Base Distribution - Story # 8643

Status:	Resolved	**Priority:**	Should have
Author:	Daniel Poetzinger	**Category:**	
Created:	2010-07-04	**Assigned To:**	
Updated:	2011-05-05	**Due date:**	
Subject:	As a Developer I want a Plugin API Concept		

Description

- Documentation of API
- Dummy Controller Example

Related issues:

related to TYPO3.Fluid - Major Feature # 8773: Implement support for Widgets	**Resolved**	**2010-07-09**

History

#1 - 2010-07-09 13:50 - Robert Lemke

- Project changed from 1204 to Core Team

#2 - 2010-07-09 13:51 - Robert Lemke

- Target version set to 593

#3 - 2010-07-28 19:08 - Sebastian Kurfuerst

http://ietherpad.com/O1ilAI68qc

Widget / Plugin API Ideas

http://forge.typo3.org/issues/8643 (main issue)
http://forge.typo3.org/issues/8773 (widget API)
http://forge.typo3.org/issues/8774 (query->execute Proxy)

GLOSSARY
PACKAGE
... a FLOW3 Package

Page
... a collection of content

Content Aggregate Root (short: CAR) (name to be discussed)
... a content object which has an externally visible identity (URI). Under this URI, a rendered representation of this content object can be found.
... somehow is the generalization of the "Page" concept in TYPO3 v4
Examples: "Page" is instance of "Content Aggregate Root", but also "News" could be instance of "Content Aggregate Root"

PLUGIN
... a selection of controllers & actions which can be inserted to a PAGE like a normal content element, which is configured through TypoScript (or YAML?).
... NEVER a Content Aggregate Root (as it lives INSIDE a page)
... configurable (context specific)
... needs Interaction (?)
Examples: see below

WIDGET
... a specialized ViewHelper that calls one/a selection of controllers; can be inserted into a FLUID TEMPLATE ("Plugin inside a Template")

The domain model of a package ...
... should be usable as "Content"
... can be made translateable, or even attached to StructureNodes (and can have multiple versions ?!)

EXAMPLES of TypoScript-Driven Model-Based output

* Single news article which should be fully rendered on a page (v4: "Insert Records")
-> Domain Model from News-Package
-> "drag-drop" -- News-Article will be attached to a Structure Node
-> rendering will be determined via TypoScript / Fluid "News" TS Object
-> News Package provides "News TS Object" (which is empty and inherits from AbstractContentObject)

```
// Default News TS Object Configuration (this is implicit)
prototype<News> {
   template.file = /the/default/template.html
}
```

* On the homepage a teaser for the above news article should be rendered
-> Same as above but the TS Object Fluid template has to be replaced

```
// overridden News TS Object Configuration
page.sections.teasers prototype<News> {
   template.file = /my/teaser/template.html
}
```

* Now I want to display a list of News articles
-> Generic List object (e.g. List<News>)
-> Same as above, but this time the editor inserts a List<News>
? what should happen internally here -- on the level of structure nodes
-> Then some kind of configuration interface appears (sort order, max items, constraints..)
? How do you specify the default sort order -> see above, creates a TS configuration (that might be attached to the current Structure node??)

* Now I want to click on an item of the above list and see the detail view of the corresponding article
-> we mark the "News" DO as "Content Aggregate Root", which means it gets an externally visible identity (i.e. an URI), and can be rendered on a separate "Web-Page". This "web-page" does not exist in the TYPO3 page tree as "Page".
-> we render the StructureNode where the "News" DO is attached to.
? how can we change content and layout of surrounding elements? Example: a static but editable text
-> use a (generic) "single view" plugin which can display (arbitary) "content aggregate roots". The "single view" plugin has no URI (i.e. it is NO "content aggregate root"), but belongs to the Page DO where it is attached to (and this in turn has a URI as it is a "Content Aggregate Root").
? a plugin is never a CAR

* VIRTUAL STRUCTURE NODES / List-Domain Object
-> like "Saved Search" or "Virtual Folder" in mail program

EXAMPLES OF PLUGINS

* Forms
* User Registration
* Lost Password
* Multi-File uploader

EXAMPLES OF WIDGETS

* Every widget will get a UUID (at first call)

* some data (like Widget arguments) can be registered as Widget Configuration

--> Widget Configuration is Persistent and will be serialized transparently and stored in user session (belonging to UUID of widget)

--> f.e. in AJAX requests, you only have access to Widget Configuration

* Page browser of objects

Example:

```
<f:paginate objects="{blogs}" as="paginatedBlogs">
  ... do something with {paginatedBlogs} ...
</f:paginate>
```

```
<!-- START UNFINISHED -->
class \F3\Fluid\ViewHelpers\Widget\Paginate extends .....\AbstractWidget (extends AbstractViewHelper) {
  $controllersAndActions = ...
  public function render($objects, $as) {
    $as = \SomeService::modifyQueryForObjects($objects);
    $this->templateVariableContainer->add($as)..
    $this->addWidgetVariable(
    $content = $this->view->render(); // UGLY!!
    $content .= $this->renderChildren();
  }
}
```

Internally: call F3\Fluid\Widgets\Paginate\Controller\PaginateController extends ...AbstractWidgetController

```
public function index($currentPage = 0) {
  $objects = $this->settings['objects'];
  $as = $this->settings['as'];

  $query = $this->persistenceManager->getQueryForResultSet($objects);
  // calculate how many pages are available
  $numberOfPages = $query->count() / $objectsPerPage;

  // modify query (merge existing start/limit)
  $as = $query->execute(); // again returns a lazy proxy
  $this->templateVariableContainer->add(...); // UGLY!!!!
  $output = ... build HTML for pagination selector ...
  $output = $this->renderChildren();
  return $output;
}

<!-- STOP UNFINISHED -->
```

* Filter of objects

* Sorter

* Google Maps

* Ajax search / autocompletion

Example:

```
<f:form.textfield.autocomplete property="title" objectsToSearch="{blogs}" />
```

-> render(): called from widget (by default)

```
F3\Fluid\....\AutocompletionWidget extends ...AbstractWidget {
    public function render($property, $objectsToSearch) {
        $this->tag->.....
        return $this->tag->render();
    }
     public function persistArguments() {
        $this->persistArgument('query', $this->persistenceManager->getQueryForResultSet($this->arguments['objectsToSearch'])); // will be saved in session scope
        // TODO: discuss names!
    }
}

class \...\SomeController extends ...ActionController {
    $supportedRequestTypes = array('...\AjaxWidgetRequest');
    public function searchAction($searchWord) {
        $query = $this->modifyQuery($this->settings['query'], $searchWord);
        $query->execute();
        // format as JSON and return
    }
}
```

-> we need a new RequestHandler that restores widget configuration from session (which is stored under the unique ID of the widget)

TODOs

TODO: we need a constraint system ("News is only allowed inside a Page DO")

#4 - 2010-08-20 12:19 - Robert Lemke

- Status changed from New to Resolved

- % Done changed from 0 to 100

The rough concept is done - enough for this sprint.
Next step is to refine it and create a prototype.

#5 - 2011-05-05 13:49 - Robert Lemke

- Project changed from Core Team to Base Distribution

- Target version deleted (593)