??? Identifier of a defined data field. The filter affects on this data field. If more than on fieldIdentifier is given, the resulting query is linked with or. Either a single identifier of a defined field, a comma separated list of identifiers or '*' to use all defined fields.
**Datatype:** fieldIdentifier or *
**CObject:** NO

#### Example

```
// use a single field
fieldIdentifier = cn_short_local

// use a list of fields
fieldIdentifier = cn_short_local, name_local

// use every defined field
fieldIdentifier = *
```

## label

label

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.label

#### Description

The label wich is displayed beside the filter.
**Datatype:** String
**CObject:** NO

## partialPath

partialPath

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.partialPat

#### Description

**CObject:** NO

## filterClassName

filterClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.filterClass

#### Description

Name of the PHP Class of this filter.
**Datatype:** String

**CObject:** NO

#### Example

```
Tx_PtExtlist_Domain_Model_Filter_StringFilter
```

## defaultValue

defaultValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.defaultVa

#### Description

The default value which is shown or selected by default.
**Datatype:** String
**CObject:** NO

#### Example

```
// Selects key 3 as default Value
defaultValue = 3

// Selects multiple values in a multi value filter (3 and 18)
defaultValue {
 10 = 3
 20 = 18
}

// Selects current user as default in an user select list
defaultValue.cObject = TEXT
defaultValue.cObject.data = TSFE:fe_user|user|username
```

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.accessGr

#### Description

Comma separated list of user groups wich have access to this filter.
**Datatype:** Comma eparated list.
**Posible values:** TYPO3 Group Ids

**CObject:** NO

## invert

invert

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.invert

#### Description

Invert the constraint of this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## invertable

invertable

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.invertable

#### Description

Show a controle to invert this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## displayFields

displayFields

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.displayFi

#### Description

One or multiple identifiers of previously defined data fields.
**Datatype:** String / Comma separated list
**Posible values:** Previously defined field indentifier.

**Default:** If not set, the field defined in fieldIdentifier is used.
**CObject:** NO

## excludeFilters

excludeFilters

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.excludeF

#### Description

List of filters that are not considered for the group query of this filter.
**Datatype:** String / Comma separated list of filterIdentifiers.
**CObject:** NO

## showRowCount

showRowCount

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.showRow

#### Description

Show the rowcount if the select filter is filled with grouped data.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

## submitOnChange

submitOnChange

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.submitOn

#### Description

Instant submit filter if a value is selected.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## inactiveOption

inactiveOption

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.inactiveO

#### Description

Label of an option that is added to the select list. If this option is selected, the filter is inactive.
**Datatype:** String
**CObject:** NO

## inactiveValue

inactiveValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.inactiveV

#### Description

The submitted value for the inactiveOption described above.
**Datatype:** String
**CObject:** NO

## options

options

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Button.options

#### Description

Defines explicit options to display in a data filter. If options are set, no data is fetched from the database.
There are to variants to define the data. See the example for details.
**Datatype:** array
**CObject:** NO

#### Example

```
# There are two variants to define the data

# Variant 1: Simple key / value
```

```
options {
 checkedin = Someone Checked in
 noonecheckedin = Noone Checked In
}

# Variant 2: Use integer keys to define option order and subKeys key / value to define the
 options
10 {
    key = 1
    value = LLL:EXT:my_ext/locallang.xml:theLabelOfTheOption
}

20 {
    ...
}
```

## radiobutton

radiobutton

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton

#### Description

**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.filter.checkbox

#### Child elements

filterIdentifier, fieldIdentifier, label, partialPath, filterClassName, defaultValue, accessGroups, invert, invertable, displayFields, multiple, excludeFilters, showRowCount, submitOnChange, inactiveOption, inactiveValue, options,

## filterIdentifier

filterIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.filter

#### Description

The unique identifier of this filter.
**Datatype:** String
**CObject:** NO

## fieldIdentifier

fieldIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.fieldI

#### Description

WARNING: Which one to choose??? Identifier of a defined data field. The filter affects on this data field. If more than on fieldIdentifier is given, the resulting query is linked with or. Either a single identifier of a defined field, a comma separated list of identifiers or '*' to use all defined fields.
**Datatype:** fieldIdentifier or *
**CObject:** NO

#### Example

```
// use a single field
fieldIdentifier = cn_short_local

// use a list of fields
fieldIdentifier = cn_short_local, name_local

// use every defined field
fieldIdentifier = *
```

## label

label

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.label

#### Description

The label wich is displayed beside the filter.
**Datatype:** String
**CObject:** NO

## partialPath

partialPath

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.parti

#### Description

**CObject:** NO

## filterClassName

filterClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.filter

#### Description

Name of the PHP Class of this filter.
**Datatype:** String
**CObject:** NO

#### Example

```
Tx_PtExtlist_Domain_Model_Filter_StringFilter
```

## defaultValue

defaultValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.defa

#### Description

The default value which is shown or selected by default.
**Datatype:** String
**CObject:** NO

#### Example

```
// Selects key 3 as default Value
defaultValue = 3

// Selects multiple values in a multi value filter (3 and 18)
defaultValue {
 10 = 3
 20 = 18
}

// Selects current user as default in an user select list
defaultValue.cObject = TEXT
defaultValue.cObject.data = TSFE:fe_user|user|username
```

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.acce

#### Description

Comma separated list of user groups wich have access to this filter.
**Datatype:** Comma eparated list.
**Posible values:** TYPO3 Group Ids
**CObject:** NO

## invert

invert

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.inver

#### Description

Invert the constraint of this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## invertable

invertable

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.inver

#### Description

Show a controle to invert this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## displayFields

displayFields

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.displ

#### Description

One or multiple identifiers of previously defined data fields.
**Datatype:** String / Comma separated list
**Posible values:** Previously defined field indentifier.
**Default:** If not set, the field defined in fieldIdentifier is used.
**CObject:** NO

## multiple

multiple

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.multi

#### Description

**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## excludeFilters

excludeFilters

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.exclu

#### Description

List of filters that are not considered for the group query of this filter.
**Datatype:** String / Comma separated list of filterIdentifiers.
**CObject:** NO

## showRowCount

showRowCount

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.show

**Description**

Show the rowcount if the select filter is filled with grouped data.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

## submitOnChange

submitOnChange

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.subm

**Description**

Instant submit filter if a value is selected.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## inactiveOption

inactiveOption

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.inact

**Description**

Label of an option that is added to the select list. If this option is selected, the filter is inactive.
**Datatype:** String
**CObject:** NO

## inactiveValue

inactiveValue

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.inact

#### Description

The submitted value for the inactiveOption described above.
**Datatype:** String
**CObject:** NO

## options

options

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].radiobutton.optio

#### Description

Defines explicit options to display in a data filter. If options are set, no data is fetched from the database. There are to variants to define the data. See the example for details.
**Datatype:** array
**CObject:** NO

#### Example

```
# There are two variants to define the data

# Variant 1: Simple key / value
options {
 checkedin = Someone Checked in
 noonecheckedin = Noone Checked In
}

# Variant 2: Use integer keys to define option order and subKeys key / value to define the
 options
10 {
    key = 1
    value = LLL:EXT:my_ext/locallang.xml:theLabelOfTheOption
}

20 {
    ...
}
```

## firstletter

firstletter

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter

#### Description

**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.filter.firstletter

#### Child elements

filterIdentifier, fieldIdentifier, label, partialPath, filterClassName, defaultValue, accessGroups, invert, invertable, displayFields, excludeFilters, showRowCount, inactiveOption, inactiveValue,

## filterIdentifier

filterIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.filterIde

#### Description

The unique identifier of this filter.
**Datatype:** String
**CObject:** NO

## fieldIdentifier

fieldIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.fieldIde

#### Description

WARNING: Which one to choose??? Identifier of a defined data field. The filter affects on this data field. If more than on fieldIdentifier is given, the resulting query is linked with or. Either a single identifier of a defined field, a comma separated list of identifiers or '*' to use all defined fields.
**Datatype:** fieldIdentifier or *
**CObject:** NO

#### Example

```
// use a single field
fieldIdentifier = cn_short_local

// use a list of fields
fieldIdentifier = cn_short_local, name_local

// use every defined field
fieldIdentifier = *
```

## label

label

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.label

#### Description

The label wich is displayed beside the filter.
**Datatype:** String
**CObject:** NO

## partialPath

partialPath

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.partialP

#### Description

**CObject:** NO

## filterClassName

filterClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.filterCla

#### Description

Name of the PHP Class of this filter.
**Datatype:** String
**CObject:** NO

#### Example

```
Tx_PtExtlist_Domain_Model_Filter_StringFilter
```

## defaultValue

defaultValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.default

#### Description

The default value which is shown or selected by default.
**Datatype:** String
**CObject:** NO

#### Example

```
// Selects key 3 as default Value
defaultValue = 3

// Selects multiple values in a multi value filter (3 and 18)
defaultValue {
 10 = 3
 20 = 18
}

// Selects current user as default in an user select list
defaultValue.cObject = TEXT
defaultValue.cObject.data = TSFE:fe_user|user|username
```

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.access

#### Description

Comma separated list of user groups wich have access to this filter.
**Datatype:** Comma eparated list.
**Posible values:** TYPO3 Group Ids
**CObject:** NO

## invert

invert

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.invert

#### Description

Invert the constraint of this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## invertable

invertable

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.invertab

#### Description

Show a controle to invert this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## displayFields

displayFields

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.display

#### Description

One or multiple identifiers of previously defined data fields.
**Datatype:** String / Comma separated list
**Posible values:** Previously defined field indentifier.
**Default:** If not set, the field defined in fieldIdentifier is used.
**CObject:** NO

## excludeFilters

excludeFilters

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.exclude

#### Description

List of filters that are not considered for the group query of this filter.
**Datatype:** String / Comma separated list of filterIdentifiers.
**CObject:** NO

## showRowCount

showRowCount

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.showR

#### Description

Show the rowcount if the select filter is filled with grouped data.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 1
**CObject:** NO

## inactiveOption

inactiveOption

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.inactive

#### Description

Label of an option that is added to the select list. If this option is selected, the filter is inactive.
**Datatype:** String
**CObject:** NO

## inactiveValue

inactiveValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].firstletter.inactive

#### Description

The submitted value for the inactiveOption described above.
**Datatype:** String
**CObject:** NO

## tagcloud

tagcloud

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud

#### Description

**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.filter.tagcloud

#### Child elements

filterIdentifier, fieldIdentifier, label, partialPath, filterClassName, defaultValue, accessGroups, invert, invertable, displayFields, excludeFilters, inactiveOption, inactiveValue, maxItems, minColor, maxColor, minSize, maxSize,

## filterIdentifier

filterIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.filterIde

#### Description

The unique identifier of this filter.
**Datatype:** String
**CObject:** NO

## fieldIdentifier

fieldIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.fieldIde

#### Description

WARNING: Which one to choose??? Identifier of a defined data field. The filter affects on this data field. If more than on fieldIdentifier is given, the resulting query is linked with or. Either a single identifier of a defined field, a comma separated list of identifiers or '*' to use all defined fields.
**Datatype:** fieldIdentifier or *
**CObject:** NO

**Example**

```
// use a single field
fieldIdentifier = cn_short_local

// use a list of fields
fieldIdentifier = cn_short_local, name_local

// use every defined field
fieldIdentifier = *
```

## label

label

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.label

**Description**

The label wich is displayed beside the filter.
**Datatype:** String
**CObject:** NO

## partialPath

partialPath

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.partialP

**Description**

**CObject:** NO

## filterClassName

filterClassName

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.filterCla

**Description**

Name of the PHP Class of this filter.
**Datatype:** String

**CObject:** NO

#### Example

```
Tx_PtExtlist_Domain_Model_Filter_StringFilter
```

## defaultValue

defaultValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.defaultV

#### Description

The default value which is shown or selected by default.
**Datatype:** String
**CObject:** NO

#### Example

```
// Selects key 3 as default Value
defaultValue = 3

// Selects multiple values in a multi value filter (3 and 18)
defaultValue {
 10 = 3
 20 = 18
}

// Selects current user as default in an user select list
defaultValue.cObject = TEXT
defaultValue.cObject.data = TSFE:fe_user|user|username
```

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.accessG

#### Description

Comma separated list of user groups wich have access to this filter.
**Datatype:** Comma eparated list.
**Posible values:** TYPO3 Group Ids
**CObject:** NO

## invert

invert

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.invert

#### Description

Invert the constraint of this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## invertable

invertable

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.invertab

#### Description

Show a controle to invert this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## displayFields

displayFields

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.displayF

#### Description

One or multiple identifiers of previously defined data fields.
**Datatype:** String / Comma separated list
**Posible values:** Previously defined field indentifier.
**Default:** If not set, the field defined in fieldIdentifier is used.
**CObject:** NO

## excludeFilters

excludeFilters

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.exclude

**Description**

List of filters that are not considered for the group query of this filter.
**Datatype:** String / Comma separated list of filterIdentifiers.
**CObject:** NO

## inactiveOption

inactiveOption

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.inactive

**Description**

Label of an option that is added to the select list. If this option is selected, the filter is inactive.
**Datatype:** String
**CObject:** NO

## inactiveValue

inactiveValue

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.inactive

**Description**

The submitted value for the inactiveOption described above.
**Datatype:** String
**CObject:** NO

## maxItems

maxItems

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.maxIter

#### Description

Maximum of items tos show.
**Datatype:** Int
**CObject:** NO

## minColor

minColor

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.minCol

#### Description

Minimum Color
**Datatype:** Int
**CObject:** NO

#### Example

```
minColor = #1c448d
```

## maxColor

maxColor

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.maxCo

#### Description

Maximum Color
**Datatype:** Int
**CObject:** NO

#### Example

```
maxColor = #b8cffa
```

## minSize

minSize

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.minSize

#### Description

Minimum size of tags in pixel.
**Datatype:** Int
**CObject:** NO

## maxSize

maxSize

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].tagcloud.maxSiz

#### Description

Maximum size of tags in pixel
**Datatype:** Int
**CObject:** NO

## Proxy

Proxy

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Proxy

#### Description

A proxy filter is not displayed. It uses a filter from an other list defined by the proxy path and sets the result constraint to the given field.
**CObject:** NO
**Prototype:** plugin.tx_ptextlist.prototype.filter.proxy

#### Child elements

proxyPath,

## proxyPath

proxyPath

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].Proxy.proxyPath

#### Description

Path to a filter from an other list. The path has the format [listId].[filterBoxId].[filterId]

**Datatype:** String
**CObject:** NO

#### Child elements

filterIdentifier, defaultValue, label, accessGroups, filterClassName, partialPath, invert, filter.breadCrumbString, size, maxLength,

## filterIdentifier

filterIdentifier

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].filterIdentifier

#### Description

The unique identifier of this filter.
**Datatype:** String
**CObject:** NO

## defaultValue

defaultValue

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].defaultValue

#### Description

The default value which is shown or selected by default.
**Datatype:** String
**CObject:** NO

#### Example

```
// Selects key 3 as default Value
defaultValue = 3

// Selects multiple values in a multi value filter (3 and 18)
defaultValue {
 10 = 3
 20 = 18
}

// Selects current user as default in an user select list
defaultValue.cObject = TEXT
defaultValue.cObject.data = TSFE:fe_user|user|username
```

## label

label

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].label

#### Description

The label wich is displayed beside the filter.
**Datatype:** String
**CObject:** NO

## accessGroups

accessGroups

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].accessGroups

#### Description

Comma separated list of user groups wich have access to this filter.
**Datatype:** Comma eparated list.
**Posible values:** TYPO3 Group Ids
**CObject:** NO

## filterClassName

filterClassName

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].filterClassName

#### Description

Name of the PHP Class of this filter.
**Datatype:** String
**CObject:** NO

#### Example

```
Tx_PtExtlist_Domain_Model_Filter_StringFilter
```

## partialPath

partialPath

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].partialPath

**Description**

**CObject:** NO

## invert

invert

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].invert

**Description**

Invert the constraint of this filter.
**Datatype:** Boolean
**Posible values:** 0,1
**Default:** 0
**CObject:** NO

## size

size

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].size

**Description**

Define the size of the string field.
**Datatype:** int
**CObject:** NO

## maxLength

maxLength

**TS-Key:**

plugin.tx_ptextlist.settings.listConfig.[yourListId].filters.[yourFilterBoxId].filterConfigs.[yourNumericFilterId].maxLength

#### Description

Define the maximum length of the string field.
**Datatype:** int
**CObject:** NO

## headerPartial

headerPartial

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].headerPartial

#### Description

Path to the header partial.
**Datatype:** String
**CObject:** NO

#### Example

```
// Fluid style. Relative to pt_extlist/Resources/Private/Partials/
// points to pt_extlist/Resources/Private/Partials/List/SpecialPartial.html
headerPartial = List/SpecialPartial

// Extension relative path
headerPartial = EXT:pt_extlist_special/Resources/Private/Partials/SpecialPartial.html
```

## bodyPartial

bodyPartial

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].bodyPartial

#### Description

Path to the body partial.
**Datatype:** String
**CObject:** NO

#### Example

```
// Fluid style. Relative to pt_extlist/Resources/Private/Partials/
// points to pt_extlist/Resources/Private/Partials/List/SpecialPartial.html
bodyPartial = List/SpecialPartial
```

```
// Extension relative path
bodyPartial = EXT:pt_extlist_special/Resources/Private/Partials/SpecialPartial.html
```

## agregateRowsPartial

agregateRowsPartial

#### TS-Key:

plugin.tx_ptextlist.settings.listConfig.[yourListId].agregateRowsPartial

#### Description

Path to the aggregate row partial.
**Datatype:** String
**CObject:** NO

#### Example

```
// Fluid style. Relative to pt_extlist/Resources/Private/Partials/
// points to pt_extlist/Resources/Private/Partials/List/SpecialPartial.html
agregateRowsPartial = List/SpecialPartial

// Extension relative path
agregateRowsPartial = EXT:pt_extlist_special/Resources/Private/Partials/SpecialPartial.html
```

# Chapter 7. Developer Guide

## 1. Extending pt_extlist

This chapter is mainly for developing and extending pt_extlist.

### 1.1. RenderChain

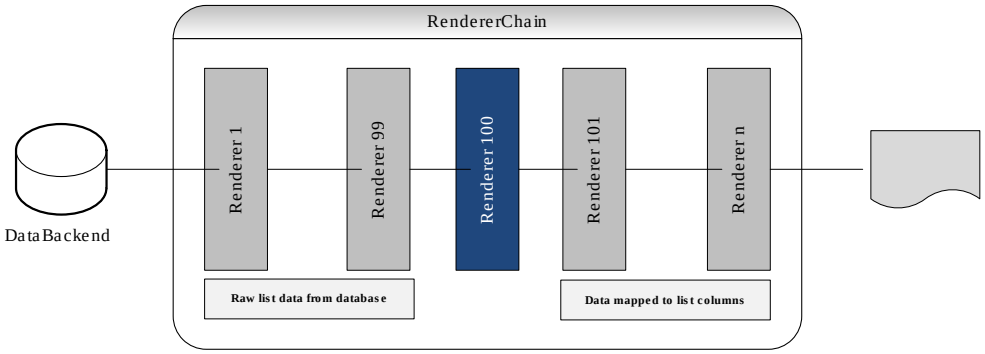


**Figure 7.1. RenderChain**