mearra

Stuff Drupal with Feeds and custom feeds plugins

Developer Days Barcelona 2012

Mikael Kundert
Joonas Meriläinen

mearra

Who are we?

- Mikael Kundert
  - Doing web stuff from age 13!
  - Working with Drupal from 2009
- Joonas Meriläinen
  - Studied information technology, mostly web technologies, at the Tampere university of technology
  - 4+ years of experience with Drupal, working in a University, as a freelancer and now as a developer at Mearra

Mearra

- Founded in 2010 by four Drupal dudes (Vesa, Juha, Joonas Kiminki, Tomi)
- 100% Drupal and open source
- Offices in Finland, Latvia and Estonia
- Market area: Europe
- We’re hiring!

Contrib modules

- AddThis Button
- Book chapters
- Bookmark Organizer
- Commerce Extra
- Commerce Stripe
- Domain Notification
- Entity Sync
- External HTTP authentication
- Growl
- Maintenance
- Menu Admin Splitter
- MRBS
- NorthID Online ID
- OG Bookmarks
- Poll Enhancements
- Poll Improved
- PROG Gallery
- Radioactivity
- Redirecting Click Bouncer
- SendGrid Integration
- Samba Explorer
- SendGrid Integration
- Snoobi web analytics
- TUPAS Authentication
- Views Menu Area
- Web Of Trust integration
- Webform Invites
- Webform Mass Email
- + 6 Finland specific modules

What is Feeds?

- For **importing** or **aggregating** data to Drupal
- Successor module of **FeedAPI** and **Feed Element Mapper**

How Feeds works?

- Requires CTools
- Plugins are written using OOP
- Each importer needs three components: **Fetcher**, **Parser**, **Processor**

Good contributed modules!

Module	Source format	D.o	Version
Feeds XPath parser	XML	feeds_xpathparser	7.x-1.0-beta3
Feeds QueryPath Parser	XML	feeds_querypath_parser	7.x-1.0-beta1
Feeds JSONPath Parser	JSON	feeds_jsonpath_parser	7.x-1.0-beta2
Feeds Excel	Excel	feeds_excel	7.x-1.0-beta1
Feeds: YouTube parser	XML	feeds_youtube	7.x-2.0-beta1
Feeds: Facebook parser	JSON	feeds_facebook	7.x-1.x-dev

How Feeds works?

FeedsPlugin

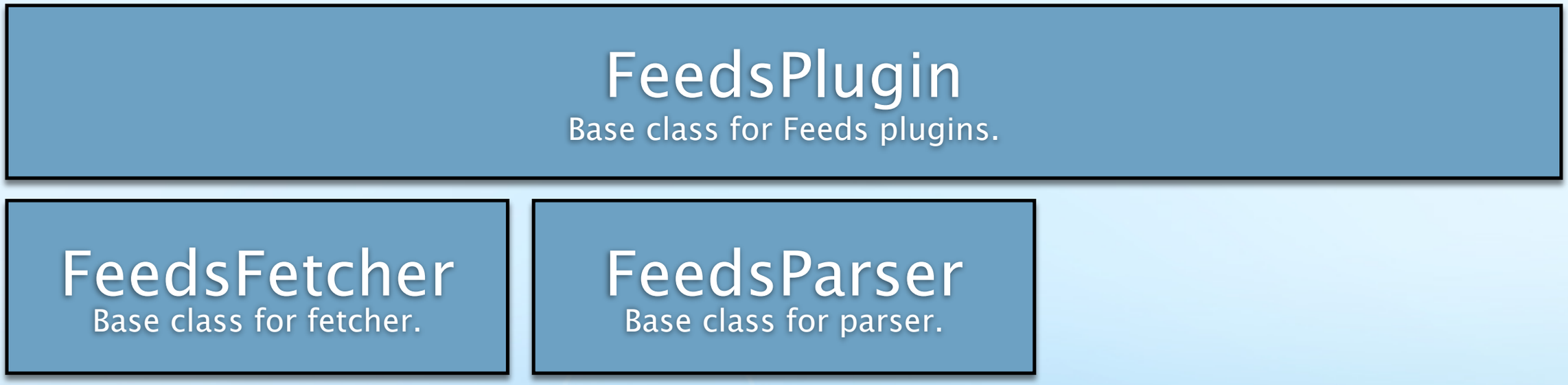
Base class for Feeds plugins.

How Feeds works?

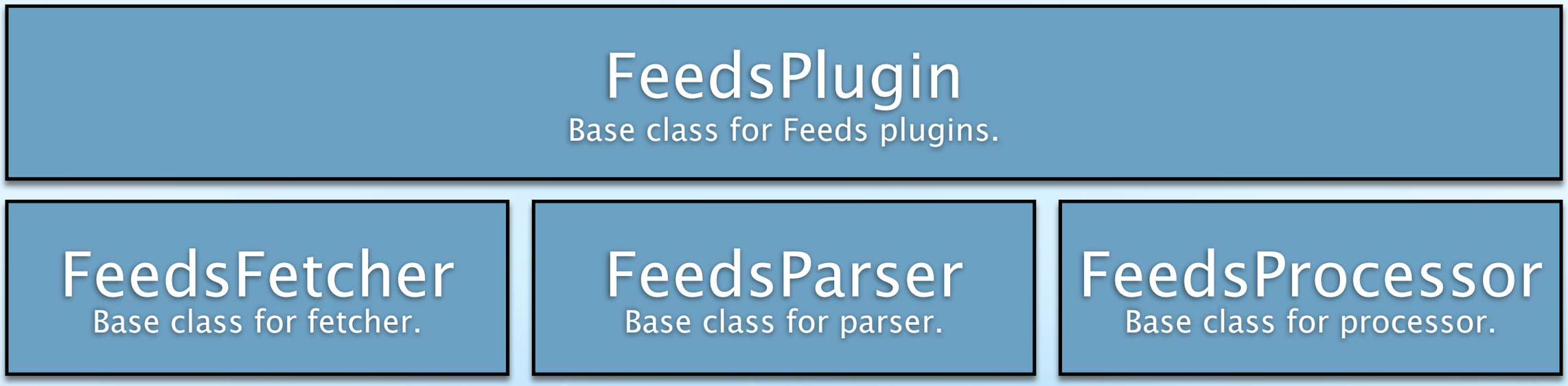
FeedsPlugin
Base class for Feeds plugins.

FeedsFetcher
Base class for fetcher.

How Feeds works?

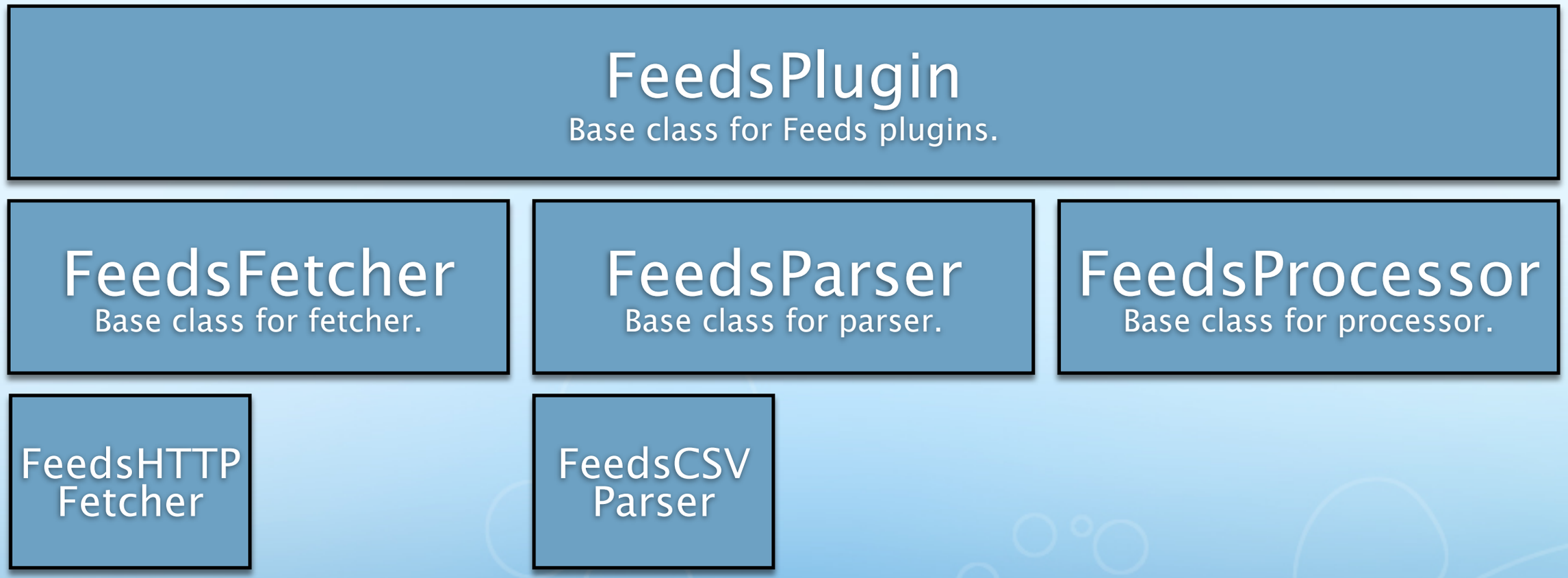


How Feeds works?

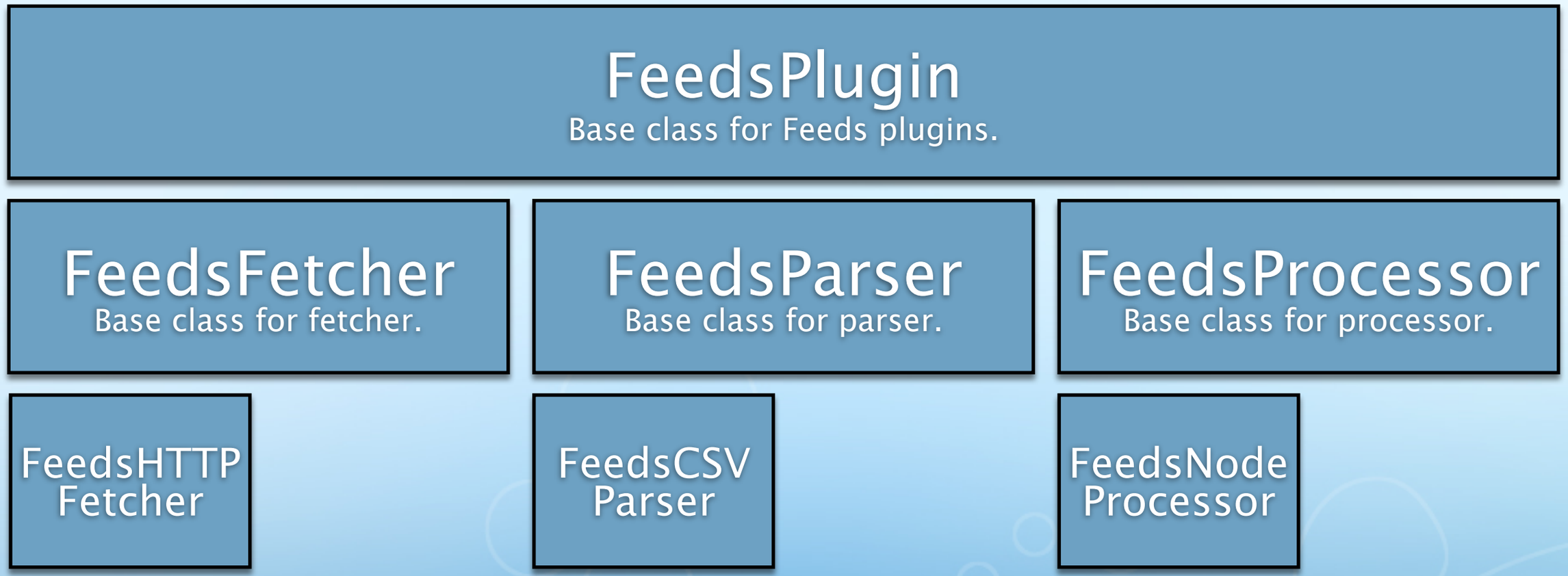


How Feeds works?

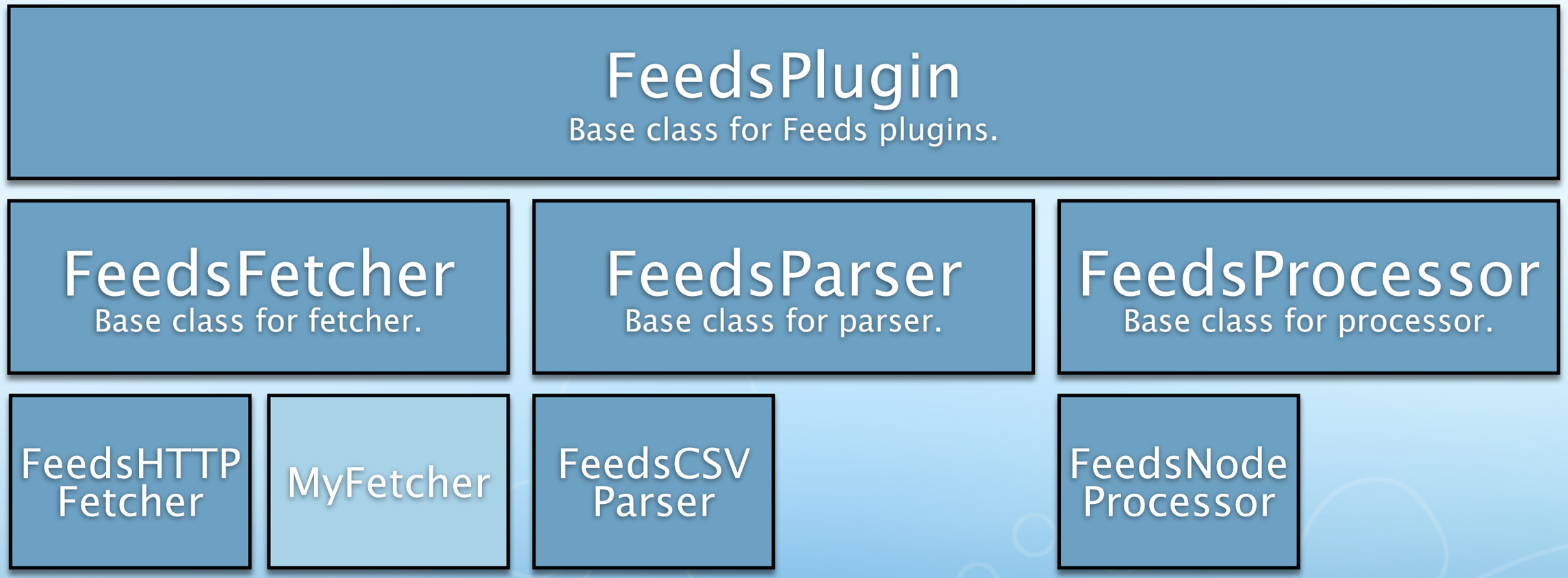
How Feeds works?



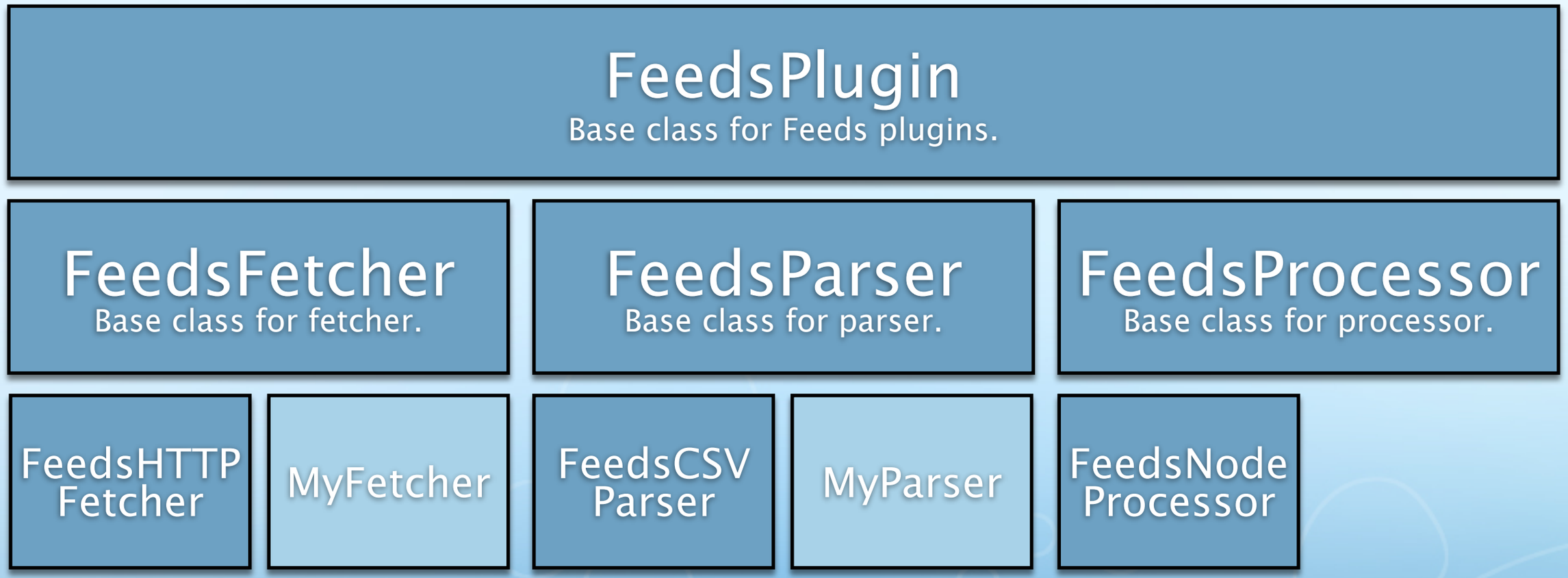
How Feeds works?



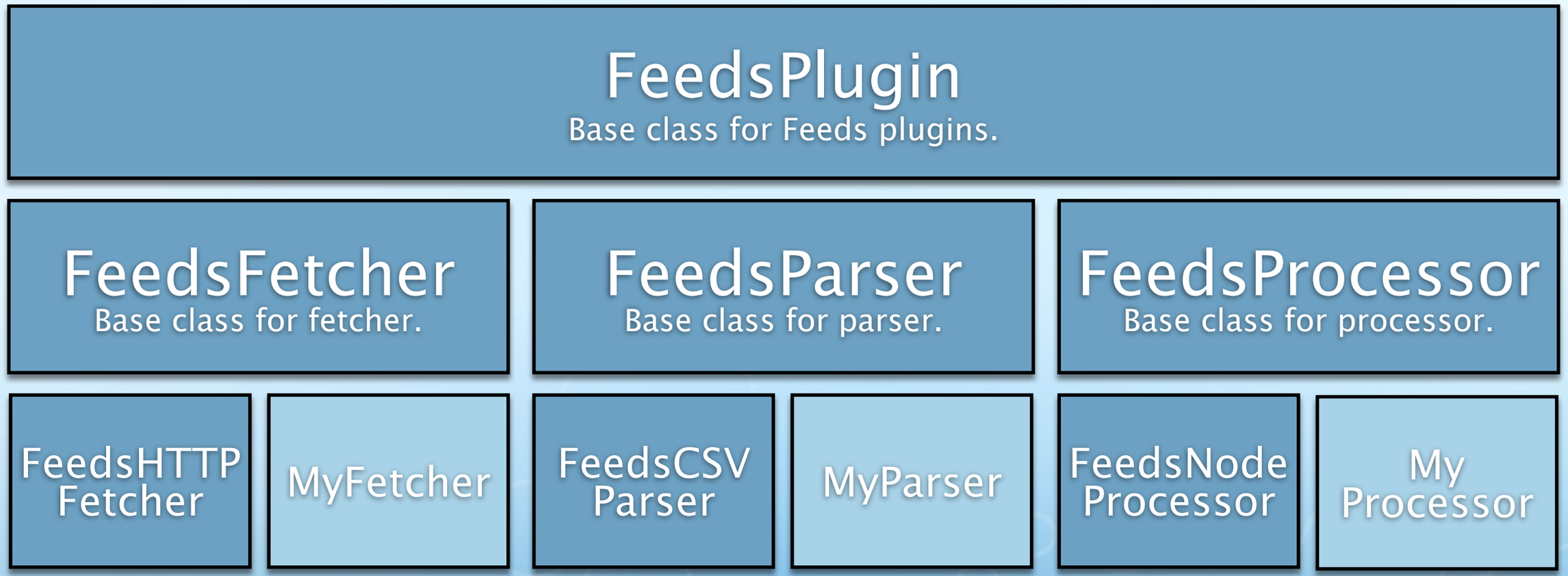
How Feeds works?



How Feeds works?



How Feeds works?



How Feeds works?

FeedsPlugin
Base class for Feeds plugins.

FeedsFetcher
Base class for fetcher.

FeedsParser
Base class for parser.

FeedsProcessor
Base class for processor.

FeedsHTTP Fetcher

MyFetcher

FeedsCSV Parser

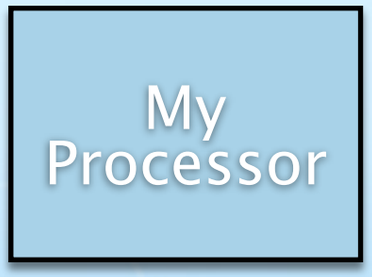
MyParser

FeedsNode Processor

My Processor

+ Mappers

How Feeds works?



Implementing a processor

- Tell Feeds that you have plugins available (CTools)
- Describe your plugins
- Implement the plugin

Implementing a processor

```php
/**
 * File: my_processor.module
 * Implements hook_ctools_plugin_api()
 */
function my_processor_ctools_plugin_api($module = '', $api = '') {
  if ($module == "feeds" && $api == "plugins") {
    return array("version" => 1);
  }
}
```

Implementing a processor

```php
/**
 * File: my_processor.module
 * Implements hook_feeds_plugins().
 */
function my_processor_feeds_plugins() {
  return array(
    'MyProcessor' => array(
      'name' => 'My custom processor',
      'description' => 'This is the description of my own processor.',
      'handler' => array(
        'parent' => 'FeedsProcessor',
        'class' => 'MyProcessor',
        'file' => 'MyProcessor.inc',
        'path' => drupal_get_path('module', 'my_processor'),
      ),
    ),
  );
}
```

Implementing a processor

Implementing a processor

Implementing a processor

```
/**
 * File: MyProcessor.inc
 */
class MyProcessor extends FeedsProcessor {

  /**
   * Required methods:
   *   - entityType()
   *   - newEntity(FeedsSource $source)
   *   - entityLoad(FeedsSource $source, $entity_id)
   *   - entitySave($entity)
   *   - entityDeleteMultiple($entity_ids)
   */

}
```

Implementing a processor

```php
/**
 * File: MyProcessor.inc
 * Class: MyProcessor
 */
public function entityType() {
  return 'my_entity';
}
```

```php
public function newEntity(FeedsSource $source) {
  $my_entity = new stdClass();
  $my_entity->id = 0;
  $my_entity->title = '';
  $my_entity->description = '';
  return $my_entity;
}
```

Implementing a processor

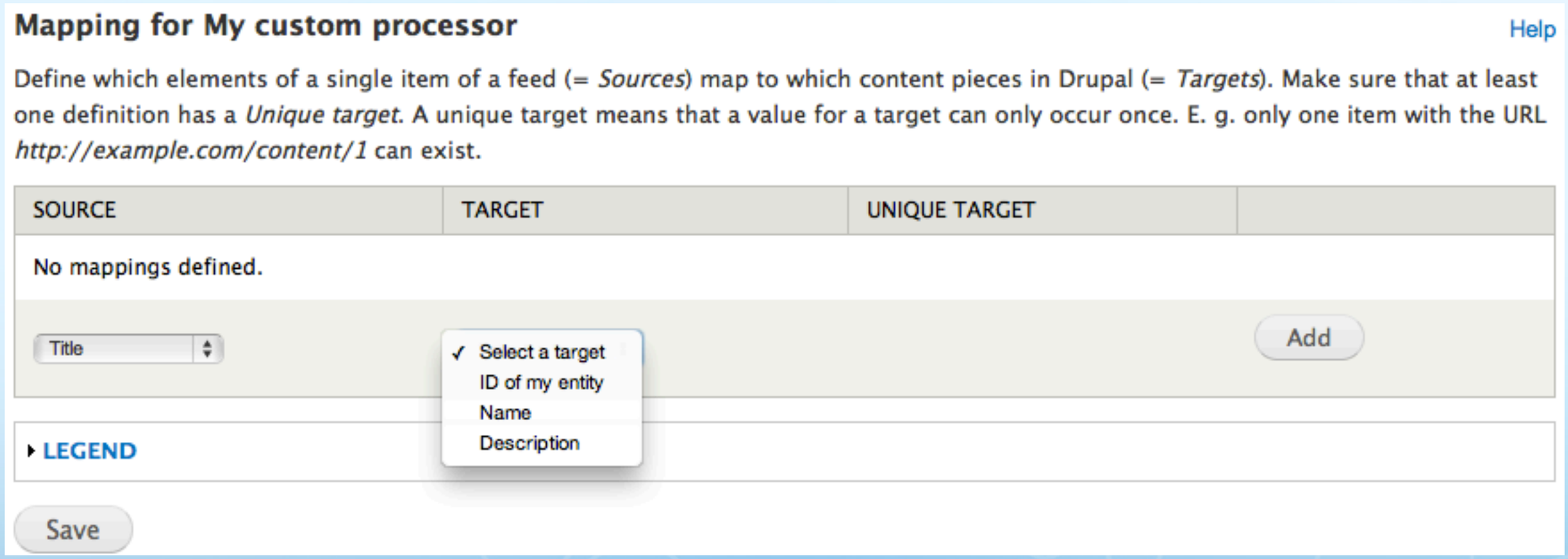
```php
/**
 * File: MyProcessor.inc
 * Class: MyProcessor
 */
public function entityLoad(FeedsSource $source, $entity_id) {
  return my_entity_load($entity_id);
}

public function entitySave($entity) {
  my_entity_save($entity);
}

public function entityDeleteMultiple($entity_ids) {
  my_entity_delete_multiple($entity_ids);
}
```

Implementing a processor

```
/**
 * File: MyProcessor.inc
 * Class: MyProcessor
 */
public function getMappingTargets() {
  return array(
    'id' => array(
      'name' => t('ID of my entity'),
      'description' => t('This ID is unique identifier for my entity.'),
      'optional_unique' => TRUE,
    ),
    'title' => array(
      'name' => t('Name'),
      'description' => t('Name of the entity item.'),
    ),
    'description' => array(
      'name' => t('Description'),
      'description' => t('Description of the entity item.'),
    ),
  );
}
```

Implementing a processor

Implementing a processor



Implementing a processor

- Other methods you probably might use:
  - setTargetElement()
  - configForm(), configDefaults(), configFormValidate(), configFormSubmit()

Extending existing processor

```
class MyOverrideNodeProcessor extends FeedsNodeProcessor {

  public function process(FeedsSource $source, FeedsParserResult $parser_result) {
    // ... snipped ...
    while ($item = $parser_result->shiftItem()) {
      $nid = $this->existingEntityId($source, $parser_result);
      $skip_nids = explode(" ", $this->config['skip_nids']);
      if (in_array($nid, $skip_nids)) {
        continue;
      }
    }
    // ... snipped ...
  }

}
```

Feeds add-ons/plugins

- Extend Feeds in many different ways to get data in from external sources in a certain format (parsers)
  - XPath, QueryPath, JSON, XSLT, REGEX, KML, iCal, Excel...
  - YouTube, Vimeo, Flickr, Slideshare, Sharepoint, Salesforce...

Source: http://drupal.org/node/856644

mearra

XPath/QueryPath Parser

- Very helpful with complex XML-files providing a generic solution
- Both have their own syntax for choosing XML-elements
  - XPath: **//p[@id="images"]/following-sibling::p**
  - QueryPath: **articlepart:has (article_part_type_id#1) data body**

mearra

XPath/QueryPath Parser

- XPath: //p[@id="images"]/following-sibling::p
- Will select the caption from this mess where the wanted p-element doesn't have any id:

```
<p id='source'>News agency</p>
<p id='images'><b>Image:</b></p>
<p>Image caption text</p>
```

mearra

XPath/QueryPath Parser

QueryPath: **articlepart:has**
(article_part_type_id#1) data body

Will select the body from this:

```
<articlepart id="272333" refType="ArticlePart">
  <article_id id="201341"/>
  <article_part_type_id id="1">Text</article_part_type_id>
  <data>
    <npdoc xmlns="http://www.infomaker.se/npdoc/2.1" version="2.1"
xml:lang="fi">
      <headline>Title comes here</headline>
      <body>Bodytext with lots of stuff...</body>
```

mearra

Feeds Tamper

From feeds_tamper.module (which is only 134 lines):

```
/**
 * Implements hook_feeds_after_parse().
 *
 *This is the meat of the whole deal. After every Feeds run,
   before going into
 * processing, this gets called and modifies the data based on
   the configuration.
 */
```

In short, it provides a plugin architecture for Feeds to manipulate data before it gets saved.

mearra

Feeds Tamper

Plenty of different plugins available

Keyword filter
Required field
HTML entity decode
HTML entity encode
Make URLs absolute
Strip tags
List
Explode
Implode

Number
Format a number
Calculate hash
Copy source value
Country to ISO code
Rewrite
Set default value

Convert case
Convert to boolean
Find replace
Find replace REGEX
Pad a string
String to Unix timestamp
Trim
Truncate

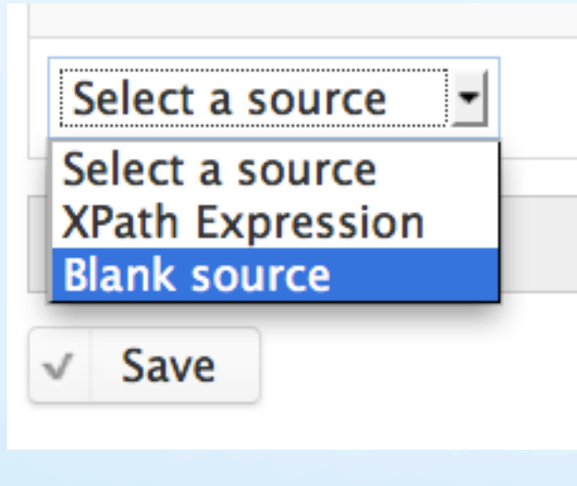
mearra

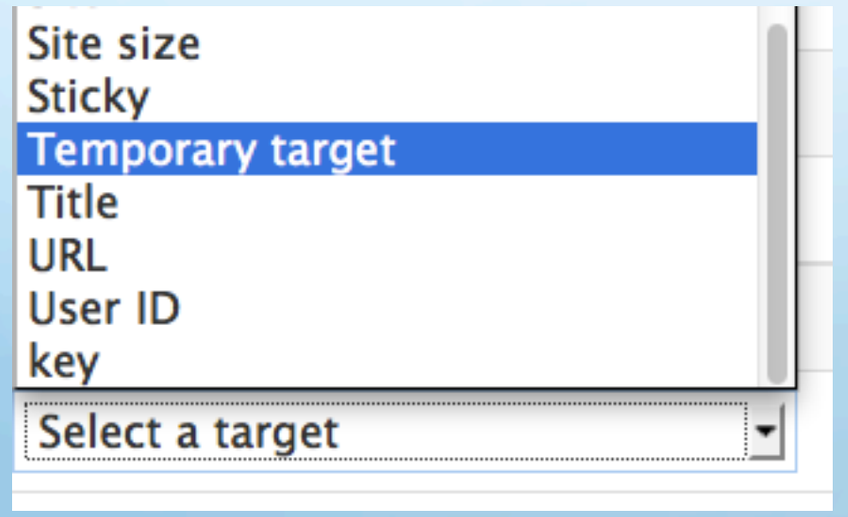
Feeds Tamper

Provides two additional mapping elements to Feeds UI

Blank Source
Useful for setting a default value or collecting values from several elements

Temporary target
Can be used to store values for processing before mapping to a field





mearra

Feeds Tamper

Writing a custom plugin is very easy

Add the following to your_module.module:

```
/**
* Implements hook_ctools_plugin_directory().
*/
function hook_ctools_plugin_directory($module, $plugin) {
  if ($module == 'feeds_tamper') {
    return 'plugins';
  }
}
```

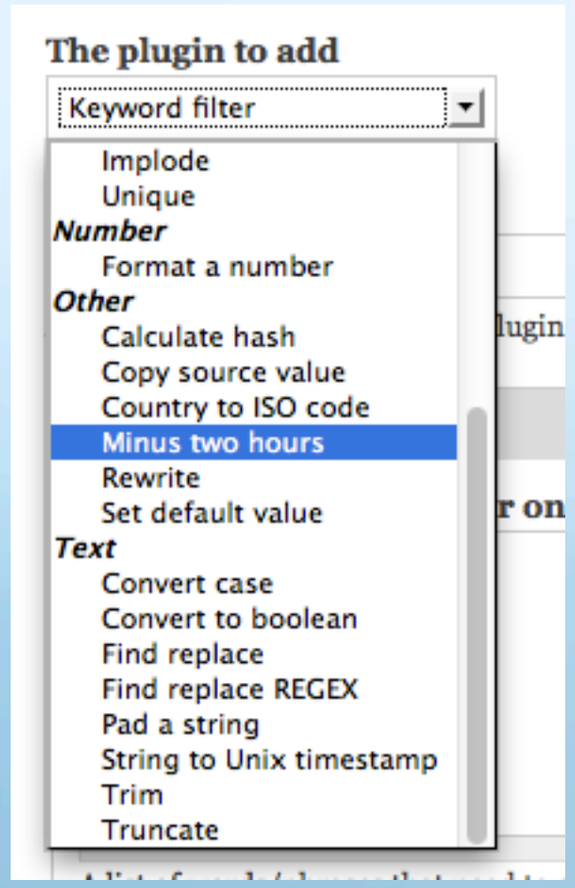
Then add your_plugin.inc to plugins directory inside your module's folder.

mearra

Feeds Tamper

```php
$plugin = array(
  'form' => 'ttt_site_minus_two_hours_form',
  'callback' => 'ttt_site_minus_two_hours_callback',
  'name' => 'Minus two hours',
  'multi' => 'loop',
  'category' => 'Other',
);
```

`$plugin` variable in your_plugin.inc will tell Feeds Tamper about your plugin

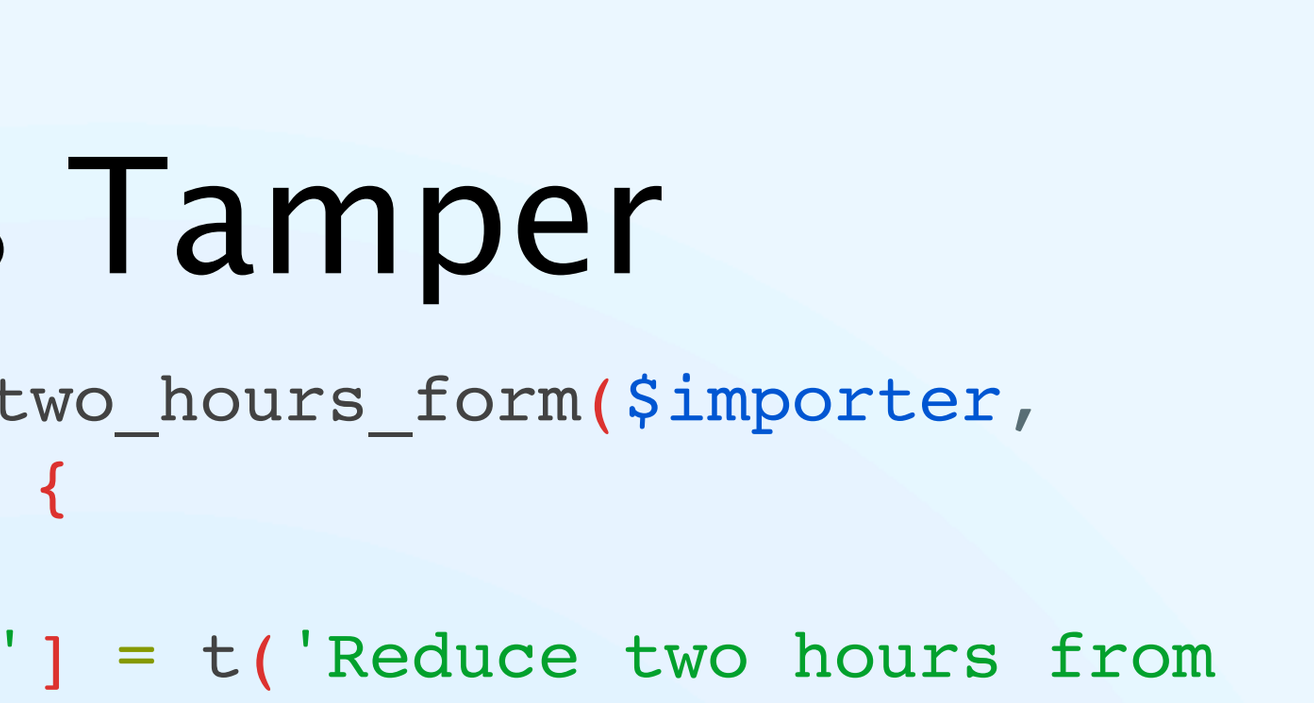


mearra

Feeds Tamper

```
function ttt_site_minus_two_hours_form($importer,
$element_key, $settings) {
  $form = array();
  $form['help']['#markup'] = t('Reduce two hours from
date. Useful if source time is something else than
UTC.');
  return $form;
}
```

Simple example of a form without any available settings



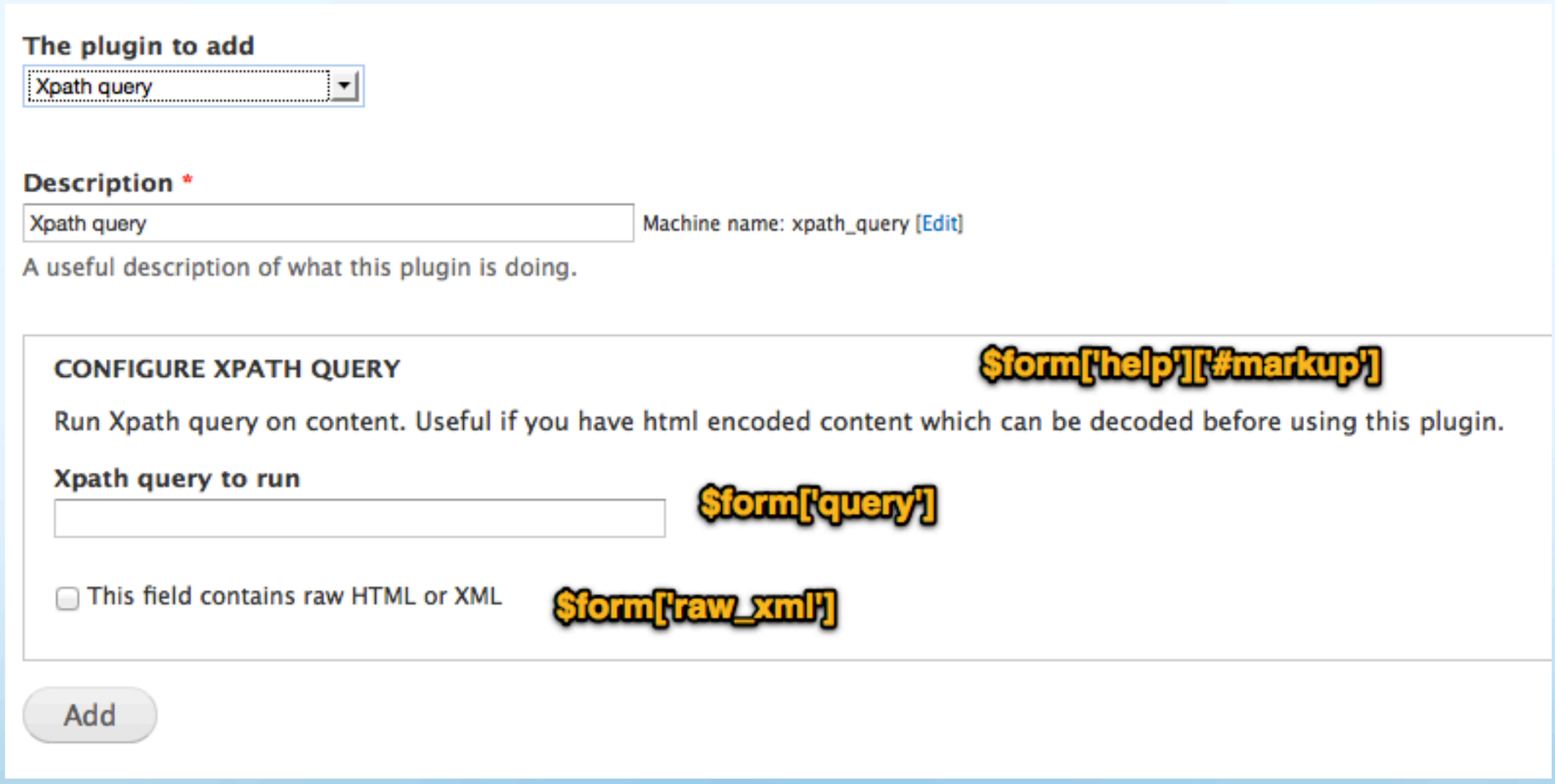
mearra

Feeds Tamper

```
function ttt_site_minus_two_hours_callback($result,
$item_key, $element_key, &$field, $settings) {
  $field = strtotime($field);
  $field = $field - 7200;
  $field = date('Y-m-d\TH:i:s', $field);
}
```

... and in the end the `$field` variable is modified according to requirements.
(the `$settings` variable is not used, because the form didn't have any items)

Feeds Tamper

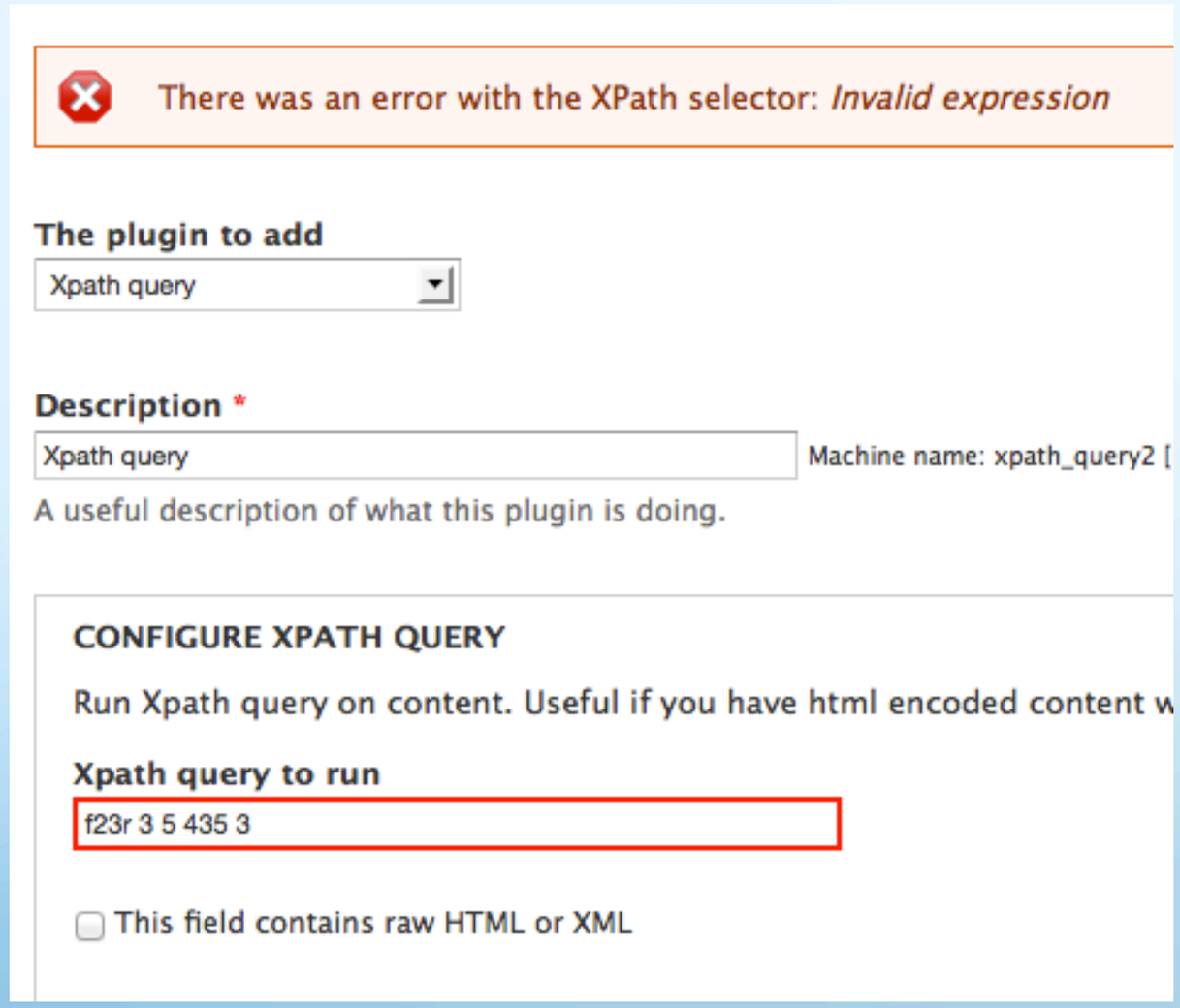


More complex example includes form items which can be used in the callback function. They will be available in the callback function of the plugin as a $settings array.

mearra

Feeds Tamper

Also validation functions can be used like in any other form in Drupal for the values



mearra

Feeds Tamper

This plugin uses the values from the `$settings` form and modifies the `$field` variable depending on the required logic.

```php
function feeds_tamper_xpath_query_callback ($result, $item_key, $element_key, &$field, $settings) {
  // Changing encoding for the query to support more characters
  $input = "<?xml version='1.0' encoding='UTF-8'?>" . $field;

  $doc = new DOMDocument();
  $doc->loadXML($input);
  $xpath = new DOMXpath($doc);
  $elements = $xpath->query($settings['query']);
  // Too many matches
  if ($elements->length > 1) {
    $message = check_plain(t('There were more than one result matching the query. Only one item should match the XPath query.'));
    drupal_set_message($message, 'error');
    $field = NULL;
  }
  // If there is exactly one result, then we return that
  elseif ($elements->length == 1) {
    foreach ($elements as $element) {
      if ($settings['raw_xml']) {
        $field = $doc->saveXML($element);
      }
      else {
        $field = $element->nodeValue;
      }
    }
  }
  else {
    // Setting field to NULL if there is no content, so Feeds won't try to process it
    $field = NULL;
  }
}
```

mearra

Use case: Real life example

News are imported to Drupal from NewsPilot publishing system in XML-format, which cannot be changed. Multiple images have to be saved per xml-file with captions.

Data source: NewsPilot
Data format: XML
Data manipulation:
Output image source and title in specific format for custom image mapper. (Feeds Tamper + custom plugin)
Parsing: Querypath, XPath
Processor: Custom NodeProcessor

Feeds Tamper

```
<articlepart id="272333" refType="ArticlePart">
  <article_id id="201341"/>
  <article_part_type_id id="1">Article</article_part_type_id>
  <data>
    <npdoc xmlns="http://www.infomaker.se/npdoc/2.1" version="2.1"
xml:lang="fi">
      <headline>Title comes here</headline>
      <body>Bodytext with lots of stuff...</body>
```

Problem 1: the type of the **<articlepart>** element is not defined in the element itself, but it can be found in the child element in the **id attribute** of the **<article_part_type_id>** element.

Problem 2: XPath doesn't like the **xmlns definition,** and won't read inside the **<npdoc>** element.

Solution: QueryPath parses the document with CSS/jQuery-like selectors and works like a charm ignoring fancy namespaces.

mearra

XPath/QueryPath Parser

QueryPath: **articlepart:has**
(article_part_type_id#1) data body

Will select the body from this:

```
<articlepart id="272333" refType="ArticlePart">
  <article_id id="201341"/>
  <article_part_type_id id="1">Text</article_part_type_id>
  <data>
    <npdoc xmlns="http://www.infomaker.se/npdoc/2.1" version="2.1"
xml:lang="fi">
      <headline>Title comes here</headline>
      <body>Bodytext with lots of stuff...</body>
```

mearra

Multiple images with attributes

```
<?xml version="1.0" encoding="UTF-8"?>
<npexchange xmlns="..." xmlns:xsi="..." version="3.5">
  <article id="201341" refType="Article">
    <articleparts>
      <articlepart id="272333" refType="ArticlePart">
        <article_id id="201341"/>
        <article_part_type_id id="1">Artikkeli</article_part_type_id>
        <data>
          <npdoc xmlns="..." version="2.1" xml:lang="fi">
            <imagecontainers>
              <imagecontainer>
                <image>
                  <description>Very cool image</description>
                  <image_author_name>The Drupal man</image_author_name>
                  <name>drupalman.jpg</name>
                </image>
...
```

Original structure (I have excluded lots of extra fields)

mearra

Multiple images with attributes

QueryPath
articlepart:has(article_part_type_id#1) imagecontainers
will choose each <imagecontainer> from the original XML document

```
<imagecontainer>
  <image>
    <description>Very cool image</description>
    <image_author_name>The Drupal man</image_author_name>
    <name>drupalman.jpg</name>
  </image>
</imagecontainer>
```

mearra

Feeds Tamper

Feeds doesn't support importing multiple images from a XML-file per node, so a custom image mapper was used.

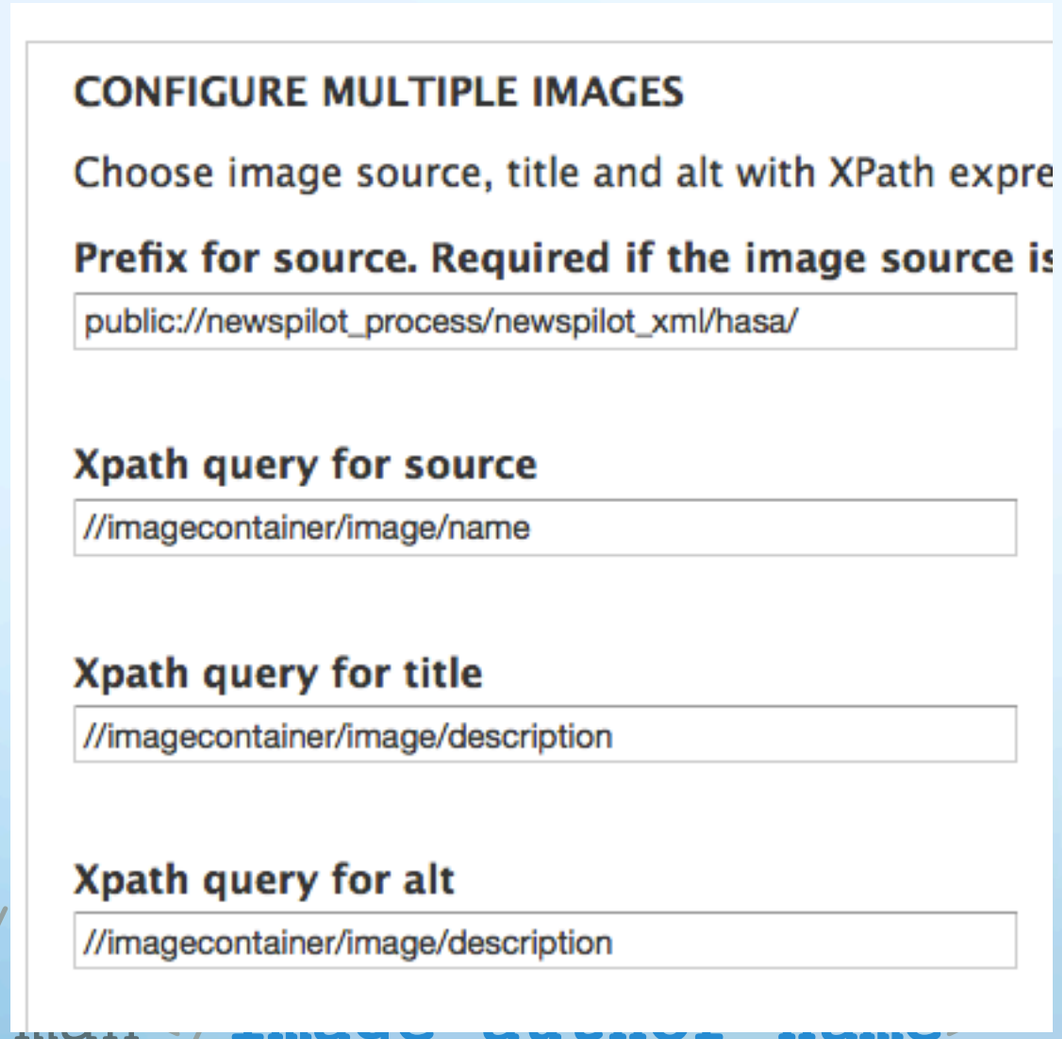
There is a issue where this problem is tackled:
http://drupal.org/node/1080386#comment-4350746

We modified that to accept the image in the following format:
src="path/image.jpg" alt="Text" title="Text";

mearra

Multiple images with attributes

Finally a custom Feeds Tamper plugin is used to read the field values with XPath and formatted to proper format for the custom image mapper




```
<imagecontainer>
  <image>
    <description>Very cool image</
    <image_author_name>The Drupal man</image_author_name>
    <name>drupalman.jpg</name>
  </image>
</imagecontainer>
```

mearra

Q&A

mearra

mearra