Textpattern

Textpattern CMS is a free, PHP open source CMS (content management system) with a browser-based interface in over 50 languages.

- Article 文章标签
- TinyMCE 使用 Ajax 和 PHP 上传图像
- TinyMCE图片上传处理 PHP image upload handler

Article □□□

Article

Syntax

```
<txp:article />
```

The powerful **article** tag can be used as either a *single* tag or *container* tag and used to output one or more articles depending on the context it is used and its attributes. Default attributes will be used when nothing specific is assigned.

It may be used as a *container* tag, in which case it must be specified as an opening and closing pair of tags, like this:

```
<txp:article>
   ...contained statements...
</txp:article>
```

This is equivalent to putting the contained statements into a form named `my_form` and using `<txp:article form="my_form" />`.

The tag is context-sensitive, which means it will grab articles from the currently viewed section/category/author, etc.

When used on the front page, article's context will include articles from all sections set to display via 'Section appears on default page?' settings (see the Sections panel for more information).

Note: `<txp:article />` is **not** the same as `<txp:article_custom />` - you can check out how that tag differs if you're unsure of the differences!

Attributes

The tag will accept the following content/behaviour and presentation attributes (**case-sensitive**) as well as the global attributes :

`allowoverride="boolean"`
: Whether to use override forms for the generated article list.
Values: `0` (no) or `1` (yes).
Default: `1`.

`customfieldname="value"`
: Restrict to articles with specified value for specified custom field name. Replace `customfieldname` with the actual name of the custom field.
Important: Using dashes `-` or spaces may cause errors or render this feature ineffectual. Underscores in both custom field names and values are confirmed to work.

`exclude="article id(s) or field(s)"` v4.6.0+
: Exclude a specific article or list of articles (each ID separated by a comma), or the articles with matching fields (author, category, etc.).
Default: unset.

`form="form name"`
: Use specified form template to process each article.
Default: `default`.

`frontpage="boolean"`
: Include only those articles with 'Section appears on front page?' set on the Sections panel. If set to `0`, all articles are displayed.
Values: `0` (no) or `1` (yes).
Default: `1`.

`keywords="keyword(s)"`
: Restrict to articles with specified keyword(s).

`limit="integer"`
: The number of articles to display.
Default: `10`.

`listform="form name"`
: Use specified form when page is displaying an article list.
Default: `article_listing`.

`match="field"` v4.6.0+
: Use comma-separated field(s) to match articles to the given URL parameters. Incoming variable names may be remapped by specifying the variable after an '=' sign. See Example 8.
Values:
`category`.
`category1`.
`category2`.
`keywords`.
any custom field.
Default: unset.

`offset="integer"`
: The number of articles to skip.
Default: `0`.

`pageby="integer"` v4.0.2+

The number of articles to jump forward or back when an older or newer link is selected. Allows you to call the article tag several times on a page without messing up older/newer links.

Default: value matches the value assigned to `limit`.

`pgonly="boolean"`

Do the article count, but do not display anything. Used when you want to show a search result count, or article navigation tags **before** the list of articles. Just make sure that, other than `pgonly`, both article tags are identical (form-related attributes are the exception, they do not need to be assigned).

Values: `0` (no) or `1` (yes).

Default: `0`.

`searchall="boolean"`

When outputting search results, include only those articles with 'Include in site search' set on the Sections panel. If set to `0`, only articles in the current section are displayed.

Values: `0` (no) or `1` (yes).

Default: `1`.

`searchform="form name"`

The form to be used for your customized search results output.

Default: `search_results`.

`searchsticky="boolean"`

When outputting search results, include articles with status `sticky`.

Values: `0` (no) or `1` (yes).

Default: `0`.

`sort="sort value(s)"` v4.0.4+

How to sort the resulting article list. Specify a value from the ones below, followed by a space and then add either `asc` or `desc` to sort in ascending or descending order, respectively.

Note: values are case sensitive on some servers.

Values:

`AuthorID` (author name).
`Category1`.
`Category2`.
`comments_count`.
`custom_n` where `n` is the number of your custom field - for numeric values use `(custom_n+0)`.
`Expires` (expiry date).
`ID` (article id#).
`Image` (article image id#).
`Keywords`.
`LastMod` (date last modified).
`LastModID` (author name of last modification).
`Posted` (date posted).
`rand()` (random).
`Section`.
`Status`.
`Title`.
`url_title`.

Each field in the `textpattern` database table can be used as a sort key.
When viewing a search results list, `score` (how well the search terms match the article) is available as an additional value.
Default: `Posted desc` (`score desc` for search results).

`status="status"`

Restrict to articles with the specified `status`.
Values: `live` or `sticky`.
Default: `live`.

`time="time"` v4.7.0+

Restrict to articles posted within specified timeframe.

Values: `past`, `future`, `any` (both `past` and `future`) or a PHP-compatible date format. In the latter case, `time` will be considered as the end date of the interval started by `month` or `expired` attribute.
Default: `past`.

Note on ‘article list’ vs. ‘individual article’ context

The **article** tag is context-sensitive. It will produce different results depending on whether the page being viewed is an article list or an individual article. Article-list context includes the default (home) page, section landing pages, and category pages. Individual-article context applies on an article page (i.e. a page with a URL like `https://example.com/archives/24/my-article`).

Examples

Example 1: Basic use as single tag

Here is the **article** tag responsible for the main content of the home page on a new Textpattern installation:

```
<txp:article limit="5" />
```

Calls the `default` article form, which may contain any variation of article output you want to create. The `default` form cannot be deleted; it is the form you see on first viewing the Forms panel.

Uses the `limit` attribute to specify the maximum number of articles displayed in article list context (if not specified, this defaults to `10`).

Example 2: Specifying a form

Expanding on **Example 1**, here is the `article` tag responsible for showing lists of articles by category in the default page of a new Textpattern installation:

```
<txp:article form="article_listing" limit="5" />
```

In article list context, the form named `article_listing` will be processed and displayed for each article in the list. In individual article context, the `default` form would be used.

To see this in action, on a new Textpattern install, from the home page select one of the category links near the bottom (right above the Comment link). Note the URL, similar to `https://example.com/category/meaningful-labor`. The `category` in the URL means this is a listing of articles by category. Here you see only the article title, posting date and article information (not the full article itself), because that is what is contained in the form named `article_listing`. Now select the article title. Note the URL, similar to `https://example.com/articles/1/welcome-to-your-site`. This is an individual article page. Once again you can see the full article, this time with comments showing (if comments are enabled).

Example 3: Offsetting article display

Continuing from the previous examples:

```
<txp:article listform="article_listing" limit="5" offset="2" />
```

Here we include the `offset` attribute to offset article display by `2` articles. This means the five articles that will be displayed (i.e. `limit="5"`) in article list context will begin with the third most recent article published in the site (the offset will not be applied in individual article context).

Why might you do it? Offsetting articles is useful in situations where the most recent article(s) are already accessible in some way and you don’t want them appearing again in normal article flow.

Example 4: Using pageby to split article output on a page

```
<div class="first">
    <txp:article limit="1" pageby="10" />
</div>
<div class="middle">
    <txp:article limit="8" offset="1" pageby="10" />
```

```
</div>
<div class="last">
   <txp:article limit="1" offset="9" pageby="10" />
</div>
```

Another:

```
<txp:article limit="5" pageby="10" />
<!-- google ad -->
<txp:article limit="5" offset="5" pageby="10" />
```

The `pageby` number should be the total number of articles displayed on the page. Without `pageby`, each article tag would page independently based on its own `limit`, as if it was the only article tag.

Example 5: Combined with custom fields

```
<txp:article colour="red" />
```

This code will display articles that have a custom field named `colour` with a value `red`.

Example 6: Article sorting

```
<txp:article sort="AuthorID asc, Posted desc" />
```

Uses the `sort` attribute to define values by which to order article output. In this case two values are declared. `AuthorID asc` first orders articles alphabetically by author names, then `Posted desc` orders them by date published (`desc` meaning newest to oldest).

Why might you do it? Sorting is a powerful way to group articles (e.g. by author), and/or give priority to articles most recently published (typically better for your website visitors).

Example 7: Build a `<section />`-separated list of article titles grouped by section

Desired result:

- `<section id="about (section name)">`
- About (section title)
- 1st Article from about section

- 2nd Article from about section
- …another article
- `</section>`
- `<section id="family (section name)">`
- Family (section title)
- 1st Article from family section
- 2nd Article from family section
- …another article
- `</txp:section>`
- …and so on

Create the following Form named `sections`:

```
<section id='<txp:section />'>
   <h2>
      <txp:section title />
   </h2>
   <+>
</section>
```

In Textpattern Page templates, add this tag to loop through all articles from all sections:

```
<txp:article sort="Section asc, Title asc" breakby="<txp:section />" breakform="sections">
   <h3>
      <txp:title />
   </h3>
</txp:article>
```

Other tags used: section, title.

Example 8: Filter articles from the URL (v4.8.6+)

Adding `<input>` elements to your page in an HTML `<form>` allows you to search and filter articles by various parameters when the form is submitted. For example, you could send `?c=cat&keywords=some,keywords,list` to the URL and interpret it with this tag:

```
<txp:article match="category, keywords" />
```

Or, if you need to fine-tune, you can remap the `key` URL parameter first by passing `?c=cat&key[]=some&key[]=keywords&key[]=list` and then interpreting it with:

```
<txp:article match="category1, keywords=key" />
```

Genealogy

Version 4.8.6

`match="keywords"` added.

Version 4.7.2

`breakform` attribute added, `breakby` attribute modified.

Version 4.7.0

`breakby` attribute added, `exclude` attribute modified.

Version 4.6.0

`exclude` and `match` attributes added.

Version 4.0.7

Can be used as a container.
`break` and `wraptag` attributes added.

Version 4.0.4

`sort` attribute added (replaces `sortby` and `sortdir`) attributes.
`sortby` and `sortdir` attributes deprecated.

Version 4.0.2

`pageby` attribute added.

TinyMCE 使用 Ajax 和 PHP 上传图像

TinyMCE 是众所周知的所见即所得 HTML 编辑器，广泛用于 Web 应用程序中创建 HTML 内容。它支持文本格式化、表格插入、链接插入、图像插入等功能。

TinyMCE 有许多插件，允许通过不同的来源将图像插入编辑器。但不允许浏览图像并上传到服务器以插入编辑器。为此，我们需要开发自己的功能。

用Ajax和PHP来实现TinyMCE图片上传。主要文件有：

- **index.php**: 用于加载 TinyMCE 编辑器的 PHP 文件。
- **tinymce_editor.js**: 用于初始化 TinyMCE 编辑器和图像上传 Ajax 的 JavaScript 文件。
- **upload.php**: 用于处理图像上传到服务器的 PHP 文件。

第 1 步: 包含 TinyMCE 和 jQuery

我们已经下载了 Ti`tinymce` `index.php` 件中。我们还将包括 jQuery [`tinymce_editor.js` 文件。

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/2.1.3/jquery.min.js"></script>
<script src="tinymce/tinymce.min.js"></script>
<script src="js/tinymce_editor.js"></script>
```

第二步：使用文本区域创建表单

`index.php` 文件中，我们将创建一个带有 Textarea 的表单来加载 TinyMCE 编辑器。

```
<div class="container">
<div class="row">
<form id="posts" name="posts" method="post">
<textarea name="message" id="message"></textarea><br>
</form>
</div>
</div>
```

第三步：初始化TinyMCE编辑器

tinymce_editor.js □□□□□□□□□□ TinyMCE □□□□□ Textarea □□□ TinyMCE WYSIWYG HTML □□□□

```
tinymce.init({
□selector: "textarea",
□plugins: "code",
□toolbar: "undo redo | bold italic | alignleft aligncenter alignright alignjustify | bullist numlist outdent indent | link
code image_upload",
□menubar:false,
    statusbar: false,
□content_style: ".mce-content-body {font-size:15px;font-family:Arial,sans-serif;}",
□height: 200
});
```

□□□□□□□□□□□□

tinymce_editor.js □□□□□□□□□□ image_upload □□□□□□□□□□□□□□□□□□□□□□□□□□□□

```
setup: function(ed) {□□
□ed.addButton('image_upload', {
□□tooltip: 'Upload Image',
□□icon: 'image',
□□onclick: function () {
□□□fileInput.trigger('click');
□□}
□});
}
```

□□□□□□□□□□□□

tinymce_editor.js □□□□□□ upload.php □□□□□□□□□□□□□□□□□□□□ TinyMCE Ajax □□□□□□

```
var fileInput = $('<input id="tinymce-uploader" type="file" name="pic" accept="image/*"
style="display:none">');
$(ed.getElement()).parent().append(fileInput);

fileInput.on("change",function(){□□□
```

```
var file = this.files[0];
var reader = new FileReader();
var formData = new FormData();
var files = file;
formData.append("file",files);
formData.append('filetype', 'image');
jQuery.ajax({
  url: "upload.php",
  type: "post",
  data: formData,
  contentType: false,
  processData: false,
  async: false,
  success: function(response){
    var fileName = response;
    if(fileName) {
      ed.insertContent('<img src="upload/'+fileName+'"/>');
    }
  }
});
reader.readAsDataURL(file);
});
```

Step6

upload.php upload

```
<?php
$fileName = $_FILES['file']['name'];
$fileType = $_POST['filetype'];
if($fileType == 'image'){
  $validExtension = array('png','jpeg','jpg');
}
$uploadDir = "upload/".$fileName;
$fileExtension = pathinfo($uploadDir, PATHINFO_EXTENSION);
$fileExtension = strtolower($fileExtension);
if(in_array($fileExtension, $validExtension)){
   if(move_uploaded_file($_FILES['file']['tmp_name'],$uploadDir)){
```

```
    echo $fileName;
  }
}
?>
```

TinyMCE□□□□□ PHP image up handler

□□□ TinyMCE V5□□

□□□□□□□ https://www.tiny.cloud/docs/advanced/php-upload-handler/

```
<?php
 /*****************************************************
  * Only these origins are allowed to upload images *
  *****************************************************/
 $accepted_origins = array("http://localhost", "http://192.168.1.1", "http://example.com");

 /*********************************************
  * Change this line to set the upload folder *
  *********************************************/
 $imageFolder = "images/";

 if (isset($_SERVER['HTTP_ORIGIN'])) {
   // same-origin requests won't set an origin. If the origin is set, it must be valid.
   if (in_array($_SERVER['HTTP_ORIGIN'], $accepted_origins)) {
     header('Access-Control-Allow-Origin: ' . $_SERVER['HTTP_ORIGIN']);
   } else {
     header("HTTP/1.1 403 Origin Denied");
     return;
   }
 }

 // Don't attempt to process the upload on an OPTIONS request
 if ($_SERVER['REQUEST_METHOD'] == 'OPTIONS') {
   header("Access-Control-Allow-Methods: POST, OPTIONS");
   return;
 }

 reset ($_FILES);
```

```
  $temp = current($_FILES);
  if (is_uploaded_file($temp['tmp_name'])){
    /*
      If your script needs to receive cookies, set images_upload_credentials : true in
      the configuration and enable the following two headers.
    */
    // header('Access-Control-Allow-Credentials: true');
    // header('P3P: CP="There is no P3P policy."');

    // Sanitize input
    if (preg_match("/([^\w\s\d\-_~,;:\[\]\(\).])|([\.]{2,})/", $temp['name'])) {
        header("HTTP/1.1 400 Invalid file name.");
        return;
    }

    // Verify extension
    if (!in_array(strtolower(pathinfo($temp['name'], PATHINFO_EXTENSION)), array("gif", "jpg", "png"))) {
        header("HTTP/1.1 400 Invalid extension.");
        return;
    }

    // Accept upload if there was no origin, or if it is an accepted origin
    $filetowrite = $imageFolder . $temp['name'];
    move_uploaded_file($temp['tmp_name'], $filetowrite);

    // Determine the base URL
    $protocol = isset($_SERVER['HTTPS']) && $_SERVER['HTTPS'] == 'on' ? "https://" : "http://";
    $baseurl = $protocol . $_SERVER["HTTP_HOST"] . rtrim(dirname($_SERVER['REQUEST_URI']), "/") . "/";

    // Respond to the successful upload with JSON.
    // Use a location key to specify the path to the saved image resource.
    // { location : '/your/uploaded/image/file'}
    echo json_encode(array('location' => $baseurl . $filetowrite));
  } else {
    // Notify editor that the upload failed
    header("HTTP/1.1 500 Server Error");
  }
?>
```