

Table of Contents

1 article	1
1.1 Attributes.....	1
1.2 Note on Article List vs. Individual Article context.....	2
1.3 Examples.....	2
1.4 Genealogy.....	3
2 article custom	4
2.1 Attributes.....	4
2.2 Examples.....	5
2.3 Genealogy.....	6
3 article id	7
3.1 Attributes.....	7
3.2 Examples.....	7
4 article image	8
4.1 Attributes.....	8
4.2 Examples.....	8
4.3 Genealogy.....	9
5 Category: Article Tags	10
6 article url title	11
6.1 Attributes.....	11
6.2 Genealogy.....	11
7 author	12
7.1 Attributes.....	12
7.2 Examples.....	12
7.3 Genealogy.....	12
8 body	13
8.1 Attributes.....	13
8.2 Examples.....	13
9 breadcrumb	14
9.1 Attributes.....	14
9.2 Examples.....	14
10 category	15
10.1 Attributes.....	15
10.2 Examples.....	15
10.3 Genealogy.....	16
11.1 Examples.....	17
11.2 Genealogy.....	18
11 category list	18
12 category1	19
12.1 Attributes.....	19
12.2 Examples.....	19
12.3 Genealogy.....	19
13 category2	20
13.1 Attributes.....	20
13.2 Examples.....	20
13.3 Genealogy.....	20
14 comment anchor	21
14.1 Attributes.....	21
14.2 Examples.....	21
15 comment email	22
15.1 Attributes.....	22
15.2 Examples.....	22
16 comment email input	23
16.1 Attributes.....	23
16.2 Examples.....	23

Table of Contents

17 comment id	24
17.1 Attributes.....	24
17.2 Examples.....	24
18 comment message	25
18.1 Attributes.....	25
18.2 Examples.....	25
19 comment message input	26
19.1 Attributes.....	26
20 comment name	27
20.1 Attributes.....	27
20.2 Examples.....	27
21 comment name input	28
21.1 Attributes.....	28
21.2 Examples.....	28
22 comment permalink	29
22.1 Attributes.....	29
22.2 Examples.....	29
23 comment preview	30
23.1 Attributes.....	30
23.2 Examples.....	30
24 comment remember	31
24.1 Attributes.....	31
24.2 Examples.....	31
25 comment submit	32
25.1 Attributes.....	32
25.2 Examples.....	32
26 Category:Comment Tags	33
27 comment time	34
27.1 Attributes.....	34
27.2 Examples.....	34
28 comment web	35
28.1 Attributes.....	35
28.2 Examples.....	35
29 comment web input	36
29.1 Attributes.....	36
29.2 Examples.....	36
30 comments	37
30.1 Attributes.....	37
31 comments count	39
31.1 Attributes.....	39
31.2 Examples.....	39
32 comments error	40
32.1 Attributes.....	40
32.2 Examples.....	40
33 comments form	41
33.1 Attributes.....	41
33.2 Examples.....	41
33.3 Genealogy.....	42
34 comments help	43
34.1 Attributes.....	43
34.2 Examples.....	43

Table of Contents

35 comments invite	44
35.1 Attributes.....	44
35.2 Examples.....	44
36 comments preview	45
36.1 Attributes.....	45
36.2 Genealogy.....	45
37 Category:Conditional Tags	46
38 css	47
38.1 Attributes.....	47
38.2 Examples.....	47
38.3 Genealogy.....	47
39 custom field	48
39.1 Attributes.....	48
39.2 Examples.....	48
39.3 Genealogy.....	49
40 else	50
40.1 Attributes.....	50
40.2 Examples.....	50
41 email	51
41.1 Attributes.....	51
41.2 Examples.....	51
41.3 Genealogy.....	51
42 Category:Error Handling Tags	52
43 error message	53
43.1 Attributes.....	53
43.2 Examples.....	53
44 error status	54
44.1 Attributes.....	54
44.2 Examples.....	54
45 excerpt	55
45.1 Attributes.....	55
45.2 Related Tags.....	55
45.3 Examples.....	55
46 expires	56
46.1 Attributes.....	56
46.2 Examples.....	56
46.3 Genealogy.....	56
46.4 Related.....	56
47 feed link	57
47.1 Attributes.....	57
47.2 Examples.....	57
47.3 Genealogy.....	57
48 file download	58
48.1 Attributes.....	58
48.2 Examples.....	58
49 file download category	59
49.1 Attributes.....	59
49.2 Examples.....	59
50 file download created	60
50.1 Attributes.....	60
50.2 Examples.....	60
51 file download description	61
51.1 Attributes.....	61
51.2 Examples.....	61

Table of Contents

51 file download description	
51.3 Genealogy.....	61
52 file download downloads.....	62
52.1 Attributes.....	62
52.2 Examples.....	62
53 file download id.....	63
53.1 Attributes.....	63
53.2 Examples.....	63
54 file download link.....	64
54.1 Attributes.....	64
54.2 Examples.....	64
55 file download list.....	65
55.1 Attributes.....	65
55.2 Examples.....	65
55.3 Genealogy.....	66
56 file download modified.....	67
56.1 Attributes.....	67
56.2 Examples.....	67
57 file download name.....	68
57.1 Attributes.....	68
57.2 Examples.....	68
58 file download size.....	69
58.1 Attributes.....	69
58.2 Examples.....	69
59 Category:File Tags.....	70
60 Category:Future Tags.....	71
61 hide.....	72
61.1 Attributes.....	72
61.2 Examples.....	72
62 if article author.....	73
62.1 Attributes.....	73
62.2 Examples.....	73
63 if article category.....	74
63.1 Attributes.....	74
63.2 Examples.....	74
64 if article id.....	76
64.1 Attributes.....	76
64.2 Examples.....	76
64.3 Genealogy.....	76
65 if article image.....	77
65.1 Attributes.....	77
65.2 Examples.....	77
65.3 Genealogy.....	77
66 if article list.....	78
66.1 Attributes.....	78
66.2 Examples.....	78
67 if article section.....	79
67.1 Attributes.....	79
67.2 Examples.....	79
68 if author.....	80
68.1 Attributes.....	80
68.2 Examples.....	80

Table of Contents

69 if category.....	.81
69.1 Attributes.....	.81
69.2 Examples.....	.81
70 if comments.....	.83
70.1 Attributes.....	.83
70.2 Examples.....	.83
71 if comments allowed.....	.84
71.1 Attributes.....	.84
71.2 Examples.....	.84
72 if comments disallowed.....	.85
72.1 Attributes.....	.85
72.2 Examples.....	.85
73 if comments error.....	.86
73.1 Attributes.....	.86
73.2 Examples.....	.86
74 if comments preview.....	.87
74.1 Attributes.....	.87
74.2 Examples.....	.87
75 if custom field.....	.88
75.1 Attributes.....	.88
75.2 Examples.....	.88
76 if different.....	.90
76.1 Attributes.....	.90
76.2 Examples.....	.90
77 if excerpt.....	.92
77.1 Attributes.....	.92
77.2 Examples.....	.92
78 if expired.....	.93
78.1 Attributes.....	.93
78.2 Examples.....	.93
78.3 Related.....	.93
78.4 Genealogy.....	.93
79 if expires.....	.94
79.1 Attributes.....	.94
79.2 Examples.....	.94
79.3 Related.....	.94
79.4 Genealogy.....	.94
80 if first article.....	.95
80.1 Attributes.....	.95
80.2 Examples.....	.95
81 if first category.....	.96
81.1 Attributes.....	.96
81.2 Examples.....	.96
81.3 Genealogy.....	.96
82 if first section.....	.97
82.1 Attributes.....	.97
82.2 Examples.....	.97
82.3 Genealogy.....	.97
83 if individual article.....	.98
83.1 Attributes.....	.98
83.2 Examples.....	.98
84 if keywords.....	.99
84.1 Attributes.....	.99
84.2 Examples.....	.99
84.3 Genealogy.....	.99

Table of Contents

85 if last article	100
85.1 Attributes.....	100
85.2 Examples.....	100
86 if last category	101
86.1 Attributes.....	101
86.2 Examples.....	101
86.3 Genealogy.....	101
87 if last section	102
87.1 Attributes.....	102
87.2 Examples.....	102
87.3 Genealogy.....	102
88 if plugin	103
88.1 Attributes.....	103
88.2 Examples.....	103
89 if search	104
89.1 Attributes.....	104
89.2 Examples.....	104
90 if search results	105
90.1 Attributes.....	105
90.2 Examples.....	105
90.3 Genealogy.....	106
91 if section	107
91.1 Attributes.....	107
91.2 Examples.....	107
92 if status	108
92.1 Attributes.....	108
92.2 Examples.....	108
93 if variable	109
93.1 Attributes.....	109
93.2 Examples.....	109
93.3 Genealogy.....	109
94 image	110
94.1 Attributes.....	110
94.2 Examples.....	110
94.3 Genealogy.....	111
95 image display	112
95.1 Attributes.....	112
95.2 Examples.....	112
96 image index	113
96.1 Attributes.....	113
96.2 Examples.....	113
96.3 Reference.....	114
97 Category:Image Tags	115
98 keywords	116
98.1 Attributes.....	116
98.2 Examples.....	116
99 lang	117
99.1 Attributes.....	117
99.2 Examples.....	117
100 link	118
100.1 Attributes.....	118
100.2 Examples.....	118

Table of Contents

101 link category	119
101.1 Attributes.....	119
101.2 Examples.....	119
102 link date	120
102.1 Attributes.....	120
102.2 Examples.....	120
103 link description	121
103.1 Attributes.....	121
103.2 Examples.....	121
103.3 Genealogy.....	121
104 link feed link	122
104.1 Attributes.....	122
104.2 Examples.....	122
104.3 Genealogy.....	122
105 link id	123
105.1 Attributes.....	123
105.2 Examples.....	123
105.3 Genealogy.....	123
106 link name	124
106.1 Attributes.....	124
106.2 Examples.....	124
106.3 Genealogy.....	124
107 Category:Link Tags	125
108 link to home	126
108.1 Attributes.....	126
108.2 Examples.....	126
109 link to next	127
109.1 Attributes.....	127
109.2 Examples.....	127
110 link to prev	128
110.1 Attributes.....	128
110.2 Examples.....	128
111 link url	129
111.1 Attributes.....	129
111.2 Examples.....	129
112 linkdescitle	130
112.1 Attributes.....	130
112.2 Examples.....	130
113 linklist	131
113.1 Attributes.....	131
113.2 Examples.....	131
113.3 Genealogy.....	132
114 Category:List Tags	133
115 Category:Markup Tags	134
116 meta author	135
116.1 Attributes.....	135
116.2 Examples.....	135
117 meta keywords	136
117.1 Attributes.....	136
117.2 Examples.....	136
118 modified	137
118.1 Attributes.....	137
118.2 Examples.....	137

Table of Contents

118 modified	
118.3 Genealogy.....	137
119 Category:Navigation Tags.....	138
120 newer.....	139
120.1 Attributes.....	139
120.2 Examples.....	139
121 next title.....	140
121.1 Attributes.....	140
121.2 Examples.....	140
122 older.....	141
122.1 Attributes.....	141
122.2 Examples.....	141
123 output form.....	143
123.1 Attributes.....	143
123.2 Examples.....	143
123.3 Genealogy.....	143
124 page title.....	144
124.1 Attributes.....	144
124.2 Examples.....	144
125 page url.....	145
125.1 Attributes.....	145
125.2 Examples.....	145
126 password protect.....	146
126.1 Attributes.....	146
126.2 Examples.....	146
127 permalink.....	147
127.1 Attributes.....	147
127.2 Examples.....	147
128 php.....	148
128.1 Attributes.....	148
128.2 Examples.....	148
129 popup.....	149
129.1 Attributes.....	149
129.2 Examples.....	149
130 posted.....	150
130.1 Attributes.....	150
130.2 Examples.....	150
130.3 Genealogy.....	151
131 prev title.....	152
131.1 Attributes.....	152
131.2 Examples.....	152
132 Category:Programmer Tags.....	153
133 recent articles.....	154
133.1 Attributes.....	154
133.2 Examples.....	155
133.3 Genealogy.....	155
134 recent comments.....	156
134.1 Attributes.....	156
134.2 Examples.....	156
134.3 Genealogy.....	157
135 related articles.....	158
135.1 Attributes.....	158
135.2 Examples.....	159

Table of Contents

135 related articles	
135.3 Genealogy.....	159
136 rsd.....	160
136.1 Attributes.....	160
136.2 Examples.....	160
136.3 Genealogy.....	160
137 search input.....	161
137.1 Attributes.....	161
137.2 Examples.....	161
137.3 Genealogy.....	161
138 search result count.....	163
138.1 Attributes.....	163
138.2 Examples.....	163
139 search result date.....	164
139.1 Attributes.....	164
139.2 Examples.....	164
140 search result excerpt.....	165
140.1 Attributes.....	165
140.2 Examples.....	165
140.3 Genealogy.....	165
141 search result title.....	166
141.1 Attributes.....	166
141.2 Examples.....	166
142 search result url.....	167
142.1 Attributes.....	167
142.2 Examples.....	167
143 Category:Search Tags.....	168
144 search term.....	169
144.1 Attributes.....	169
144.2 Examples.....	169
144.3 Genealogy.....	169
145 section.....	170
145.1 Attributes.....	170
145.2 Examples.....	170
145.3 Genealogy.....	171
146.1 Examples.....	172
146.2 Genealogy.....	172
146 section list.....	173
147 site name.....	174
147.1 Attributes.....	174
147.2 Examples.....	174
148 site slogan.....	175
148.1 Attributes.....	175
148.2 Examples.....	175
149 site url.....	176
149.1 Attributes.....	176
149.2 Examples.....	176
150 Category:Structural Tags.....	177
151 text.....	178
151.1 Attributes.....	178
151.2 Examples.....	178

Table of Contents

152 thumbnail	179
152.1 Attributes.....	179
152.2 Examples.....	179
152.3 Genealogy.....	179
153 title	181
153.1 Attributes.....	181
153.2 Examples.....	181
154 txp die	182
154.1 Attributes.....	182
154.2 Examples.....	182
155 variable	183
155.1 Attributes.....	183
155.2 Examples.....	183
155.3 Genealogy.....	185
156 yield	186
156.1 Attributes.....	186
156.2 Examples.....	186
156.3 Genealogy.....	186

1 article

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

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

The basic **article** tag can be used as either a *single* tag or *container* tag and used to output one or more articles depending on the attributes used. 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 "On front page".

See this [comparison of how article and article_custom differ](#).

1.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

customfieldname="value"

Restrict to articles with specified value for specified custom field name. Replace "customfieldname" with the name of the custom field.

keywords="keyword(s)"

Restrict to articles with specified keyword(s).

status="status"

Restrict to articles with specified status.

Values: live, sticky

Default: live.

time="time"

Restrict to articles posted within specified timeframe.

Values: past, future, or any (both past and future).

Default: past.

sort="sort value(s)"

How to sort resulting list.

Values:

ID (article id#)

AuthorID (author name)

LastMod (date last modified)

LastModID (author name of last modification)

Posted (date posted)

Expires (expiry date)

Title

Category1

Category2

comments_count

Status

Section

Keywords

Image (article image id#)

url_title

custom_1 through custom_10

(From 4.2.0 on: custom_n)

rand() (random).

Each field in the "textpattern"-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)

offset="integer"

The number of articles to skip.

Default: 0.

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

pageby="integer"
 The number of articles to jump forward or back when an older or newer link is clicked. 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).
 Default: 0 (no).

allowoverride="boolean"
 Whether to use override forms for the generated article list.
 Default: 1 (yes).

searchsticky="boolean"
 When outputting search results, include articles with status "sticky".
 Default: 0 (no).

form="form name"
 Use specified form.
 Default: default.

listform="form name"
 Use specified form when page is displaying an article list.

searchform="form name"
 The form to be used for your customized search results output.
 Default: search_results.

wraptag="tag text"
 The (X)HTML tag, without brackets, to wrap around each article.
 Default: unset.

class="class name"
 The (X)HTML class name to apply to the wraptag.
 Default: unset.

break="tag text"
 The (X)HTML tag (without brackets) or string to separate list items.
 Default: br.

1.2 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 front pages, and category pages. Individual-article context applies on an article page (i.e., a page with a URL like `<http://example.com/archives/24/my-article>`).

1.3 Examples

1.3.1 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 4.0.7 installation:

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

What this does...

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** tab.

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

1.3.2 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 4.0.7 installation:

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

What this does...

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 click on one of the category links near the bottom (right above the Comment link). Note the URL, similar to `<http://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 and posting date, because that is what is contained in the form named "article_listing".

Now click on the article title. Note the URL, similar to `<http://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.

1.3.3 Example 3: Offsetting Article Display

Continuing from the previous examples:

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

What this does...

Here we include the *offset* attribute to offset article display by two (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 you might 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.

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

```
<div id="first"><txp:article limit="1" pageby="10" /></div>
<div id="middle"><txp:article limit="8" offset="1" pageby="10" /></div>
<div id="last"><txp:article limit="1" offset="9" pageby="10" /></div>
```

Another:

```
<txp:article limit="5" pageby="10" /> <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. – From [Alex's original forum post](#).

1.3.5 Example 5: Combined with Custom Fields

This code will display articles that have a custom field named "colour" with a value "red":

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

1.3.6 Example 6: Article Sorting

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

What this does...

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 you might 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 web site visitors).

1.4 Genealogy

1.4.1 Version 4.0.7

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

1.4.2 Version 4.0.4

- *sort* added (replaces *sortby* and *sortdir*)
- *sortby* and *sortdir* deprecated

1.4.3 Version 4.0.2

- *pageby* added

2 article custom

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:article_custom />
```

The **article_custom** tag is a *single* or a *container* tag that provides a variety of custom options for sorting, selecting, and displaying articles (the tag will be replaced with one or more articles).

If used as a container, it must be specified as an opening and closing pair of tags, like this:

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

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

Unlike the [article](#) tag, **article_custom** will always return an article list and *is not context-sensitive*. This means while the [article](#) tag can only see posts within the currently viewed section/category/author and so forth, **article_custom** can see all posts from all sections, categories and authors unless you restrict it via attributes (see below), thus *context-sensitive navigation tags, such as [older](#) and [newer](#), will not work*.

[A comparison of how article and article_custom differ.](#)

Related Info: [Customizing txp:recent_articles.](#)

2.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

id="article ID"

Display the specific article or list of articles (each ID separated by a comma).

IMPORTANT: When a list is supplied, this does *not* imply a sort order.

Default: unset.

customfieldname="value"

Restrict to articles with specified value for specified custom field name. Replace "customfieldname" with the name of the custom field.

Default: unset.

section="section name"

Restrict to articles from specified section(s).

Value: (comma separated list of) section name(s).

Default: unset, retrieves from all sections.

category="category name"

Restrict to articles from specified category/ies. Note: the category names may be different to the Title you typed when you created the category, as the names are sanitized for URL use. Check the [Categories](#) tab to ensure you are using the correct names.

Value: (comma separated list of) category name(s).

Default: unset, retrieves from all categories.

keywords="keyword(s)"

Restrict to articles with specified keyword(s).

Default: unset.

excerpted="y"

Restrict to articles with/without an excerpt.

Value: `y` (yes, return only those containing an excerpt).

Default: unset (no, return all).

status="status"

Restrict to articles with specified status.

Values: `live`, `sticky`.

Default: `live`.

time="time"

Restrict to articles posted within specified timeframe.

Values: `past`, `future`, or `any` (both `past` and `future`).

Default: `past`.

month="yyyy"/"yyyy-mm"/"yyyy-mm-dd"

Restrict to articles posted within the specified year/month/day.

Default: unset.

author="author name"

Restrict to articles by specified author(s).

Value: (comma separated list of) author name(s).

Default: unset, retrieves from all authors.

sort="sort value(s)"
 How to sort resulting list.
 Values:

- ID (article id#)
- AuthorID (author)
- LastMod (date last modified)
- LastModID (author of last modification)
- Posted (date posted)
- Expires (expiry date)
- Title
- Category1
- Category2
- comments_count
- Status
- Section
- Keywords
- Image (article image id#)
- url_title
- custom_1 through custom_10
 (From 4.2.0 on: custom_n)
- rand() (random).

Each field in the "textpattern"-table can be used as a sort key.
 Default: Posted desc.

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

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

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

form="form name"
 Use specified form.
 Default: default.

wraptag="tag text"
 The (X)HTML tag, without brackets, to wrap around the article list.
 Default: unset.

class="class name"
 The (X)HTML class name to apply to the wraptag.
 Default: unset.

break="tag text"
 The (X)HTML tag (without brackets) or string to separate list items.
 Default: unset.

2.2 Examples

2.2.1 Example 1: List articles published in specified month

```
<txp:article_custom form="month_list" sort="Section asc" month="2004-10" />
```

2.2.2 Example 2: Select by keyword

```
<txp:article_custom sort="Posted desc" keywords="One" />
```

2.2.3 Example 3: Select by author

```
<txp:article_custom form="author_list" author="Parkling" />
```

the **author_list** article form might go thus.

```
<p><txp:permalink><txp:title /></txp:permalink></p>
```

Other tags used: [permalink](#), [title](#)

2.2.3.1 Example 3a: Container tag

The following is exactly equivalent to Example 3:

```
<txp:article_custom author="Parkling">
<p><txp:permlink><txp:title /></txp:permlink></p>
</txp:article_custom>
```

2.2.4 Example 4: Combined with custom fields

This code will display articles that have a custom field named "colour" with a value "red":

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

2.2.5 Example 5: Article sorting

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

What this does...

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 you might 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 web site visitors).

2.3 Genealogy

2.3.1 Version 4.0.7

- Can be used as a container tag.
- `id` can take a comma-separated list of IDs.
- `wraptag` and `break` attributes added.

2.3.2 Version 4.0.6

- support added for comma separated lists for section, category and author attributes

2.3.3 Version 4.0.4

- `listform` deprecated (it never made a difference to `article_custom` anyway)
- `sort` added (replaces `sortby` and `sortdir`)
- `sortby` and `sortdir` deprecated

3 article id

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:article_id />
```

The **article_id** tag is a *single* tag which returns the numeric ID of the article being displayed. This number will also be reflected as a part of the article permanent URL if it has been chosen as the *Permanent link mode* in the [Basic Preferences](#) tab.

3.1 Attributes

This tag has no attributes.

3.2 Examples

3.2.1 Example 1: Hyperlinked to the article

```
<txp:permalink><txp:article_id /></txp:permalink>
```

Other tags used: [permalink](#)

3.2.2 Example 2: Conditional use

This will only display the hyperlinked article ID when on an individual article page.

```
<txp:if_individual_article>  
Article ID: <txp:permalink><txp:article_id /></txp:permalink>  
</txp:if_individual_article>
```

Other tags used: [if_individual_article](#), [permalink](#)

4 article image

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:article_image />
```

The **article_image** tag is a *single* tag. Textpattern will replace this tag with the `` HTML tag matching the numeric ID or URL assigned when the article is posted.

The image to be associated with the tag is set under the **Write** tab. Click "Advanced Options" and enter either the URL of the image, or the Textpattern ID (a number set by Textpattern at upload) into the *Article image* field. Most of the time you will use the image ID here. Note that you can only assign a single image to each article.

4.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

thumbnail="boolean"

Use the thumbnail rather than full-size image.

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

Default: 0.

escape="html"

Escape HTML entities such as `<`, `>` and `&` for the image's `alt` and `title` attributes.

Value: `html` or `unset`.

Default: `html`.

wraptag="tag"

HTML tag to be used to wrap the `img` tag, specified without brackets.

Default: `unset`.

class="class name"

CSS `class` attribute to apply to the image (or to the `wraptag`, if set).

Default: `unset`.

html_id="id"

The HTML `id` attribute assigned to the image (or to the `wraptag`, if set).

Default: `unset`.

style="style rule"

Inline CSS style rule.

Default: `unset`.

align^{deprecated}="HTML value"

HTML `align` attribute for the `img` tag. Recommended that you use CSS via `class` or `html_id` attribute instead.

Default: `unset`.

4.2 Examples

4.2.1 Example 1: Use wraptag and class for styling

```
<txp:article_image wraptag="p" class="article-image" />
```

What this does...

This will wrap the image in paragraph tags, applying the class to the paragraph: `<p class="article-image"></p>`.

Why you might do it...

It gives you full control over the image's appearance using CSS.

Note

Without the `wraptag`, the class is applied directly to the `img` tag

4.2.2 Example 2: Link thumbnail to the article

Used in an article list form this will display an article list consisting of hyperlinked article images' thumbnails.

```
<txp:permalink><txp:article_image thumbnail="1" /></txp:permalink>
```

Other tags used: [permalink](#)

4.3 Genealogy

4.3.1 Version 4.2.0

- attribute `align` deprecated

4.3.2 Version 4.0.7

- default value for attribute `escape` changed from `unset` to `html`

4.3.3 Version 4.0.4

- `class`, `escape`, `html_id`, `thumbnail`, and `wraptag` added.

5 Category:Article Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Article Tags are a subcategory of the [Tag Reference](#). They are tags that represent a field in the textpattern table and show or otherwise reference the 'article' content type. See the [Articles](#) page.

[Download Category:Article_Tags book](#) (see [here](#) for more details)

6 article url title

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:article_url_title />
```

The ***article_url_title*** tag is a *single* tag which returns the dumbed-down "URL title" of the article being displayed. This URL title may also be part of the page's address depending on the *Permanent link mode* chosen in [Basic Preferences](#).

6.1 Attributes

This tag has no attributes.

6.2 Genealogy

6.2.0.1 Version 4.0.5

- Tag support added

7 author

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:author />
```

The **author** tag is a *single* tag that is used to return the name of the author of the currently displayed article.

7.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

link="boolean"

Make text a link to the author's posts.

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

Default: 0.

section="section name"

Only link to articles from the named section.

Default: unset.

this_section="boolean"

Only link to other articles from the same section as the current article.

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

Default: 0.

7.2 Examples

7.2.1 Example 1: Link to list of author's articles

The author's name in this article form is a hyperlink to a list of articles by this author.

```
<h1><txp:title /></h1>
```

```
<txp:body />
```

```
<p class="author-date">
```

```
Posted By: <txp:author link="1" /> @ <txp:posted />
```

```
</p>
```

Other tags used: [posted](#), [title](#), [body](#)

7.3 Genealogy

7.3.1 Version 4.0.4

- *section*, *this_section* added.

8 body

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:body />
```

The **body** tag is a *single* tag which is used to return the text, or content, of the article being displayed (the article itself).

8.1 Attributes

This tag has no attributes.

8.2 Examples

8.2.1 Example 1: Display the article text

```
<p><txp:title /></p>
<div class="post">
  <p><txp:author /> @ <txp:posted /><br />
  <txp:body /></p>
</div>
```

When used as part of your article form, this displays the article title, author and posted date, then the body text beneath that.

Other tags used: [author](#), [posted](#), [title](#)

9 breadcrumb

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:breadcrumb />
```

The **breadcrumb** tag is a *single* tag which is used to create [breadcrumb navigation](#). It provides either hyperlinked navigation, or plain text positional display, any time you are *not* on the Home page.

9.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

label="value"

This string will be prepended to the output.
Default: *Site name*.

sep="value"

Character to be used as the breadcrumb separator. N.B.: This attribute will be renamed **separator** from the next release.
Default: » .

title="boolean"

Whether to display the title or not.
Values: 1 (yes) or 0 (no, display name).
Default: 0.

link="value"

Whether to hyperlink breadcrumbs.
Values: 1 (yes) or 0 (no).
Default: 1.

linkclass="class name"

(X)HTML class attribute applied to the breadcrumb links.
Default: *noline*.

class="class name"

(X)HTML class attribute applied to the *wraptag*.
Default: unset.

wraptag="tag text"

(X)HTML tag (without brackets) to wrap around the output.
Default: *p*.

9.2 Examples

9.2.1 Example 1: Display a hyperlinked breadcrumb trail

```
<txp:breadcrumb label="Navigation" sep="::" link="1" wraptag="p" />
```

Provides hyperlinks to sections or categories in a breadcrumb style, linking back to your home page.

Breadcrumbs are not displayed on the "Default" section of your site

9.2.2 Example 2: Display a text only breadcrumb trail

```
<txp:breadcrumb label="Navigation" sep=":" link="0" wraptag="p" />
```

Provides a breadcrumb guide that reflects where a user is within the site's navigation.

10 category

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:category />
```

The **category** tag can be used as either a *single* or *containing* tag. It will display information of the category as defined by the `name` attribute, or the one currently being viewed. When used as a containing tag, it will turn the contents into a link to the category. Otherwise, it will return plain text.

May be used in any context.

10.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="category name"

Display specific category. Note the category name is specified in lower case regardless of how you typed its title in the Category tab. Also note that if you had called your category **My Category Name** it becomes **my-category-name** when used in tags
Default: unset (use current category).

type="category type"

Values: `article`, `image`, `link`, `file`.
Default: `article`.

title="boolean"

Whether to display category's title.
Values: `1` (yes) or `0` (no, display name).
Default: `0`.

link="boolean"

Whether to display as link.
Values: `1` (yes) or `0` (no).
Default: `0`.

section="section name"

Restricts category search to named section.
Default: current section (for backwards compatibility).

this_section="boolean"

Only link to articles from the current section. `section` attribute overrides this setting.
Values: `1` (yes) or `0` (no).
Default: `0`.

class="class name"

(X)HTML class attribute, applied to `wraptag`. If no `wraptag` is supplied (and `link="1"`), the class is applied to the anchor instead.
Default: unset.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around output.
Default: unset.

10.2 Examples

10.2.1 Example 1: Displays the current category name

```
<txp:category />
```

10.2.2 Example 2: Display hyperlinked category title

```
<txp:category title="1" link="1" />
```

10.2.3 Example 3: Display a specific category's title, hyperlinked

```
<txp:category name="articles" title="1" link="1" wraptag="p" />
```

10.2.4 Example 4: Container Example

```
<txp:category name="book">My books</txp:category>
```

10.3 Genealogy

10.3.1 Version 4.0.7

- Applies `class` attribute to the `<a>` element when `wraptag` is empty.

10.3.2 Version 4.0.4

- `this_section` added.

11 category list

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:category_list />
```

The **category_list** tag can be used as either a *single* tag or *container* tag which is used to produce a list of linked categories.

11.0.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

parent="category name"

Return only specified category and its "children" categories.

categories="category name(s)"

Comma-separated list of categories to include, displayed in the order specified (unless overridden by `sort` attribute). Use category names not Titles here -- note that TXP automatically converts the names to lower case and converts spaces to hyphens when they are created
Default: unset (all categories).

exclude="category name(s)"

List of category names which will be excluded from the list. `categories` takes precedence over `exclude`.
Default: unset.

type="category type"

Available values: article, image, link, file.
Default: article.

children="boolean"

Can limit the list depth to one level below the parent category.
Values: 0 (no children, i.e. only show one level below the parent) or 1 (all nested categories).
Default: 1

sort="sort value(s)"

How to sort the resulting list.
Values:

id
name
type
parent
title
rand() (random)

Default: name asc.

form="form name"

Use the specified form to process each included category.

label="text"

String to be prepended to the list.
Default: unset.

section="section name"

Link to specified section.
Default: unset, resulting in links without section restriction.

this_section="boolean"

Link to currently active section (overrides `section` attribute).
Values: 0 (no) or 1 (yes).
Default: 0.

labeltag="tag"

(X)HTML tag (without brackets) to wrap around `label`.
Default: unset.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around list.
Default: unset.

break="value"

(X)HTML tag (without brackets) or string used to separate list items.
Default: br.

class="class name"

(X)HTML class attribute to be applied to `wraptag`.
Default: category_list.

active_class="class name"

(X)HTML class attribute to be applied to the "active" or current link in a list (only works in the *single* tag without the `form` attribute).
Default: unset.

11.1 Examples

11.1.1 Example 1: Labelled category list

```
<txp:category_list label="Categories" wraptag="p" break="br" />
```

11.1.2 Example 2: As an unordered list

```
<txp:category_list break="li" wraptag="ul" />
```

Styles could go this way

```
.category_list {  
    list-style-type:none;  
}
```

11.1.3 Example 3: Set active class using the container tag

This code will add `class="active"` to the `` element around the "current" category in the list.

```
<txp:category_list wraptag="ul" break="">  
<li<txp:if_category name='<txp:category />'> class="active"</txp:if_category>>  
<txp:category title="1" link="1" />  
</li>  
</txp:category_list>
```

11.2 Genealogy

11.2.1 Version 4.0.7

- Can be used as a container tag.
- `form` and `children` attributes added.

11.2.2 Version 4.0.4

- `active_class`, `categories`, `exclude`, `section`, `this_section` added.

12 category1

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:category1 />
```

The **category1** tag can be used as either a *single* tag or *container* tag. It will display information of the category as defined by Cat1 of the article being displayed. When used as a containing tag, it will turn the contents into a link to that category. Otherwise, it will return plain text.

This tag may be used within either an article form, or in a page, wrapped in an [if_individual_article](#) conditional tag.

12.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

title="boolean"

Whether to output category title, rather than name.
Values: 1 (yes) or 0 (no, output name).
Default: 0.

link="boolean"

Whether to link to articles from the same category.
Values: 1 (yes) or 0 (no).
Default: 0.

section="section name"

Only link to articles from the specified section.
Default: unset.

this_section="boolean"

Whether to only link to articles from the section containing the current article.
Values: 1 (yes) or 0 (no, allow from any section).
Default: 0.

class="class name"

(X)HTML class attribute to be applied to wraptag.
Default: unset.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around output.
Default: unset.

12.2 Examples

12.2.1 Example 1: Category name in plain text

```
<txp:category1 />
```

12.2.2 Example 2: Hyperlinked category title

```
<txp:category1 link="1" title="1" />
```

If category1 is "General", this tag will display the word "General" as a hyperlink to a list of articles in the same category.

12.2.3 Example 3: Container example

```
<txp:category1>Other articles in category <txp:category1 title="1" /></txp:category1>
```

12.3 Genealogy

12.3.1 Version 4.0.4

- class, section, this_section, wraptag added.

13 category2

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:category2 />
```

The **category2** tag can be used as either a *single* tag or *container* tag. It will display information of the category as defined by Cat2 of the article being displayed. When used as a containing tag, it will turn the contents into a link to that category. Otherwise, it will return plain text.

This tag may be used within either an article form, or in a page, wrapped in an [if_individual_article](#) conditional tag.

13.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

title="boolean"

Whether to output category title, rather than name.
Values: 1 (yes) or 0 (no, output name).
Default: 0.

link="boolean"

Whether to link to articles from the same category.
Values: 1 (yes) or 0 (no).
Default: 0.

section="section name"

Only link to articles from the specified section.
Default: unset.

this_section="boolean"

Whether to only link to articles from the section containing the current article.
Values: 1 (yes) or 0 (no, allow from any section).
Default: 0.

class="class name"

(X)HTML [class attribute](#) to be applied to wraptag.
Default: unset.

wraptag="tag"

(X)HTML [tag](#) (without brackets) to wrap around output.
Default: unset.

13.2 Examples

13.2.1 Example 1: Category name in plain text

```
<txp:category2 />
```

13.2.2 Example 2: Hyperlinked category title

```
<txp:category2 link="1" title="1" />
```

If category2 is "General", this tag will display the word "General" as a hyperlink to a list of articles in the same category.

13.2.3 Example 3: Container example

```
<txp:category2>Other articles in category <txp:category2 title="1" /></txp:category2>
```

13.3 Genealogy

13.3.1 Version 4.0.4

- `class`, `section`, `this_section`, `wraptag` added.

14 comment anchor

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_anchor />
```

The ***comment_anchor*** tag is a *single* tag which is used to produce an empty anchor tag with an id attribute reflecting the comment ID. Used in the form that renders your comments (the default form is named "comments").

14.1 Attributes

This tag has no attributes.

14.2 Examples

14.2.1 Example 1: Generate current comment anchor

When the comment number is *000005* you will see:

```
<a id="000005"></a>
```

15 comment email

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_email />
```

The ***comment_email*** tag is a *single* tag which is used to display the commenter's email address, if entered at the time of posting. Used in a comments display form.

15.1 Attributes

This tag has no attributes.

15.2 Examples

15.2.1 Example 1: Comments Display Form with linked email and comment id

```
<txp:comment_message />
```

```
<p><small><a href="mailto:<txp:comment_email />">Email</a> |  
<txp:comment_permalink><txp:comment_id /></txp:comment_permalink></small></p>
```

Other tags used: [comment_id](#), [comment_message](#), [comment_permalink](#)

16 comment email input

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_email_input />
```

The ***comment_email_input*** tag is a *single* tag which is used to display a text entry field to accept the commenter's email address. Used in the comment input form.

16.1 Attributes

This tag has no attributes.

16.2 Examples

16.2.1 Example 1: Comment Form

```
<table cellpadding="4" cellspacing="0" border="0">
  <tr>
    <td align="right" valign="top">
      <txp:text item="name" />
    </td>
    <td valign="top">
      <txp:comment_name_input />
    </td>
    <td valign="top" align="left">
      <txp:comment_remember />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="email" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_email_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      http://
    </td>
    <td valign="top" colspan="2">
      <txp:comment_web_input />
    </td>
  </tr>
  <tr>
    <td valign="top" align="right">
      <txp:text item="message" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_message_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top"> </td>
    <td valign="top" align="left">
      <txp:comments_help />
    </td>
    <td align="right" valign="top">
      <txp:comment_preview />
      <txp:comment_submit />
    </td>
  </tr>
</table>
```

Other tags used: [comment_message_input](#), [comment_name_input](#), [comment_web_input](#), [comment_preview](#), [comment_remember](#), [comment_submit](#), [comments_help](#), [text](#)

17 comment id

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_id />
```

The **comment_id** tag is a *single* tag which is used to display the comment's internal id as assigned by Textpattern at the time of posting. Used in a comments display form.

17.1 Attributes

This tag has no attributes.

17.2 Examples

17.2.1 Example 1: Comments Display Form with linked comment id

```
<txp:comment_message />
```

```
<p><small>&#8212; <txp:comment_name /> <txp:comment_time />  
<txp:comment_permalink><txp:comment_id /></txp:comment_permalink></small></p>
```

Other tags used: [comment_message](#), [comment_name](#), [comment_permalink](#), [comment_time](#)

18 comment message

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_message />
```

The ***comment_message*** tag is a *single* tag which is used to display the message text, or comment. Used in a comments display form.

18.1 Attributes

This tag has no attributes.

18.2 Examples

18.2.1 Example 1: Comments Display Form

```
<txp:comment_message />
```

```
<p><small>&#8212; <txp:comment_name /> <txp:comment_time />  
<txp:comment_permalink>#</txp:comment_permalink></small></p>
```

Other tags used: [comment_name](#), [comment_permalink](#), [comment_time](#)

19 comment message input

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_message_input />
```

The **`comment_message_input`** tag is a *single* tag which is used to display a text entry field to accept the commenter's message text. Used in the comment input form.

19.1 Attributes

This tag has no attributes.

19.1.1 Examples

19.1.2 Example 1: Comment Form

```
<table cellpadding="4" cellspacing="0" border="0">
  <tr>
    <td align="right" valign="top">
      <txp:text item="name" />
    </td>
    <td valign="top">
      <txp:comment_name_input />
    </td>
    <td align="left" valign="top">
      <txp:comment_remember />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="email" />
    </td>
    <td align="left" valign="top" colspan="2">
      <txp:comment_email_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      http://
    </td>
    <td align="left" valign="top" colspan="2">
      <txp:comment_web_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="message" />
    </td>
    <td align="left" valign="top" colspan="2">
      <txp:comment_message_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top"> </td>
    <td align="left" valign="top">
      <txp:comments_help />
    </td>
    <td align="right" valign="top">
      <txp:comment_preview />
      <txp:comment_submit />
    </td>
  </tr>
</table>
```

Other tags used: [comment_email_input](#), [comment_name_input](#), [comment_web_input](#), [comment_preview](#), [comment_remember](#), [comment_submit](#), [comments_help](#), [text](#)

20 comment name

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_name />
```

The **comment_name** tag is a *single* tag which is used to display a link using the commenter's name as text. If email address is supplied and [allowed to be viewed](#), an email link is created. Otherwise, if a website is entered, the website URL is used. If neither is supplied, name displays as plain text.

Commenter's name and/or email address can be set as a requirement.

Used in a comments display form.

20.1 Attributes

link="boolean"

Whether to display as link or plain text.

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

Default is 1.

20.2 Examples

20.2.1 Example 1: Comments Display Form

```
<txp:comment_message />
```

```
<p><small>&#8212; <txp:comment_name /> <txp:comment_time />  
<txp:comment_permalink>#</txp:comment_permalink></small></p>
```

Other tags used: [comment_message](#), [comment_permalink](#), [comment_time](#)

21 comment name input

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_name_input />
```

The **`comment_name_input`** tag is a *single* tag which is used to display a text entry field to accept the commenter's name. Used in the comment input form.

21.1 Attributes

This tag has no attributes.

21.2 Examples

21.2.1 Example 1: Comment Form

```
<table cellpadding="4" cellspacing="0" border="0">
  <tr>
    <td align="right" valign="top">
      <txp:text item="name" />
    </td>
    <td valign="top">
      <txp:comment_name_input />
    </td>
    <td valign="top" align="left">
      <txp:comment_remember />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="email" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_email_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      http://
    </td>
    <td valign="top" colspan="2">
      <txp:comment_web_input />
    </td>
  </tr>
  <tr>
    <td valign="top" align="right">
      <txp:text item="message" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_message_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top"> </td>
    <td valign="top" align="left">
      <txp:comments_help />
    </td>
    <td align="right" valign="top">
      <txp:comment_preview />
      <txp:comment_submit />
    </td>
  </tr>
</table>
```

Other tags used: [comment_email_input](#), [comment_web_input](#), [comment_preview](#), [comment_remember](#), [comment_submit](#), [comments_help](#), [text](#)

22 comment permalink

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

`<txp:comment_permalink>`

The **`comment_permalink`** tag is a *container* tag which is used to return the permanent link of the article comment being displayed. The container tag wraps the text assigned to the link.

22.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

`anchor`="boolean"

Whether to apply the comment's id to the hyperlink tag (as the id attribute), setting this comment permanent link as the comment page anchor.

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

Default is 0.

22.2 Examples

22.2.1 Example 1: Display a link for the article comment being displayed

`<txp:comment_permalink>#</txp:comment_permalink>`

23 comment preview

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_preview />
```

The ***comment_preview*** tag is a *single* tag which is used to display a Preview button the user can use to preview the comment text. Used in the comment input form.

23.1 Attributes

This tag has no attributes.

23.2 Examples

23.2.1 Example 1: Comment Form

```
<table cellpadding="4" cellspacing="0" border="0">
  <tr>
    <td align="right" valign="top">
      <txp:text item="name" />
    </td>
    <td valign="top">
      <txp:comment_name_input />
    </td>
    <td valign="top" align="left">
      <txp:comment_remember />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="email" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_email_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      http://
    </td>
    <td valign="top" colspan="2">
      <txp:comment_web_input />
    </td>
  </tr>
  <tr>
    <td valign="top" align="right">
      <txp:text item="message" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_message_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top"> </td>
    <td valign="top" align="left">
      <txp:comments_help />
    </td>
    <td align="right" valign="top">
      <txp:comment_preview />
      <txp:comment_submit />
    </td>
  </tr>
</table>
```

Other tags used: [comment_email_input](#), [comment_name_input](#), [comment_web_input](#), [comment_remember](#), [comment_submit](#), [comments_help](#), [text](#)

24 comment remember

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_remember />
```

The **comment_remember** tag is a *single* tag which is used to display a check box input field. If checked the users details are remembered by the system the next time they open a comment form. Used in the comment input form.

24.1 Attributes

This tag has no attributes.

24.2 Examples

24.2.1 Example 1: Comment Form

```
<table cellpadding="4" cellspacing="0" border="0">
  <tr>
    <td align="right" valign="top">
      <txp:text item="name" />
    </td>
    <td valign="top">
      <txp:comment_name_input />
    </td>
    <td valign="top" align="left">
      <txp:comment_remember />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="email" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_email_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      http://
    </td>
    <td valign="top" colspan="2">
      <txp:comment_web_input />
    </td>
  </tr>
  <tr>
    <td valign="top" align="right">
      <txp:text item="message" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_message_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top"> </td>
    <td valign="top" align="left">
      <txp:comments_help />
    </td>
    <td align="right" valign="top">
      <txp:comment_preview />
      <txp:comment_submit />
    </td>
  </tr>
</table>
```

Other tags used: [comment_email_input](#), [comment_name_input](#), [comment_web_input](#), [comment_preview](#), [comment_submit](#), [comments_help](#), [text](#)

25 comment submit

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_submit />
```

The **comment_submit** tag is a *single* tag which is used to display a Submit button. Clicking the Submit button writes the comment information to the database. Used in the comment input form.

25.1 Attributes

This tag has no attributes.

25.2 Examples

25.2.1 Example 1: Comment Form

```
<table cellpadding="4" cellspacing="0" border="0">
  <tr>
    <td align="right" valign="top">
      <txp:text item="name" />
    </td>
    <td valign="top">
      <txp:comment_name_input />
    </td>
    <td valign="top" align="left">
      <txp:comment_remember />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="email" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_email_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      http://
    </td>
    <td valign="top" colspan="2">
      <txp:comment_web_input />
    </td>
  </tr>
  <tr>
    <td valign="top" align="right">
      <txp:text item="message" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_message_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top"> </td>
    <td valign="top" align="left">
      <txp:comments_help />
    </td>
    <td align="right" valign="top">
      <txp:comment_preview />
      <txp:comment_submit />
    </td>
  </tr>
</table>
```

Other tags used: [comment_email_input](#), [comment_name_input](#), [comment_web_input](#), [comment_preview](#), [comment_remember](#), [comments_help](#), [text](#)

26 Category:Comment Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Comment Tags are a subcategory of the [Tag Reference](#). They are tags that all relate to the visitor commenting system built into Textpattern. See the [Comments](#) page.

[Download Category:Comment_Tags book](#) (see [here](#) for more details)

27 comment time

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_time />
```

The ***comment_time*** tag is a *single* tag which is used to display the time and date the comment was submitted. Used in a comment form.

27.1 Attributes

This tag will accept the following attributes (**case-sensitive**):

format="value"

Override default date format, as set in preferences.
Values: any valid `strftime()` string or `since`.

27.2 Examples

27.2.1 Example 1: Comments Display Form

```
<txp:comment_message />
```

```
<p><small>&#8212; <txp:comment_name /> <txp:comment_time />  
<txp:comment_permalink>#</txp:comment_permalink></small></p>
```

Other tags used: [comment_name](#), [comment_message](#), [comment_permalink](#)

28 comment web

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

<txp:comment_web>

The **comment_web** tag can be used as either a *single* or a *container* tag. It is used to display (a link to) the commenter's web address, if entered at the time of posting.

When used as a container tag, it will turn the contents into a link to that web address. Otherwise, it will return the web address. Used in a comment form.

28.1 Attributes

This tag has no attributes.

28.2 Examples

28.2.1 Example 1: Comments Display Form with linked website and comment id

```
<txp:comment_message />
```

```
<p><small>&#8212; <a href="<txp:comment_web />"><txp:comment_web /></a> <txp:comment_time />
<txp:comment_permalink>#</txp:comment_permalink></small></p>
```

Other tags used: [comment_name](#), [comment_message](#), [comment_permalink](#), [comment_time](#)

28.2.2 Example 2: Container Example

```
<txp:comment_web>Website</txp:comment_web>
```

29 comment web input

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comment_web_input />
```

The `comment_web_input` tag is a *single* tag which is used to display a text entry field to accept the commenter's domain name. This tag can be used in page or form.

Function assumes `http://` for all URLs.

29.1 Attributes

This tag has no attributes.

29.2 Examples

29.2.1 Example 1: Comment Form

```
<table cellpadding="4" cellspacing="0" border="0">
  <tr>
    <td align="right" valign="top">
      <txp:text item="name" />
    </td>
    <td valign="top">
      <txp:comment_name_input />
    </td>
    <td valign="top" align="left">
      <txp:comment_remember />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="email" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_email_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      http://
    </td>
    <td valign="top" colspan="2">
      <txp:comment_web_input />
    </td>
  </tr>
  <tr>
    <td valign="top" align="right">
      <txp:text item="message" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_message_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top"> </td>
    <td valign="top" align="left">
      <txp:comments_help />
    </td>
    <td align="right" valign="top">
      <txp:comment_preview />
      <txp:comment_submit />
    </td>
  </tr>
</table>
```

Other tags used: [comment_email_input](#), [comment_message_input](#), [comment_preview](#), [comment_remember](#), [comment_submit](#), [comments_help](#), [text](#)

30 comments

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comments />
```

The **comments** tag is a *single* tag which is used to display the comments associated with a particular article. Comments will be displayed for the present individual article as a default, or to the article set by the "id" attribute.

30.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

sort="sort value(s)"

How to sort the resulting list.

Values:

discussid (comment ID)
parentid (article ID)
name
email
web
ip (IP address)
posted
message
rand() (random)

Default: posted asc.

form="form name"

Default is comments.

wraptag="tag text"

(X)HTML tag (without brackets) to wrap around list.

Default depends upon [preference setting](#), either `ol` or `unset`.

break="text"

(X)HTML tag (without brackets) or string used to separate comments.

Default depends upon [preference setting](#), either `li` or `div`.

class="class name"

CSS class attribute to be applied to `wraptag`.

Default: comments.

breakclass="class name"

CSS class attribute to be applied to `break` (when value supplied is a tag).

Default: unset.

offset="integer"

The number of comments to skip.

Default: 0.

limit="integer"

The number of comments to display.

Default: 0 (no limit).

30.1.1 Example 1: Display comments, and give an indication of *Comments* status

Comments for articles can be turned off or on at the author's discretion for any article that is published; by using the following scheme in an article form, you can still have the on/off control over comments while still giving users indication of comment status.

```
<txp:comments />
```

```
<txp:if_comments_allowed>
```

```
  <txp:comments_form />
```

```
<txp:else />
```

```
  <p>Comments are turned off for this article.</p>
```

```
</txp:if_comments_allowed>
```

Other tags used: [comments_form](#), [if_comments_allowed](#), [else](#)

30.1.2 Example 2: Conditional comments

Tags

```
<txp:if_comments_allowed>
  <txp:comments form="lineitem" breakclass="special" break="li" wraptag="ul" />

  <txp:comments_form />
</txp:if_comments_allowed>
```

Form (lineitem) (type: comment)

```
<small><txp:comment_id /></small>
```

Styles could go this way

```
.special
{
  display:list-item;
  list-style-type:none;
}
```

What it does...

For the article, list id numbers and a comment input form; but only if comments are currently allowed.

Other tags used: [comment_id](#), [comments_form](#), [if_comments_allowed](#)

31 comments count

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comments_count />
```

The **comments_count** tag is a *single* tag which is used to display the number of comments associated with a particular article.

Though `comments_count` can be used independently, it is also called by `comments_invite` to append the comments count to the `comments_invite` link. Used in an article form.

31.1 Attributes

This tag has no attributes.

31.2 Examples

31.2.1 Example 1: Display comment invitation and count

But only if any comments are associated with the current article.

```
<txp:if_comments>
  <p><txp:comments_invite showcount="0" /> <txp:comments_count /> people have already responded.</p>
</txp:if_comments>
```

Other tags used: [comments_invite](#), [if_comments](#)

32 comments error

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comments_error />
```

The **comments_error** tag is a *single* tag which is used to produce the current comments error.

32.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

wraptag="tag text"

(X)HTML tag (without brackets) to wrap around the list.

Default is unset.

break="value"

(X)HTML tag (without brackets) or string used to separate list items.

Default is `br`.

class="class name"

CSS class to be applied to `wraptag`.

Default is `comments_error`.

32.2 Examples

32.2.1 Example 1: Display comments error when an error exists

```
<txp:if_comments_error>
  <txp:comments_error break="li" wraptag="ul" />
</txp:if_comments_error>
```

Other tags used: [if_comments_error](#)

33 comments form

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comments_form />
```

The **comments_form** tag is a *single* tag which is used to display a comment form. Comments will be attached to present individual article as a default, or to the article set by the "id" attribute.

33.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

isize="integer"

(X)HTML *size* attribute to be applied to the (X)HTML form input.

Default: 25.

msgcols="integer"

(X)HTML *cols* attribute to be applied to (X)HTML form textarea output.

Default: 25.

msgrows="integer"

(X)HTML *rows* attribute to be applied to (X)HTML form textarea output.

Default: 5.

msgstyle="value"

CSS inline *style* attribute to be applied to (X)HTML form textarea output. Recommended that you use CSS via textarea's class or id attribute instead.

form="form name"

Use specified form.

Default: comment_form.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around output.

Default: unset.

class="CSS class name"

CSS class attribute to be applied to wraptag.

Default: comments_form.

33.2 Examples

33.2.1 Example 1: Give visitors indication of *Comments* status

Comments for articles can be turned off or on at the author's discretion for any article that is published; by using the following scheme in an article form, you can still have the on/off control over comments while still giving users indication of comment status.

```
<txp:if_comments_allowed>
  <txp:comments_form />
<txp:else />
  <p>Comments are turned off for this article.</p>
</txp:if_comments_allowed>
```

Other tags used: [if_comments_allowed](#), [else](#)

33.2.2 Example 2: Text area changes in preview

Using some conditional tags the size of the comment input text area can be changed in the preview.

```
<txp:if_comments_preview>
  <txp:comments_preview form="comments" />
  <p style="color:red;">This is just a preview of your comment!</p>
  <txp:comments_form isize="30" msgcols="55" msgrows="5" />
<txp:else />
  <txp:if_comments_allowed>
    <txp:comments_form isize="30" msgcols="55" msgrows="15" />
  </txp:if_comments_allowed>
  <p>Comments are turned off for this article.</p>
</txp:if_comments_preview>
```

Other tags used: [comments_preview](#), [if_comments_allowed](#), [if_comments_preview](#), [else](#)

33.2.3 Example 3: Display conditional comments and form

Tags

```
<txp:if_comments_allowed>
  <txp:comments form="lineitem" break="li" wraptag="ul" breakclass="special" />
  <txp:comments_form />
</txp:if_comments_allowed>
```

Form (lineitem) Type(comment)

```
<small><txp:comment_id /></small>
```

Styles could go this way

```
.special
{
  display:list-item;
  list-style-type:none;
}
```

What it does...

For the current article, returns a list of id numbers for comments and a comment input form, but only if comments are currently allowed.

Other tags used: [comment_id](#), [comments](#), [if_comments_allowed](#)

33.3 Genealogy

33.3.1 Version 4.0.4

- [show_preview](#) removed

34 comments help

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comments_help />
```

The **comments_help** tag is a *single* tag which is used to display a Textile help link. This tag can be used in a page or a form.

34.1 Attributes

This tag has no attributes.

34.2 Examples

34.2.1 Example 1: Comment Form

```
<table cellpadding="4" cellspacing="0" border="0">
  <tr>
    <td align="right" valign="top">
      <txp:text item="name" />
    </td>
    <td valign="top">
      <txp:comment_name_input />
    </td>
    <td valign="top" align="left">
      <txp:comment_remember />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      <txp:text item="email" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_email_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top">
      http://
    </td>
    <td valign="top" colspan="2">
      <txp:comment_web_input />
    </td>
  </tr>
  <tr>
    <td align="right" align="right">
      <txp:text item="message" />
    </td>
    <td valign="top" colspan="2">
      <txp:comment_message_input />
    </td>
  </tr>
  <tr>
    <td align="right" valign="top"> </td>
    <td valign="top" align="left">
      <txp:comments_help />
    </td>
    <td align="right" valign="top">
      <txp:comment_preview />
      <txp:comment_submit />
    </td>
  </tr>
</table>
```

Other tags used: [comment_email_input](#), [comment_name_input](#), [comment_web_input](#), [comment_preview](#), [comment_remember](#), [comment_submit](#), [text](#)

35 comments invite

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comments_invite />
```

The **comments_invite** tag is a *single* tag which is used to display a link to an article comment form. Text used for the link will be taken from the invitation field on the "write" screen.

This tag can be used in both Page templates and [Forms](#).

35.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

showcount="boolean"

Whether to display comment count.
Values: 1 (yes), 0 (no).
Default: 1.

textonly="boolean"

Whether to display invite as text, rather than a hyperlink.
Values: 1 (yes), 0 (no).
Default: 0.

showalways="boolean"

Whether to display invite on individual article page.
Values: 1 (yes), 0 (no).
Default: 0.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around invite text.
Default: unset.

class="class name"

CSS class name to be applied to wraptag.
Default is comments_invite.

35.2 Examples

35.2.1 Example 1: Display comments invitation and comment count

But only if there are any comments associated with the current article.

```
<txp:if_comments><p><txp:comments_invite /></p></txp:if_comments>
```

Other tags used: [if_comments](#)

36 comments preview

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:comments_preview />
```

The **comments_preview** tag is a *single* tag which is used to display a preview of a visitor's comment.

36.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

form="value"

Default is comments.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around the list.

Default depends upon [preference setting](#), either `ol` or `unset`.

class="class name"

CSS class name to be applied to `wraptag`.

Default is `comments_preview`.

36.2 Genealogy

36.2.1 Version 4.0.4

- Use is necessary in comments display form (`comments_display`, by default).

36.2.2 Version 4.0.3

- Support added.

37 Category:Conditional Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Conditional Tags are a subcategory of the [Tag Reference](#). They are tags that are used to take action depending if a condition is met.

Most tags beginning `if_...` optionally take the [else](#) tag to allow you to take some action if the condition does not meet the given criteria.

[Download Category:Conditional_Tags book](#) (see [here](#) for more details)

38 CSS

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:css />
```

The basic **css** tag is a *single* tag and used to output the URL of the style sheet assigned in the [Sections](#) tab.

38.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

n="style name"

Link to specified style.

format="value"

How to format output: either return complete (X)HTML link tag with necessary (X)HTML attributes, or only the StyleSheet's URL. Available values are `link` or `url`. Default is `url`.

rel="value"

(X)HTML *rel* attribute to be applied to link tag (when invoked with *format*="link"). Default is `stylesheet`.

media="value"

(X)HTML *media* attribute to be applied to link tag (when invoked with *format*="link"). Default is `screen`.

title="value"

(X)HTML *title* attribute to be applied to link tag (when invoked with *format*="link"). Default is `unset`.

38.2 Examples

38.2.1 Example 1: Output the URL to the section's default style sheet

```
<head>
<txp:css />
</head>
```

38.2.2 Example 2: Output the URL to a named style sheet

```
<head>
<txp:css n="style_name" />
</head>
```

38.2.3 Example 3: Output print and alternate style sheets

```
<head>
<txp:css n="plain" rel="alternate" title="Plain and Simple Style" />
<txp:css n="glossy" rel="alternate" title="Glossy Style"/>
<txp:css n="print" media="print" />
</head>
```

38.3 Genealogy

38.3.1 Version 4.0.4

- *format*, *media*, *rel*, *title* added.

39 custom field

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:custom_field />
```

The basic *custom_field* tag is a *single* tag and used to display the contents of a custom field.

Custom fields are useful if you publish content that has a consistent structure. They are defined in the [Advanced Preferences](#) tab, under the [Custom Fields](#) heading. You may have up to and including 10 fields. With a default Textpattern installation, you will just see two custom fields with the names "custom1" and "custom2", which you may rename or remove.

The contents of custom fields are limited to 255 characters in length.

Custom names may include letters (uppercase or lowercase), numbers and under_scores, but no spaces or other special characters.

There are certain names **reserved** by Textpattern, which should not be used to name custom fields:

- thisid
- posted
- annotate
- comments_invite / comments_count
- authorid
- title / url_title
- category / category1 / category2
- section
- keywords
- article_image
- body / excerpt
- status
- sort / offset / limit / form, or any other [tag attribute](#), just in case

A symptom of a name clash is when you go to check or display the contents of a custom field and receive unexpected (or no) output. Check your names aren't already used by Textpattern in the first instance as it might save you a headache.

To remove a custom field, simply clear its name. **Don't forget to save your changes.**

Once defined, you may set values for your custom fields for each post (under "advanced options" in the [Write](#) tab), and may display custom field information in any article form.

Custom fields can save a lot of work, and to make them even more useful there is the [if_custom_field](#) conditional tag to complement them. More explanation and examples on that page.

39.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

```
name="fieldname"  
    Display specified custom field.  
default="value"  
    Default value to use when field is empty.  
escape="html"  
    Escape HTML entities prior to echoing the field contents.  
    Values: html or unset  
    Default: html
```

39.2 Examples

39.2.1 Example 1: Book Reviews

You might, for example, publish book reviews for which you add the author, the title of the book, the publishing company and the year of publication.

With:

- a custom field named "Book_Author" containing J.R.R. Tolkien

- a custom field named "Book_Title" containing The Lord of the Rings
- a custom field named "Book_Publisher" containing HarperCollins
- a custom field named "Book_Year" containing 2004

and an article form like the following:

```
<p><txp:custom_field name="Book_Author" />: <txp:custom_field name="Book_Title" /><br />
  Published by <txp:custom_field name="Book_Publisher" /> in <txp:custom_field name="Book_Year" />.</p>
```

(X)HTML returned would be:

```
<p>J.R.R. Tolkien: The Lord of the Rings<br />
  Published by HarperCollins in 2004.</p>
```

39.2.2 Example 2: Power A Linklog

This works well with [variation of Sencer's Txp bookmarklet](#).

With an article title of Textpattern, an excerpt of Textpattern is awesome., a custom field named "Link" containing <http://textpattern.com/>, and an article form like the following:

```
<div class="linklog-entry">
  <div style="float: left;"><a href="<txp:custom_field name="Link" />"><txp:title /></a></div>
  <div style="float: right;"><txp:posted format="%d %d %Y" /></div><br>
  <txp:excerpt />
</div>
```

(X)HTML returned would be:

```
<div class="linklog-entry">
  <div style="float: left;"><a href="http://textpattern.com/">Textpattern</a></div>
  <div style="float: right;"><txp:posted format="08 Aug 2005" /></div>
  <p>Textpattern is awesome.</p>
</div>
```

Other tags used: [title](#), [posted](#), [excerpt](#)

39.2.3 Example 3: Unescaping (X)HTML output

With a custom field named "foo" containing:

```
<a href=" ../here/">
```

using the following:

```
<txp:custom_field name="foo" />
```

will return this hunk of (X)HTML:

```
&#60;a href=&#34; ../here/&#34;&#62;
```

whereas using:

```
<txp:custom_field name="foo" escape="" />
```

will render the URL as you'd expect, exactly as written in the custom field itself. Thus, it will be rendered as a link by the browser.

39.3 Genealogy

39.3.1 Version 4.0.7

- default value for attribute `escape` changed from `unset` to `html`

40 else

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:else />
```

The **else** tag is a *single* tag that is used within a containing conditional tag to provide the means to assign default, or alternative, behavior when the condition in the surrounding tag is *not* met.

Visually, this is the general structure in which it is used:

```
<txp:if_conditional_tag>
  ...Content if true...
<txp:else />
  ...Content if not true...
</txp:if_conditional_tag>
```

40.1 Attributes

This tag has no attributes.

40.2 Examples

40.2.1 Example 1: Display excerpt when available

```
<txp:if_excerpt>
  And Furthermore &#183; <txp:excerpt />
<txp:else />
  <txp:section link="1" />
</txp:if_excerpt>
```

What this does...

When the excerpt is available it is displayed, but when it is missing a hyperlinked section name is displayed instead.

Other tags used: [excerpt](#), [if_excerpt](#), [section](#)

41 email

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

<txp:email>

The **email** tag is both a *single* tag and a *container* tag. Textpattern will replace it with a `mailto:` email link, according to the attributes set.

41.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

email="value"

The e-mail address from which to make the link.

Value: Any valid email address.

Default: unset.

linktext="value"

The displayed link text.

Value: Any text.

Default: Contact.

title="value"

The title attribute to assign to the link.

Value: Any valid (X)HTML title.

Default: unset.

41.2 Examples

41.2.1 Example 1: Simple e-mail link

```
<txp:email email="john.doe@example.com" linktext="Contact me" title="Send me an Email" />
```

41.2.2 Example 2: Pre-populate message subject and body

```
<txp:email email="john.doe@example.com?subject=Lorem Ipsum&body=Sit amend" />
```

41.2.3 Example 3: Container example

```
<txp:email email="foo@example.com" title="Email us!"></txp:email>
```

41.3 Genealogy

41.3.1 Version 4.0.5

- Can be used as a container.

42 Category:Error Handling Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Error Handling Tags are a subcategory of the [Tag Reference](#). They are tags that allow you to deal with or otherwise intercept/display the result of exceptional circumstances in the page flow.

[Download Category:Error_Handling_Tags book](#) (see [here](#) for more details)

43 error message

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:error_message />
```

The **error_message** tag is a *single* tag that Textpattern will replace with the error message text for the error status as set by the server. Should be used in an error_xxx or error_default [page template](#).

43.1 Attributes

This tag takes no attributes.

43.2 Examples

43.2.1 Example 1: Display error information

```
<h3><txp:error_status /></h3>  
<p><txp:error_message /></p>
```

What this does...

With the tags arranged like this (as they are in the error_default page template), they display the error status code as a header and the relevant server message beneath it, usually to indicate to the visitor that something went wrong.

Other tags used: [error_status](#)

44 error status

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:error_status />
```

The **error_status** tag is a *single* tag that Textpattern will replace with the error status as set by the server. Should be used in an error_XXX or error_default [page template](#).

44.1 Attributes

This tag takes no attributes.

44.2 Examples

44.2.1 Example 1: Display error information

```
<h3><txp:error_status /></h3>  
<p><txp:error_message /></p>
```

What this does...

With the tags arranged like this (as they are in the error_default page template), they display the error status code as a header and the relevant server message beneath it, usually to indicate to the visitor that something went wrong.

Other tags used: [error_message](#)

45 excerpt

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:excerpt />
```

The **excerpt** tag is a *single* tag which is used to return the excerpt text, if any, associated with the article being displayed.

45.1 Attributes

This tag has no attributes.

45.2 Related Tags

The conditional tag [if_excerpt](#) can be used to check if there is an excerpt.

45.3 Examples

45.3.1 Example 1: Excerpt and 'read more' button

This example explains how you could display the excerpt in an article list, and excerpt + body in an individual article. Use the following in an article form:

```
<txp:if_article_list>
  <txp:if_excerpt>
    <txp:excerpt />
    <p class="read-more">
      <a href="<txp:permalink />#body"
        title="<txp:title />">&#187; Read more</a>
    </p>
  <txp:else />
  <txp:body />
</txp:if_excerpt>
<txp:else />
  <txp:if_excerpt>
    <txp:excerpt />
  </txp:if_excerpt>
  <div id="body">
    <txp:body />
  </div>
</txp:if_article_list>
```

Other tags used: [body](#), [if_article_list](#), [if_excerpt](#), [permalink](#), [title](#)

45.3.2 Example 2: Display the excerpt text or a default link

Use the following within an article form:

```
<txp:if_excerpt>
  <txp:excerpt />
<txp:else />
  <p>Section: <txp:section title="1" link="1" /></p>
</txp:if_excerpt>
```

Other tags used: [if_excerpt](#), [section](#)

46 expires

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:expires />
```

A single tag used to indicate when an article should no longer appear in a site, particularly when the information is date sensitive (e.g., events like conferences, meetings and so forth). The tag is defined by expiration date values that are set under the **More** section of the [Write](#) admin-side panel.

For more on this tag see its announcement, [So, you'd like to stick a ?Best Before? label on those articles?](#)

46.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

format="format string"

Override the default date format set in the preferences.
Values: any valid `strftime()` string values, or `since`.
Default: unset (date format set via preferences).

gmt="boolean"

Return either local time according to the set time zone preferences or GMT time.
Default: 0 (local time set in preferences).

lang="ISO language code"

Format time string suitable for the specified language (locale).
Values: locales adhere to [ISO-639](#).
Default: unset (time format set via preferences).

46.2 Examples

46.2.1 Example 1: Custom format date setting

```
<p>Expires: <txp:expires format="%b %d, %Y" /></p>
```

would result in:

```
<p>Expires: Sep 10, 2010</p>
```

46.3 Genealogy

46.3.1 Version 4.0.7

- expires tag added.

46.4 Related

Related tags are:

- [if_expires](#)
- [if_expired](#)

47 feed link

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

<txp:feed_link>

The **feed_link** tag can be used as either a *single* or *container* tag and is used to output an XHTML link to the site's articles' RSS feed. When used as a container tag, it will turn the contents into a link to the feed, otherwise the value of attribute "label" will be used as link text. Should be used in a page.

47.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

flavor="value"

Whether to output a link to the RSS or Atom version of the feed. Available values: `rss` or `atom`.

Default: `rss`.

format="value"

Whether to output a (X)HTML `a` tag or a (X)HTML `link` tag.

Values: `a` or `link`.

Default: `a`.

section="section name"

Restrict to specified section.

Default: current section.

category="category name"

Restrict to specified category. Note: the category name may be different to the Title you typed when you created the category, as the names are sanitized for URL use. Check the [Categories](#) tab to ensure you are using the correct name.

Default: current category.

limit="integer"

Number of articles to display in the feed.

Default: depends upon [preference setting](#).

title="value"

(X)HTML `title` attribute.

Default: depends upon *flavor* used, either `RSS feed` or `Atom feed`.

label="value"

Link text for feed link when using *format* of `a`.

wraptag="tag"

(X)HTML `tag` (without brackets) to wrap around output when using *format* of `a`.

47.2 Examples

47.2.1 Example 1: Display an RSS feed link for specific section and category

```
<txp:feed_link flavor="rss" section="about" category="general" label="XML" wraptag="p" />
```

47.2.2 Example 2: Container example

```
<txp:feed_link wraptag="p"></txp:feed_link>
```

47.3 Genealogy

47.3.1 Version 4.0.4

- `format` added.

48 file download

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download />
```

The **file_download** tag is a *single* tag which Textpattern will replace with a file download [form](#). Inside that form go the other.

48.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

form="name"

Use the specified form.

Default is files.

id="integer"

File id of the file to link to.

Default is unset; nothing is returned.

filename="name"

Filename of the file to link to.

Default is unset; nothing is returned.

48.2 Examples

48.2.1 Example 1: Display a download form

```
<txp:file_download form="files" id="1" />
```

48.2.1.1 Default "files" form

```
<txp:text item="file" />:
<txp:file_download_link>
<txp:file_download_name /> [<txp:file_download_size format="b" decimals="2" />]
</txp:file_download_link>
<br />
<txp:text item="category" />: <txp:file_download_category /><br />
<txp:text item="download" />: <txp:file_download_downloads />
```

Other tags used: [file_download_category](#), [file_download_downloads](#), [file_download_link](#), [file_download_name](#), [file_download_size](#), [text](#)

49 file download category

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_category />
```

The **file_download_category** tag is a *single* tag that Textpattern will replace with the category of the file to download. Should be used in a [download form](#).

49.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

title="boolean"

Whether to display the category name or its title

Values: 0 (name), or 1 (title).

Default: 0.

wraptag="tag text"

HTML tag to be used to wrap the category with, specified without brackets.

Default: unset.

class="class name"

CSS class attribute to apply to the category wraptag.

49.2 Examples

49.2.1 Example 1: Display a category name following "category:"

```
<txp:text item="category" />: <txp:file_download_category />
```

Other tags used: [text](#)

50 file download created

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_created />
```

The **file_download_created** tag is a *single* tag that Textpattern will replace with the upload date of the file to download. Should be used in a [download form](#).

50.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

format="format string"

Override the default Archive date format set in the [Basic Preferences](#).

Values: any valid [strftime\(\)](#) string values.

50.2 Examples

50.2.1 Example 1: Display formatted file upload date

```
<txp:file_download_created format="%c" />
```

51 file download description

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_description />
```

The **file_download_description** tag is a *single* tag which Textpattern will replace with the description of the file to download, as defined when the file was uploaded. Should be used in a [download form](#).

51.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

escape="html"

Escape HTML entities such as <, > and & for the file's description attribute.

Values: html or unset

Default: html.

wraptag="tag text"

HTML tag to wrap description text with. Specify it without brackets.

Default: unset.

class="class name"

CSS class attribute to apply to the wraptag surrounding the description text.

51.2 Examples

51.2.1 Example 1: Display a file s description following "description:"

```
<txp:text item="description" />: <txp:file_download_description />
```

Other tags used: [text](#)

51.3 Genealogy

51.3.1 Version 4.0.7

- default value for attribute `escape` changed from `unset` to `html`

52 file download downloads

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_downloads />
```

The ***file_download_downloads*** tag is a *single* tag that Textpattern will replace with the number of times the current file has been downloaded. Should be used in a [download form](#).

52.1 Attributes

This tag has no attributes.

52.2 Examples

52.2.1 Example 1: Display the number of downloads following "downloads:"

```
<txp:text item="downloads" />: <txp:file_download_downloads /><code>
```

Other tags used: [text](#)

53 file download id

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_id />
```

The ***file_download_id*** tag is a *single* tag that Textpattern will replace with the internal ID number of the file to be downloaded. Should be used in a [download form](#).

53.1 Attributes

This tag has no attributes.

53.2 Examples

53.2.1 Example 1: Display a file id following "File number:"

```
<txp:text item="File number" />: <txp:file_download_id />
```

Other tags used: [text](#)

54 file download link

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_link />
```

The **file_download_link** tag is both a *single* tag and a *container* tag. Thus it may be used as an opening and closing pair:

```
<txp:file_download_link>
...containing statements...
</txp:file_download_link>
```

When used as a single tag, Textpattern will replace the tag with a download link to the file being downloaded. As a container, it will assign the link to the given text or tag, while the single tag outputs the file's plain URL.

54.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

id="integer"
Numeric id of the file to download.

filename="text"
Name of the file to download.

Note: "id" takes precedence over "filename". If neither is defined and the tag is not used within the context of a file, nothing is returned.

54.2 Examples

54.2.1 Example 1: Provide a link to download file #4

```
<txp:file_download_link id="4">
<txp:file_download_name /> [<txp:file_download_size format="mb" decimals="2" />]
</txp:file_download_link>
```

What this does...

Makes a link to the given file (#4) comprising its file name and size

Other tags used: [file_download_name](#), [file_download_size](#)

55 file download list

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_list />
```

The **file_download_list** tag is a *single* or a *container* tag which is used to produce a list of download links according to the given attributes. Each file in the list is formatted by the file tags used in the given form (default is the `files` form).

If used as a container, it must be specified as an opening and closing pair of tags, like this:

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

55.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

category="category name"

Restrict to files from the specified category. Allows a comma separated list of category names. Note: category names may be different to the Title you typed when you created the category, as the names are sanitized for URL use. Check the [Categories](#) tab to ensure you are using the correct names
Default: unset.

status="file status"

Restrict to files with the specified status.
Values: hidden, pending, live.
Default: live.

sort="by what and order"

How to sort the resulting list.
Values: id, filename, category, description, downloads, created, modified, rand() ([random](#)). Adding a space and then one of either `asc` or `desc` orders by ascending or descending value, respectively.
Default: filename asc.

limit="integer"

Number of files to display.
Default is 10.

offset="integer"

Number of files to skip.
Default: unset.

form="form name"

Use the specified form to process the files.
Default: `files`.

label="text"

Label for the top of the list.
Default: unset.

labeltag="tag text"

The (X)HTML tag, without brackets, to wrap around the label.
Default: unset.

wraptag="tag text"

The (X)HTML tag, without brackets, to wrap around the file list.
Default: unset.

break="tag text"

The (X)HTML tag (without brackets) or string to separate list items.
Default: `br`.

class="class name"

The CSS class attribute to be applied to the `wraptag`.
Default: `file_download_list`.

id="file ID"

Display the specific file or list of files.
Value: (comma separated list of) file ID(s).
Default: unset.

55.2 Examples

55.2.1 Example 1: Display the ten most popular downloads

```
<txp:file_download_list limit="10" break="li" wraptag="ul" sort="downloads desc" />
```

Styles could go this way

```
.file_download_list  
{  
    list-style-type:none;  
}
```

55.3 Genealogy

55.3.1 Version 4.2.0

- New attribute id.

55.3.2 Version 4.0.7

- Can be used as a container tag.

55.3.3 Version 4.0.6

- support added for comma separated list for category attribute

56 file download modified

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_modified />
```

The `[[file_download_modified]]` tag is a *single* tag that Textpattern will replace with the last modified date of the file to download. Should be used in a [download form](#).

56.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

format="format string"

Override the default Archive date format set in the [Basic Preferences](#).

Values: any valid `strftime()` string values.

56.2 Examples

56.2.1 Example 1: Display formatted file modified date

```
<txp:file_download_modified format="%c" />
```

57 file download name

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_name />
```

The **`file_download_name`** tag is a *single* tag that Textpattern will replace with the name of the file to download. Should be used in a [download form](#) or within a [file_download_link](#) tag.

57.1 Attributes

This tag has no attributes.

57.2 Examples

57.2.1 Example 1: Display the name of a file, linked to download

```
<txp:file_download_link filename="my_presentation.pdf">
<txp:file_download_name /> [<txp:file_download_size format="mb" decimals="2" />]
</txp:file_download_link>
```

Other tags used: [file_download_link](#), [file_download_size](#)

Note: `id` takes precedence over `filename`. If neither is defined and the tag is not used within a file context, nothing is returned.

58 file download size

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:file_download_size />
```

The **file_download_size** tag is a *single* tag which Textpattern will replace with the formatted file size of the file to be downloaded. Should be used in a [download form](#).

58.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

format="numbering style"

The way to represent the number, based on the file's expected size.

Valid options are **b** (bytes), **kb** (kilobytes), **mb** (megabytes), **gb** (gigabytes), **tb** (terabytes), or **pb** (petabytes).

Default: **mb**.

decimals="places"

Number of decimal places to format the value to.

Default: **2**.

58.2 Examples

58.2.1 Example 1: Display formatted file size in kilobytes

```
<txp:file_download_size format="kb" decimals="2" />
```

59 Category:File Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

File Tags are a subcategory of the [Tag Reference](#). They are tags that are used to display files that are managed through the [Files](#) Tab.

Download [Category:File_Tags](#) book ([see here](#) for more details)

60 Category:Future Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Tags listed here are not yet in use, but are under development and anticipated in future Txp versions. See [Tags In Development](#) for more details.

Download [Category:Future_Tags](#) book (see [here](#) for more details)

61 hide

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:hide>
```

The **hide** tag is a *container* tag which is used to suppress the interpretation of all enclosed contents. Use it for comments, temporary concealment of article text parts or non-destructive form changes.

61.1 Attributes

This tag has no attributes.

61.2 Examples

61.2.1 Example 1: Insert a useful note in a template

```
<txp:hide>This is essential as a work-around for the Peekaboo  
bug in Internet Explorer 6</txp:hide>
```

61.2.2 Example 2: Comment out part of a form for testing

If you want to try something out to see how it affects the layout without actually deleting the content, wrap it in **hide** tags:

```
<div class="entry-content">  
  <txp:body />  
</div>  
  
<txp:hide>  
  
<address class="vcard author">  
  <span class="fn"><txp:author /></span>  
</address>  
<txp:comments_invite wraptag="p" />  
  
</txp:hide>
```

What this does...

Renders the body text inside the `entry-content` div but skips the `address` and `comments_invite` tags.

Other tags used: [body](#), [author](#), [comments_invite](#)

62 if article author

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_article_author>
```

The ***if_article_author*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_article_author>
...conditional statement...
</txp:if_article_author>
```

The tag will execute the contained statement if the author name associated with a particular article matches the value of the name attribute. Should be used in an article form.

The *name* attribute requires an author's login name not their Real Name

62.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="author"

Comma-separated list of author (login) names.

Default: unset (i.e. any author at all)

62.2 Examples

62.2.1 Example 1: Display some text dependent on an article's author

```
<txp:if_article_author name="admin">
  <p>Publisher</p>
</txp:if_article_author>
```

What this does...

Displays the text "Publisher" if the article was written (posted) by the author "admin".

63 if article category

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_article_category>
```

The ***if_article_category*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_article_category>
...conditional statement...
</txp:if_article_category>
```

It will execute the contained statement if the category name associated with a particular article (Category1 or Category2) matches the values of the name and number attributes. Should be used in an article [form](#).

63.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="category"
Comma-separated list of category names to match.
Default: unset.

number="number"
match category in Cat1 or Cat2 (or both).
Values: 1 or 2
Default: unset, causing both categories to be matched against the specified name.

63.2 Examples

63.2.1 Example 1: Display matched category

```
<txp:if_article_category name="Prose" number="1">
  <p><txp:category1 /></p>
</txp:if_article_category>
```

What this does...

If the Category1 assigned to the article is "Prose", the category is displayed.

Other tags used: [category1](#)

63.2.2 Example 2: Using the tag with **else**

```
<txp:if_article_category name="Prose" number="1">
  <p>Fun With Prose</p>
<txp:else />
  <p><a href="index.php">Home</a></p>
</txp:if_article_category>
```

What this does...

Displays the welcome text if the category and category number match the given values, or shows a default link otherwise.

Other tags used: [else](#)

63.2.3 Example 3: Display a list of matching links

In an article form, put the following set of conditionals for each category you want to look for:

```
<txp:if_article_category name="yourcategory" number="1">
<ul>
  <txp:article_custom form="sub" category="yourcategory" sortby="Posted" sortdir="asc" />
</ul>
</txp:if_article_category>
```

And then your article form (in this case called 'sub') can be used to list links to other articles like this:

```
<li><txp:permalink><txp:title /></txp:permalink></li>
```

What it does...

Lists articles of the same category as the current article's Category1

Other tags used: `article_custom`, `else`, `title`, `permalink`

64 if article id

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_article_id>
```

The ***if_article_id*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_article_id>
...conditional statement...
</txp:if_article_id>
```

The tag will execute the contained statement if the article id associated with a particular article matches the id attribute. Should be used in an article form/container. The *id* attribute *must* be used in an *article list context* (when producing a page that displays more than one article) or the tag will do nothing.

64.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

id="integer"

Comma delimited integer article ID list.

Default: current article's ID if available (i.e. on a page that displays a single article).

64.2 Examples

64.2.1 Example 1: Display info if the article id matches

```
<txp:if_article_id id="33">
  <p><txp:title /></p>
</txp:if_article_id>
```

What this does...

Displays the article title if the id of the current article is 33.

Other tags used: [title](#)

64.3 Genealogy

- As of September 2009 (version 4.2.0), there is a [motion](#) to simplify the semantics of this tag in the case of undefined *id* attribute.

64.3.1 Version 4.0.7

- Defaults to the current article's ID.

65 if article image

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_article_image>
```

The ***if_article_image*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_article_image>  
...conditional statement...  
</txp:if_article_image>
```

The tag will execute the contained statements if an image is associated (through the [article image field](#)) with the article being displayed.

65.1 Attributes

This tag has no attributes.

65.2 Examples

65.2.1 Example 1: Display default image if no article image exists

```
<txp:if_article_image>  
  <txp:article_image />  
<txp:else />  
  <txp:image id="5" />  
</txp:if_article_image>
```

Other tags used: [article_image](#), [else](#), [image](#)

65.3 Genealogy

65.3.1 Version 4.2.0

- Added as a new tag

66 if article list

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_article_list>
```

The **if_article_list** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_article_list>
...conditional statement...
</txp:if_article_list>
```

The tag will execute the contained statement if an article list is being displayed (i.e. not showing an individual article).

66.1 Attributes

This tag has no attributes.

66.2 Examples

66.2.1 Example 1: Article/Article List Navigation

This example shows how to setup article navigation so that **prev-next** is used at the individual article level **or older-newer** with article lists.

```
<txp:article />

<txp:if_individual_article>
  <p>
    <txp:link_to_prev><txp:prev_title /></txp:link_to_prev>
    <txp:link_to_next><txp:next_title /></txp:link_to_next>
  </p>
</txp:if_individual_article>

<txp:if_article_list>
  <p>
    <txp:older>Previous</txp:older>
    <txp:newer>Next</txp:newer>
  </p>
</txp:if_article_list>
```

Other tags used: [link_to_prev](#), [link_to_next](#), [prev_title](#), [next_title](#), [if_individual_article](#), [older](#), [newer](#),

66.2.2 Example 2: In Combination with the **else** Tag

This example shows the **if_article_list** in combination with **else** to display a site's **site_name** or logo when an article list is displayed or not, respectively.

```
<txp:if_article_list>
  <p><txp:site_name /></p>
<txp:else />
  <p></img></p>
</txp:if_article_list>
```

Other tags used: [else](#), [site_name](#)

67 if article section

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_article_section>
```

The ***if_article_section*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_article_section>
...conditional statement...
</txp:if_article_section>
```

The tag will execute the contained statements if the section name associated with a particular article matches the value of the `name` attribute. Should be used in an article form.

67.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

`name="section"`
Comma-separated list of section names.

67.2 Examples

67.2.1 Example 1: Check the article's section

```
<txp:if_article_section name="Poetry">
<p>by <txp:author /></p>
</txp:if_article_section>
```

What this does...

Displays the author name if the current article belongs to the section named `Poetry`.

Other tags used: [author](#)

67.2.2 Example 2: Using the tag with **else**

```
<txp:if_article_section name="Poetry">
<p>Fun With Poetry</p>
<txp:else />
<p><a href="index.php">Home</a></p>
</txp:if_article_section>
```

What this does...

Display the welcome text if the article's section matches `Poetry`, or shows a default link otherwise.

Other tags used: [else](#)

68 if author

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_author>
```

The ***if_author*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_author>
...conditional statement...
</txp:if_author>
```

The tag will execute the contained statement if the called page is the result of an article search by a specific author's name.

This is *not* the same as checking if the current article was written (posted) by the given author. Use [if_article_author](#) for that situation

68.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="author"

Comma-separated list of author names.

Default is unset, which determines whether *any* author listing is being viewed.

68.2 Examples

68.2.1 Example 1: Select a stylesheet based on author

Selects a stylesheet named "author_list" when a list by author "admin" is being displayed, or a stylesheet determined by the active section for normal page display.

```
<txp:if_author name="admin">
<link rel="stylesheet" href="<txp:css n="author_list" />" type="text/css" />
<txp:else />
<link rel="stylesheet" href="<txp:css />" type="text/css" />
</txp:if_author>
```

Other tags used: [else](#), [css](#)

69 if category

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_category>
```

The **if_category** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_category>
...conditional statement...
</txp:if_category>
```

The tag will execute the contained statements if the `name` attribute matches a category search value, or the list is an article list by category (`?c=category`)

Should be used in a page template; if checking the category in an article form, use [if_article_category](#).

69.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

`name="category"`

Comma-separated list of category names. Note the category name is specified in lower case regardless of how you typed its title in the Category tab. Also note that if you had called your category **My Category Name** it becomes **my-category-name** when used in tags. Default is unset, which determines whether *any* category listing is being viewed.

69.2 Examples

69.2.1 Example 1: Display info depending on list category

```
<txp:if_category name="prose">
<p><txp:author /></p>
</txp:if_category>
```

What this does...

Displays the author's name if the article list is of category "Prose".

Other tags used: [author](#)

69.2.2 Example 2: Use tag with **else**

```
<txp:if_category name="prose">
  <p><txp:category /></p>
<txp:else />
  <h3><txp:site_name /></h3>
</txp:if_category>
```

What this does...

Displays the category name if the article list is of category "Prose", otherwise show the site's name.

Other tags used: [category](#), [else](#), [site_name](#)

69.2.3 Example 3: Display an appropriate heading

```
<txp:if_category>
  <h3>Articles in category <txp:category title=1 />:</h3>
<txp:else />
  <h3>All articles:</h3>
</txp:if_category>
```

What this does...

Displays an appropriate heading for both category and non-category pages.

Other tags used: [category](#), [else](#)

69.2.4 Example 4: Display a category / article list

Given the defined article categories: Prose, Poetry, and Opinions.

```
<txp:category_list label="Category Navigation" wraptag="p" class="list" />

<txp:if_category name="prose">
<txp:recent_articles label="Prose" limit="25" break="br" wraptag="p" class="list" category="prose" />
</txp:if_category>

<txp:if_category name="poetry">
<txp:recent_articles label="Poetry" limit="25" break="br" wraptag="p" class="list" category="poetry" />
</txp:if_category>

<txp:if_category name="opinions">
<txp:recent_articles label="Opinions" limit="25" break="br" wraptag="p" class="list" category="opinions" />
</txp:if_category>
```

What this does...

Shows a category list and, underneath it, a list of related articles in the currently selected category. Changing the category using the list changes the related articles underneath.

Styles could go this way:

```
p.list
{
    font-family: Verdana, "Lucida Grande", Tahoma, Helvetica;
    font-size: 11px;
    color:#333;
    margin-left: 10px;
    border-left: 3px solid #ccc;
}

p.list a
{
    color:#333;
    margin-left:15px;
}

p.list a:hover
}
    border-bottom: 1px dashed #333;
}
```

Other tags used: [category_list](#), [recent_articles](#)

70 if comments

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_comments>
```

The ***if_comments*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_comments>
...conditional statement...
</txp:if_comments>
```

The tag will execute the contained statements if there are one or more comments associated with a particular article. Should be used in an article form.

70.1 Attributes

This tag has no attributes.

70.2 Examples

70.2.1 Example 1: Number of comments associated with an article

```
<txp:if_comments>
<p><txp:comments_count /> Comments</p>
<txp:else />
<p>No Current Comments</p>
</txp:if_comments>
```

What this does...

Displays the number of comments written against the current article, otherwise display default text to indicate there are no comments.

Other tags used: [comments_count](#), [else](#)

71 if comments allowed

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_comments_allowed>
```

The **`if_comments_allowed`** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_comments_allowed>
...conditional statement...
</txp:if_comments_allowed>
```

The tag will execute the contained statements if comments are allowed for a given article.

This tag can be used in pages, or in article or comment forms; though it will allow you to use an id attribute in a comment form, the default behavior (no attribute) will ensure consistency in comment/article matching when viewing an individual article.

When used in a page template, it will test the article identified by the attribute for status and act, or not, according to that status. It will not pass the id attribute to the contained statement, such as [comments](#) or [comments_form](#), they must be added as attributes to the contained tag.

This tag is mainly used in combination with [if_comments_disallowed](#).

71.1 Attributes

This tag has no attributes.

71.2 Examples

71.2.1 Example 1: Give an indication of Comments status

Comments for articles can be turned off or on at the author's discretion for any article that is published; by using the following scheme in an article form, you can still have the on/off control over comments while also giving an indication of the comment status.

```
<txp:if_comments_allowed>
  <txp:comments_form />
</txp:if_comments_allowed>

<txp:if_comments_disallowed>
  <p>Comments are turned off for this article.</p>
</txp:if_comments_disallowed>
```

Other tags used: [comments_form](#), [if_comments_disallowed](#)

71.2.2 Example 2: Display a list of IDs

```
<txp:if_comments_allowed>
<txp:comments form="lineitem" break="li" wraptag="ul" breakclass="special" />
<txp:else />
<p>Comments closed</p>
</txp:if_comments_allowed>
```

Form (lineitem) (type: comment)

```
<txp:comment_id />
```

Styles could go this way

```
.special
{
  display:list-item;
  list-style-type:none;
}
```

What this does...

Displays a list of id numbers for comments on the current article, if comments are currently allowed.

Other tags used: [comments](#), [else](#), [comment_id](#)

72 if comments disallowed

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_comments_disallowed>
```

The ***if_comments_disallowed*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_comments_disallowed>
...conditional statement...
</txp:if_comments_disallowed>
```

The tag will execute the contained statements if comments are disallowed for a given article.

The *if_comments_disallowed* tag can be used in pages, and in article and comment forms. When used in a page template, it will test the article identified by the attribute for status and act, or not, according to that status. It will not pass the `id` attribute to the contained statement, such as `comments` or `comments_form`; they must be added as attributes to the contained tag.

Although you can use an `id` attribute in a comment form, the default behavior (no attribute) will ensure consistency in comment/article matching when viewing an individual article.

This tag is mainly used in combination with [if_comments_allowed](#).

72.1 Attributes

This tag has no attributes.

72.2 Examples

72.2.1 Example 1: Giving an indication of comment status

Comments for articles can be turned off or on at the authors discretion for any article that is published; by using the following scheme in an article form, you can still have the on/off control over comments while still giving users indication of comment status.

```
<txp:if_comments_allowed>
  <txp:comments_form />
</txp:if_comments_allowed>

<txp:if_comments_disallowed>
  <p>Comments are turned off for this article.</p>
</txp:if_comments_disallowed>
```

Other tags used: [if_comments_allowed](#), [comments_form](#)

72.2.2 Example 2: List of comment links

```
<txp:if_comments_disallowed>
  <txp:comments form="lineitem" break="li" wraptag="ul" breakclass="special" />
</txp:if_comments_disallowed>
```

Form (lineitem) (type: comment)

```
<small><txp:comment_permalink><txp:comment_id /></txp:comment_permalink></small>
```

Styles could go this way

```
.special
{
  display:list-item;
  list-style-type:none;
}
```

What this does...

Displaying a list of links to comments for the current article, using the comment id as text, but only if comments are currently **not allowed**.

Other tags used: [comments](#), [comment_permalink](#), [comment_id](#)

73 if comments error

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_comments_error>
```

The ***if_comments_error*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_comments_error>
...conditional statement...
</txp:if_comments_error>
```

The tag will execute the contained statements when an error exists with the comments.

73.1 Attributes

This tag has no attributes:

73.2 Examples

73.2.1 Example 1: Display comments error when an error occurs

```
<txp:if_comments_error>
  <txp:comments_error />
</txp:if_comments_error>
```

Other tags used: [comments_error](#)

74 if comments preview

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_comments_preview>
```

The ***if_comments_preview*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_comments_preview>  
...conditional statement...  
</txp:if_comments_preview>
```

The tag will execute the contained statements if a comment is being previewed.

74.1 Attributes

This tag takes no attributes:

74.2 Examples

74.2.1 Example 1: Display text prompt when comment is previewed

```
<txp:if_comments_preview>  
  <p>Please review your comment before submitting</p>  
</txp:if_comments_preview>
```

75 if custom field

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_custom_field>
```

The **if_custom_field** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_custom_field>
...conditional statement...
</txp:if_custom_field>
```

The tag will execute the contained statements if one or more custom fields for a given article have content. The contents of a custom field can be displayed with the [custom_field](#) tag.

75.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="field name"

The custom field name you wish to check.

val="field value"

The custom field content you want to check for a match. **NB:** Attribute will be **changed to value** in the next release

75.2 Examples

75.2.1 Example 1: Display contents of custom fields

```
<txp:if_custom_field name="subtitle">
  <txp:custom_field name="subtitle" />
</txp:if_custom_field>
```

What it does...

Checks if a custom field has any content (at all) and display it.

Why you might do it...

Say, you are publishing book reviews on your site and you use custom fields to enter the author, title, publisher and year of publication ([see example](#)). Some of the books have a subtitle, others don't so the conditional checks if the custom field you named "subtitle" holds any content and if it does, it will be displayed. If it's empty, the field won't turn up on the page.

The whole set of custom fields could look like this:

```
<p><txp:custom_field name="author" />: <txp:custom_field name="title" /> <br />
  <txp:if_custom_field name="subtitle">
    <txp:custom_field name="subtitle" /><br />
  </txp:if_custom_field>
published by <txp:custom_field name="publisher" /> in <txp:custom_field name="year" />.</p>
```

For a **book that has a subtitle**, this may be seen:

```
<p>Stephen Covey: The Seven Habits of Highly Effective People<br />
Powerful Lessons in Personal Change<br />
published by Simon & Schuster in 2002.</p>
```

For a **book without a subtitle**, this might be shown:

```
<p>J.R.R. Tolkien: The Lord of the Rings<br />
published by HarperCollins in 2004.</p>
```

Other tags used: [custom_field](#)

75.2.2 Example 2: Check custom field value

A mood indicator:

```
<txp:if_custom_field name="mood" val="happy">
  
</txp:if_custom_field>
```

```
<txp:if_custom_field name="mood" val="sad">
  
</txp:if_custom_field>
```

What it does...

Checks the content of the custom field named `mood` to see if it matches the text "happy" or "sad". Depending which one it matches determines which of the emoticons is displayed.

Why you might do it...

If you define a custom field "mood", you can enter a word to indicate your mood while writing an article. You enter either "happy" or "sad".

75.2.3 Example 3: Use the tag with **else**

```
<txp:if_custom_field name="website">
  <txp:custom_field name="website" />
<txp:else />
  <p>Unfortunately, this band hasn't got a website.</p>
</txp:if_custom_field>
```

What it does...

If the custom field named 'website' has some content, display it, otherwise display a standard message.

Why you might do it...

If you publish music reviews and you've set up some custom fields for the band name, the album title and the band's website. But not all bands have a website and you want to display a standard message if a band hasn't got one.

Other tags used: [custom_field](#), [else](#)

76 if different

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_different>
```

The **if_different** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_different>
...conditional statement with <txp:other_tag>...
</txp:if_different>
```

The tag will execute the contained statement when the value of the contained statement differs from the preceding value for that contained statement. Can be used in article, link, comment, and file forms.

76.1 Attributes

This tag has no attributes.

76.2 Examples

76.2.1 Example 1: Display posting time per article once per day

Weblog

```
<txp:if_different>
<h3><txp:posted format="%d %B %Y" /></h3>
</txp:if_different>
```

Other tags used: [posted](#)

76.2.2 Example 2: Build an indented list of article titles grouped by section

Intention:

- Display a list of all articles' titles grouped by sections and order by article title
- Add headings from section titles to designate an article's context
- Sort alphabetically by section name, then by article title

Desired result:

- about (section title)
 - ◆ 1st Article from about section
 - ◆ 2nd Article from about section
 - ◆ ?another articles
- family (section title)
 - ◆ 1st Article from family section
 - ◆ 2nd Article from family section
 - ◆ ?another articles
- people (section title)
 - ◆ 1st Article from people section
 - ◆ 2nd Article from people section
 - ◆ ?another articles

In a page template, add this tag to loop through all articles from all sections:

```
<txp:article_custom sort="section ASC, Title ASC" form="tree" />
```

The contents of the `tree` form used by the snippet above lists all article titles and renders an intermittent heading element whenever a **different** section is encountered while the articles loop through:

```
<txp:if_different><h2><txp:section title="1" /></h2></txp:if_different>
<txp:title />
```

Other tags used: [article_custom](#), [section](#)

77 if excerpt

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_excerpt>
```

The **if_excerpt** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_excerpt>
...conditional statement...
</txp:if_excerpt>
```

The tag will execute the contained statements if an excerpt is associated with the article being displayed.

77.1 Attributes

This tag has no attributes.

77.2 Examples

77.2.1 Example 1: Display the excerpt if it exists

```
<txp:if_excerpt>
  <txp:excerpt />
<txp:else />
  Section <a href="/subdirname/index.php?s=<txp:section />"><txp:section /></a>
</txp:if_excerpt>
```

If there is no excerpt associated with the current article, the section name linked as **/"subdirname/index.php?s=section_name"** will be displayed instead, providing a means of displaying the articles from the same section as the current article.

Other tags used: [excerpt](#), [else](#), [section](#)

78 if expired

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_expired>
```

The ***if_expired*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_expired>
...conditional statement...
</txp:if_expired>
```

The tag will execute the contained statements, if a particular article is expired. Should be used in an article form.

For more on this tag see its announcement, [So, you'd like to stick a 'Best Before' label on those articles?](#)

78.1 Attributes

None.

78.2 Examples

78.2.1 Example 1: Show when article has (or will) expire

```
<txp:if_expired>
This article is already expired. It expires <txp:expires />.
<txp:else />
This page isn't expired. It expires <txp:expires />.
</txp:if_expired>
```

Other tags used: [expires](#)

78.3 Related

Related tags are:

- [expires](#)
- [if_expires](#)

78.4 Genealogy

78.4.1 Version 4.0.7

- tag added

79 if expires

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_expires>
```

The ***if_expires*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_expires>
...conditional statement...
</txp:if_expires>
```

The tag will execute the contained statements, if a particular article has an expiry date set. Should be used in an article form.

For more on this tag see its announcement, [So, you'd like to stick a 'Best Before' label on those articles?](#)

79.1 Attributes

None.

79.2 Examples

79.2.1 Example 1: Show when an article is to expire

```
<txp:if_expires>
This article expires on <txp:expires />.
</txp:if_expires>
```

Other tags used: [expires](#)

79.3 Related

Related tags are:

- [expires](#)
- [if_expired](#)

79.4 Genealogy

79.4.1 Version 4.0.7

- tag added

80 if first article

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_first_article>
```

The ***if_first_article*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_first_article>
...conditional statement...
</txp:if_first_article>
```

The tag will execute the contained statements if the displayed article is the first in the currently displayed list. It will display in both single article and article list modes. Should be used in an article form.

80.1 Attributes

This tag has no attributes.

80.2 Examples

80.2.1 Example 1: Add a linked section by title

```
<h3><txp:permalink><txp:title /></txp:permalink> · <txp:posted /> by <txp:author />
<txp:if_first_article>
· Section: <txp:section link="1" title="1" />
</txp:if_first_article></h3>
<txp:body />
<txp:comments_invite wraptag="p" />
```

What this does...

Displays a link to the header of the first article in an article list.

Other tags used: [permalink](#), [title](#), [author](#), [posted](#), [section](#), [body](#), [comments_invite](#)

81 if first category

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_first_category>
```

The ***if_first_category*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_first_category>
...conditional statements...
</txp:if_first_category>
```

The tag will execute the contained statements if the current category (usually one inside the container or form of a [category_list](#)) is the first in the currently displayed list.

81.1 Attributes

This tag has no attributes.

81.2 Examples

81.2.1 Example 1: Identify 1st cat in category_list

```
<txp:category_list parent="group-1" children="0">
  <txp:if_first_category>
    <h3><txp:category /></h3>
  <txp:else />
    <txp:category link="1" />
  </txp:if_first_category>
</txp:category_list>
```

Other tags used: [category_list](#), [category](#), [else](#)

What this does...

Prevents the first category in the list from being hyperlinked to a category page

Why you might do it...

If you nest categories under a 'header' category you might want to show the header of the group but not allow people to link to its category page

81.3 Genealogy

81.3.1 Version 4.0.7

- Added as a new tag.

82 if first section

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_first_section>
```

The ***if_first_section*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_first_section>  
...conditional statements...  
</txp:if_first_section>
```

The tag will execute the contained statements if the current section (usually one inside the container or form of a [section_list](#)) is the first in the currently displayed list.

82.1 Attributes

This tag has no attributes.

82.2 Examples

82.2.1 Example 1: To be written

TBW

82.3 Genealogy

82.3.1 Version 4.0.7

- Added as a new tag.

83 if individual article

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_individual_article>
```

The ***if_individual_article*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_individual_article>
...conditional statement...
</txp:if_individual_article>
```

The tag will execute the contained statements if an individual article is being displayed (i.e. not an article list).

83.1 Attributes

This tag has no attributes.

83.2 Examples

83.2.1 Example 1: Select next-prev or older-newer navigation

```
<txp:article />

<txp:if_individual_article>
  <p>
    <txp:link_to_prev><txp:prev_title /></txp:link_to_prev>
    <txp:link_to_next><txp:next_title /></txp:link_to_next>
  </p>
</txp:if_individual_article>

<txp:if_article_list>
  <p>
    <txp:older>Previous</txp:older>
    <txp:newer>Next</txp:newer>
  </p>
</txp:if_article_list>
```

What this does...

Shows links to the next / previous article if the current page is an article, or shows links to the next / previous page of results if the current page is an article list.

Other tags used: [link_to_prev](#), [link_to_next](#), [next_title](#), [prev_title](#), [if_article_list](#), [older](#), [newer](#)

83.2.2 Example 2: Use the tag with else

```
<txp:if_individual_article>
  <p><txp:site_name /></p>
<txp:else />
  <p></img></p>
</txp:if_individual_article>
```

What this does

Displays the site's name when showing a single article, and a logo when not displaying a single article.

Other tags used: [else](#), [site_name](#)

84 if keywords

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_keywords>
```

The ***if_keywords*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_keywords>
...conditional statements...
</txp:if_keywords>
```

The tag will execute the contained statement if the current article's *keywords* field has one or more entries.

84.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

keywords="keywords"

Comma-separated list of keywords.

Default is unset, which determines whether *any* keywords are assigned to the article.

84.2 Examples

84.2.1 Example 1: Supply meta tag if keywords exist

```
<head>
[...]
```

```
<txp:if_individual_article>
```

```
    <txp:if_keywords>
```

```
        <meta name="keywords" content="<txp:keywords />" />
```

```
    <txp:else />
```

```
        <meta name="keywords" content="apple, orange, pear, foo, bar" />
```

```
    </txp:if_keywords>
```

```
<txp:else />
```

```
    <meta name="keywords" content="apple, orange, pear, foo, bar" />
```

```
</txp:if_individual_article>
```

```
[...]
```

```
</head>
```

Other tags used: [keywords](#), [if_individual_article](#), [else](#)

84.3 Genealogy

84.3.1 Version 4.0.7

- tag added

85 if last article

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_last_article>
```

The ***if_last_article*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_last_article>
...conditional statement...
</txp:if_last_article>
```

The tag will execute the contained statements if the displayed article is the last in the currently displayed list. It will display in both single article and article list modes. Should be used in an article form.

85.1 Attributes

This tag has no attributes.

85.2 Examples

85.2.1 Example 1: Add an image after the last article in a list

```
<h3><txp:permalink><txp:title /></txp:permalink> · <txp:posted /> by <txp:author /></h3>
<txp:body />
<txp:comments_invite wraptag="p" />
<txp:if_last_article>
</img>
</txp:if_last_article>
```

Other tags used: [permalink](#), [title](#), [posted](#), [author](#), [body](#), [comments_invite](#)

86 if last category

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_last_category>
```

The ***if_last_category*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_last_category>
...conditional statements...
</txp:if_last_category>
```

The tag will execute the contained statements if the current category (usually one inside the container or form of a [category_list](#)) is the last in the currently displayed list.

86.1 Attributes

This tag has no attributes.

86.2 Examples

86.2.1 Example 1: To be written

TBW

86.3 Genealogy

86.3.1 Version 4.0.7

- Added as a new tag.

87 if last section

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_last_section>
```

The ***if_last_section*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_last_section>
...conditional statements...
</txp:if_last_section>
```

The tag will execute the contained statements if the current section (usually one inside the container or form of a [section_list](#)) is the last in the currently displayed list.

87.1 Attributes

This tag has no attributes.

87.2 Examples

87.2.1 Example 1: To be written

TBW

87.3 Genealogy

87.3.1 Version 4.0.7

- Added as a new tag.

88 if plugin

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_plugin>
```

The **if_plugin** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_plugin  
...conditional statement...  
</txp:if_plugin>
```

The tag will execute the contained statements if the name attribute matches a currently installed and enabled plugin, and the current version number is equal to or greater than the ver attribute assigned (if used).

88.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="text"
Plugin name as defined on [Plugins](#) tab.

ver="number"
Minimum plugin version number.

88.2 Examples

88.2.1 Example 1: Check plugin exists before using a tag

```
<txp:if_plugin name="zem_plugin_lang" ver="4">  
<txp:zem_contact to="dest@example.com" />  
</txp:if_plugin>
```

What this does...

Apply the tag zem_contact if the zem_contact_lang plugin is installed, activated, and the version number is equal to or greater than 4.

89 if search

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_search>
```

The ***if_search*** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_search>
...conditional statement...
</txp:if_search>
```

The tag will execute the contained statements if the called page is the result of a search.

89.1 Attributes

This tag has no attributes.

89.2 Examples

89.2.1 Example 1: Select a different stylesheet during search

```
<txp:if_search>
  <link rel="stylesheet" href="<txp:css n="search" />" type="text/css" />
<txp:else />
  <link rel="stylesheet" href="<txp:css />" type="text/css" />
</txp:if_search>
```

What this does...

Selects a stylesheet named "search" when search results are being displayed, or a stylesheet determined by the active section for normal page display.

Other tags used: [else](#), [css](#)

90 if search results

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_search_results>
```

The `if_search_results` tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_search_results>
...conditional statement...
</txp:if_search_results>
```

The tag will execute the contained statements if the current article list contains a certain amount of entries matching the search term – mostly more than zero.

A typical application of this tag is the conditional output of a "Sorry, we found no items matching your search request." message, but the `min` and `max` attributes allow for a finer grained reaction to search queries.

IMPORTANT: you cannot use this tag directly inside an `if_search` tag without using an `article` tag first to actually perform the search! See Example 2 for clarification

90.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

`min="number"`

If the search results count is at least equal to `min`, the tags enclosed by this conditional tag are rendered.

Default: 1

`max="number"`

If the search results count is no higher than `max`, the tags enclosed by this conditional tag are rendered.

Default: unset, which results in no upper limit.

90.2 Examples

90.2.1 Example 1: More informative search result output

```
<txp:if_search_results>
  <p>These articles match your search request: </p>
<txp:else />
  <p>Sorry, we were not able to find a page matching your search request <strong><txp:search_term /></strong>.</p>
</txp:if_search_results>
```

What this does...

Ensures the visitor does not see a blank page when no articles match the search request.

Other tags used: `else`, `search_term`

90.2.2 Example 2: In context within `if_search`

```
<txp:if_search>
  <txp:article pgonly="1" limit="10" />
  <txp:if_search_results>
    <p>These articles match your search request: </p>
    <txp:article limit="10" searchform="results" />
  <txp:else />
    <p>Sorry, we were not able to find a page matching your search request <strong><txp:search_term /></strong>.</p>
  </txp:if_search_results>
</txp:if_search>
```

What this does...

Detects if a search is in progress, calls the `article` tag to perform the search but **inhibits display via the `pgonly` attribute**. Once the search has been performed (internally) and Textpattern knows how many search results there are, you can then use `if_search_results` to detect whether there were any or not.

Why you have to do this...

Because it's the only way to use the tag! Trying to use it without first calling an `article` tag will give unexpected results and, more often than not, a "Page template ... does not contain a txp:article tag"

You must ensure that all attributes used in your two article tags are identical (except for any `form` attributes, which can safely be omitted when using `pgonly`). Failure to keep the tags in sync will result in strange article counts or odd behaviour.
Other tags used: [if_search](#), [article](#), [else](#), [search_term](#)

90.2.3 Example 3: Take action when there are too many hits

```
<txp:if_search_results max="500">
  <p>These articles match your search request: </p>
<txp:else />
  <p>Seems like you are looking for a very common search term. Try using a more specific search phrase.</p>
</txp:if_search_results>
```

What this does...

Advises the visitor to search for something more specific in the case where their search term generated an excessive amount of hits.

Other tags used: [else](#)

90.3 Genealogy

90.3.1 Version 4.0.6

- tag added

91 if section

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_section>
```

The **if_section** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_section>
...conditional statement...
</txp:if_section>
```

The tag will execute the contained statements if the called page is part of the section specified with the name attribute.

91.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="section"

Comma-separated list of section names. For the default section, either use the text *default* or a single comma (for example, both *name=" , other_section"* and *name="default, other_section"* are equivalent).

91.2 Examples

91.2.1 Example 1: Conditionally display text for a section

```
<txp:if_section name="about">
  <p>danger, ego pages ahead!</p>
<txp:else />
  <p>nothing. just nothing. any ideas? anybody?</p>
</txp:if_section>
```

Other tags used: [else](#)

91.2.2 Example 2: Add a special class to mark the currently active section

```
<h4>Menu</h4>
<ul class="nav">
  <li<txp:if_section name=",article"> class="active"</txp:if_section>>
    <txp:section link="1" title="1" name="" />
  </li>
  <li<txp:if_section name="portfolio"> class="active"</txp:if_section>>
    <txp:section link="1" title="1" name="portfolio" />
  </li>
  <li<txp:if_section name="about"> class="active"</txp:if_section>>
    <txp:section link="1" title="1" name="about" />
  </li>
</ul>
```

A different way of marking the active section can be accomplished by using [section_list](#) and its attribute *active_class*. While the above snippet will mark the list item, [section_list](#) will mark solely the link.

Other tags used: [section](#)

92 if status

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_status>
```

The **if_status** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_status>  
...conditional statement...  
</txp:if_status>
```

The tag will execute the contained statements depending on the requested page's HTTP status condition. Normal pages result in a status code of "200", while missing pages set Textpattern's status to "404".

This tag provides a method of sharing one page template between common pages and error pages, but including different output depending on the page's HTTP status.

92.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

status="number"
Numerical HTTP status code.
Default: 200 (OK)

92.2 Examples

92.2.1 Example 1: Conditionally display text on missing pages

```
<txp:if_status status="404">  
  <p>The page you requested could not be found.</p>  
</txp:if_status>
```

93 if variable

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:if_variable>
```

The **if_variable** tag is a *conditional* tag and always used as an opening and closing pair, like this...

```
<txp:if_variable>
...conditional statements...
</txp:if_variable>
```

It tests the existence and/or value of a global variable set with the [variable](#) tag.

93.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="variable name"

The name of the variable you wish to check.

value="value"

(optionally) the value which the named variable must match in order for the contained statements to be executed. If this attribute is omitted, the tag returns *true* if the named variable contains any value at all (i.e. it is not empty, or is defined).

93.2 Examples

See the [variable](#) page for examples.

93.3 Genealogy

93.3.1 Version 4.0.7

- Added as a new tag.

94 image

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:image />
```

The **image** tag is a *single* tag that Textpattern will replace with the `` HTML tag matching the image of the numeric `id` assigned by Textpattern when the image was uploaded via the TXP [Images](#) tab.

94.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

id="integer"

Specifies the `id` assigned at upload of the image to display. Can be found on the [Images](#) tab. If both `name` and `id` are specified, `name` is used while `id` is ignored.

name="image name"

Specifies which image to display by its image name as shown on the [Images](#) tab.

html_id="id number"

The HTML `id` attribute applied to the `wraptag`, if set, otherwise to the `img` tag.

Default: unset.

escape="html"

Escape HTML entities such as `<`, `>` and `&` for the image's `alt` and `title` attributes.

Values: `html` or `unset`.

Default: `html`.

wraptag="tag text"

HTML tag to be used to wrap the `img` tag, specified without brackets.

Default is `unset`.

class="class name"

CSS `class` attribute applied to the `wraptag`, if set, otherwise to the `img` tag.

Default: `unset`.

style="style rule"

Inline CSS style rule.

Default: `unset`.

align^{deprecated}="HTML value"

HTML `align` attribute for the `img` tag. Recommended that you use CSS via `class` or `html_id` attribute instead.

94.2 Examples

94.2.1 Example 1: Display the given image

```
<txp:image id="42" />
```

What it does...

Displays the image uploaded as ID #42.

94.2.2 Example 2: Apply a CSS class

```
<txp:image name="chickens.jpg" class="boxit" />
```

What it does...

Displays the image named "chickens.jpg" and assigns a CSS class called `boxit` to the `` tag.

The style could be defined like this

```
img.boxit {
  background: #fff;
  border: 1px solid #ccc;
  display: block;
  margin: -5px 5px 5px -5px;
  padding: 4px;
  position: relative;
}
```

Had the `wraptag` attribute been used, the `boxit` class would have been applied to that instead of directly to the image.

94.3 Genealogy

94.3.1 Version 4.2.0

- attribute `align` deprecated

94.3.2 Version 4.0.7

- default value for attribute `escape` changed from `unset` to `html`

94.3.3 Version 4.0.4

- `html_id` added
- `escape` added
- `wraptag` added

95 image display

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:image_display />
```

The ***image_display*** tag is a *single* tag that is intended to be used in tandem with [image_index](#).

The ***image_display*** tag displays an image specified by the page URL which in turn is built by its tandem tag [image_index](#).

To use this tag successfully, it has to be placed either inside an article which shares a common category with the images to display (thereby linking article and image categories), or in a location at the page template which is displayed without any special article context.

If this tag seems to display no image at all, it probably resides inside an article which is never rendered as it does not belong to the currently active category.

95.1 Attributes

This tag takes no attributes.

95.2 Examples

95.2.1 Example 1: Display a single image as chosen by image_index

```
<txp:image_display />
```

96 image index

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:image_index />
```

The **image_index** tag is a *single* tag that is intended to be used in tandem with **image_display**.

It renders thumbnails of all images contained in an image category. This category can be specified as an attribute to the tag and defaults to the current site category as given in the page's URL.

The thumbnail images are linked to an address which will pass the image id plus the active category on to the tandem **image_display** tag. It is up to the user to include this tandem tag at an appropriate place inside the page template.

As the image category is passed into **image_display**, it requires to either place the "receiving" **image_display** on an article independent portion of the page (i.e. outside of the article form), or otherwise both the article used for display and the images have to share a **common** category.

96.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

c="category name"
Category of images to display.
Default: presently viewed category.

label="label text"
Label for the top of the list.
Default: unset.

labeltag="labeltag text"
Wraptag for label for the top of the list.
Default: unset.

break="tag"
(X)HTML tag (without brackets) or text to be used for line breaks.
Default: br.

wraptag="wraptag"
(X)HTML tag to be used as the wraptag, without brackets.
Default: unset.

class="class name"
CSS class attribute applied to wraptag.
Default: image_index.

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

limit="integer"
The number of images to display.
Default: 0 (no limit).

sort="sort value(s)"
How to sort resulting list.
Values:
 id (image id#)
 name (image name)
 category (image category)
 ext (image extension)
 w (image width)
 h (image height)
 alt (image alt text)
 caption (image caption text)
 date (date posted)
 author
 thumbnail
 rand() (random)
 Each field in the "txp_image"-table can be used as a sort key.
Default: name asc.

96.2 Examples

96.2.1 Example 1: Create a list of images in a category

```
<txp:image_index c="personal" break="li" wraptag="ol" />
```

What it does...

Shows the thumbnail images from the category "personal".

96.3 Reference

[image_display](#)

97 Category:Image Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Image Tags are a subcategory of the [Tag Reference](#). They are tags that are used to display pictures that are managed through the [Images](#) tab.

Download [Category:Image_Tags](#) book (see [here](#) for more details)

98 keywords

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:keywords />
```

The **keywords** tag is a *single* tag that Textpattern will replace with the keywords associated with the article being displayed. Should be used in an article form.

98.1 Attributes

This tag takes no attributes.

98.2 Examples

98.2.1 Example 1: Display the list of keywords

```
<h3>
  <txp:title />
  <span class="permlink"><txp:permlink>::</txp:permlink></span>
  <span class="date"><txp:posted /></span>
</h3>
<txp:body />
<hr>
<txp:keywords />
```

What this does...

In an article form, the title, a link to the article, its posted date, the body text and finally any keywords associated with an article are displayed

Other tags used: [title](#), [permlink](#), [posted](#), [body](#)

99 lang

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:lang />
```

The **lang** tag is a *single* tag. Textpattern will replace this tag with the 2-letter code of the language which is set as the site's language preference on the [Preferences Subtab](#), according to [RFC 1766](#).

99.1 Attributes

This tag has no attributes.

99.2 Examples

99.2.1 Example 1: Define a document's language

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="<txp:lang/>" lang="<txp:lang/>">
<head>
  <title><txp:site_name/> | <txp:site_slogan/></title>
  <meta http-equiv="Content-Language" content="<txp:lang/>" />
</head>
```

Why you might do this...

When declaring a DTD, namespace and language that a site is served, the `lang` tag is useful for ensuring translators, search engines and content parsers handle the document in the correct manner.

Other tags used: [site_name](#), [site_slogan](#)

100 link

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link />
```

The **link** tag is a *single* tag which is used to return an (X)HTML hyperlink defined under the [Links](#) tab. It uses the **Title** field as the link's text.

This tag is used in "link" forms or inside the [linklist](#) container tag.

100.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

rel="relation"

Value for the HTML `rel` attribute.

Default: unset.

100.2 Examples

100.2.1 Example 1: Display a link and its description

```
<p><txp:link /><br /><txp:link_description /></p>
```

Other tags used: [link_description](#)

101 link category

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_category />
```

The *link_category* tag is a *single* tag which returns the link category as text. This tag is used in a "link" form or inside the *linklist* container tag to return information about the current link in a *linklist*.

101.1 Attributes

label="text"

Label for the top of the list.
Default: unset.

labeltag="tag text"

(X)HTML tag to wrap the label at the top of the list.
Default: unset.

wraptag="tag text"

(X)HTML tag to be used to wrap the category, without brackets.
Default: unset.

class="class name"

CSS class to apply to the wraptag
Default: unset.

title="integer"

Display link category name or its title.
Values: 0 (category name) or 1 (category title).
Default: 0.

101.2 Examples

101.2.1 Example 1: Display a link with class attribute

```
<a class="thislink" href="<txp:link_url />">  
<txp:link_name escape="html" />  
</a> : <txp:link_category title="1" />
```

Other tags used: [link_url](#), [link_name](#)

102 link date

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_date />
```

The *link_date* tag is a *single* tag which returns the date the link was created as text. This tag is used in a "link" form or inside the [linklistcontainer](#) tag to return information about the current link in a [linklist](#).

102.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

format="format string"

Override default date format, as set in [Basic Preferences](#).

Values: any valid [strftime\(\)](#) string values, or `since`.

gmt="integer"

Return either local time — according to the set time zone preferences — or GMT time.

Values: 0 (local time) or 1 (GMT time).

Default is 0.

lang="ISO language code"

Format time string suitable for the specified language (locale). Locales adhere to [ISO-639](#).

Default: unset, resulting in a time format set via preferences.

102.2 Examples

102.2.1 Example 1: Display a link with class attribute

```
<a class="thislink" href="<txp:link_url />">  
<txp:link_name escape="html" />  
</a> : <txp:link_date />
```

Other tags used: [link_url](#), [link_name](#)

103 link description

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_description />
```

The **link_description** tag is a *single* tag which is used to return the text from the **Description** field as defined under the "links" tab. This tag is used in a "link" form or inside a [linklist](#) container tag to display information about the current link.

103.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

label="text"

Label for the top of the list.

Default: unset.

labeltag="tag"

(X)HTML tag (without brackets) to wrap around label.

Default: unset.

escape="html"

Escape HTML entities in description text.

Value: html or unset.

Default: html.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around description.

Default: unset.

class="class name"

CSS class attribute for description text.

Default: unset.

103.2 Examples

103.2.1 Example 1: Display a link and its description field contents

```
<p><txp:link /><br /><txp:link_description /></p>
```

Other tags used: [link](#)

103.3 Genealogy

103.3.1 Version 4.0.7

- default value for attribute `escape` changed from `unset` to `html`

104 link feed link

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_feed_link />
```

The *link_feed_link* tag is a *single* tag. Textpattern will replace this tag with an XHTML link to the sites links RSS feed.

104.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

flavor="value"

Whether to output a link to the RSS or Atom version of the feed.

Values: `rss` or `atom`.

Default: `rss`.

format="value"

Whether to output a (X)HTML `a` tag or a (X)HTML `link` tag.

Values: `a` or `link`.

Default: `a`.

category="category name"

Restrict to specified category. Note: the category name may be different to the Title you typed when you created the category, as the names are sanitized for URL use. Check the [Categories](#) tab to ensure you are using the correct name

Value: category name.

Default: current category.

title="value"

(X)HTML `title` attribute.

Default: depends upon format used, either `RSS feed` or `Atom feed`.

label="value"

Link text for feed link when using format of `a`.

Default: unset.

wraptag="tag"

(X)HTML `tag` (without brackets) to wrap around output when using format of `a`.

Default: unset.

104.2 Examples

104.2.1 Example 1: Atom feed link with custom label

```
<txp:link_feed_link flavor="atom" label="Commerce Links" />
```

104.3 Genealogy

104.3.1 Version 4.0.4

- format added.

105 link id

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_id />
```

The *link_id* tag is a *single* tag which returns the numerical ID of the link. This tag is used in a [Links](#) form or inside the [linklist](#) container tag to show information about the current link in the list.

105.1 Attributes

This tag has no attributes.

105.2 Examples

105.2.1 Example 1: Display link information

```
<txp:linklist category="dogs">
  Link: <a href="<txp:link_url />"><txp:link_name /></a>
      (id: <txp:link_id /> | cat: <txp:link_category />)
</txp:linklist>
```

Other tags used: [linklist](#), [link_name](#), [link_category](#)

105.3 Genealogy

105.3.1 Version 4.2.0

- Added as a new tag

106 link name

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_name />
```

The **link_name** tag is a *single* tag which returns the name of the link as assigned on the links pane as text. This tag is used in a [Links](#) form or inside the [linklist](#) container tag to display information about the current link.

106.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

escape="html"

Escape HTML entities in link name text.

Values: `html` or `unset`.

Default: `html`.

106.2 Examples

106.2.1 Example 1: Display a link with class attribute

```
<a class="thislink" href="<txp:link_url />"><txp:link_name escape="html" /></a>
```

Other tags used: [link_url](#)

106.3 Genealogy

106.3.1 Version 4.0.7

- default value for attribute `escape` changed from `unset` to `html`

107 Category:Link Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Link Tags are a subcategory of the [Tag Reference](#). They are tags that are used to display links that are managed through the [Links](#) tab.

Download [Category:Link_Tags](#) book (see [here](#) for more details)

108 link to home

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_to_home>
```

The ***link_to_home*** tag is primarily a *container* tag that returns a link to the site's home page. It will apply a hyperlink to whatever it wraps.

The tag can, however, be used as a *single* tag to generate a raw base URL of the site:

```
<txp:link_to_home />
```

In this mode it operates identically to [site_url](#).

108.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

class="class name"

CSS class attribute to apply to the anchor. Will be ignored if used as a single tag.

Default: unset.

108.2 Examples

108.2.1 Example 1: Home page link with Site's name

```
<txp:link_to_home><txp:site_name /></txp:link_to_home>
```

Other tags used: [site_name](#)

109 link to next

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_to_next>
```

The *link_to_next* tag can be used as a *single* tag or a *container* tag to return the permanent URL of the next article by posting date.

If used as a container tag, the HTML required to output a hyperlink is returned; if used as a single tag, only the URL itself is returned.

109.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

showalways="boolean"

Show the wrapped value even when no next article exists.

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

Default: 0.

109.2 Examples

109.2.1 Example 1: Link to next article using the article title

```
<txp:link_to_next><txp:next_title /></txp:link_to_next>
```

Other tags used: [next_title](#)

109.2.2 Example 2: Link to next article using static text

```
<txp:link_to_next showalways="1">Next</txp:link_to_next>
```

This will always display the text "Next", even when there is no next article.

Note: while *showalways* will enable this tag to display what is wrapped inside it, *next_title* returns nothing if there is no next title, so nothing is displayed. Use text or the returned value that you need displayed.

109.2.3 Example 3: Customizing links

The container tag returns only a very basic link, which doesn't allow for customizing the link title, or adding a CSS class, etc. Using the tag in its single tag capacity opens up a lot more possibilities.

For example, to give the link an HTML title attribute of the next article's title, and also apply a class to it:

```
<a href="<txp:link_to_next />" title="<txp:title />" class="orange"><txp:title /></a>
```

Other tags used: [title](#)

110 link to prev

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_to_prev>
```

The *link_to_prev* tag can be used as a *single* tag or a *container* tag to return the permanent URL of the previous article by posting date.

If used as a container tag, the HTML required to output a hyperlink is returned; if used as a single tag, only the URL itself is returned.

110.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

showalways="boolean"

Show the wrapped value even when no previous article exists.

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

Default: 0

110.2 Examples

110.2.1 Example 1: Link to previous article using its title

```
<txp:link_to_prev><txp:prev_title /></txp:link_to_prev>
```

Other tags used: [prev_title](#)

110.2.2 Example 2: Link to previous article using static text

```
<txp:link_to_prev showalways="1">Previous</txp:link_to_prev>
```

Note: while *showalways* will enable this tag to display what is wrapped inside it, [prev_title](#) returns nothing if there is no previous title, so nothing is displayed. Use text or the returned value that you need displayed.

110.2.3 Example 3: Customizing links

The container tag returns only a very basic link, which doesn't allow for customizing the link title, or adding a CSS class, etc. Using the tag in its single tag capacity opens up a lot more possibilities.

For example, to give the link an HTML title attribute of the previous article's title, and also apply a class to it:

```
<a href="<txp:link_to_prev />" title="<txp:title />" class="orange"><txp:title /></a>
```

Other tags used: [title](#)

111 link url

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:link_url />
```

The *link_url* tag is a *single* tag which returns the URL of the link as text. This tag is used in a [Links](#) form or inside the [linklist](#) container tag to show information about the current link in the list.

111.1 Attributes

This tag has no attributes.

111.2 Examples

111.2.1 Example 1: Display a link with class attribute

```
<a class="thislink" href="<txp:link_url />"><txp:link_name /></a>
```

Other tags used: [link_name](#)

112 linkdesctitle

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:linkdesctitle />
```

The **linkdesctitle** tag is a *single* tag which is used to return an (X)HTML hyperlink, defined under the [Links](#) tab.

It uses the **Title** field as the link's text; the **Description** field contents will be displayed as an anchor title attribute. This tag is used in a link form or inside the [linklist](#) container tag.

112.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

rel="relation"

Value for the HTML `rel` attribute.

Default: unset.

112.2 Examples

112.2.1 Example 1: Display a link and its Title field contents

```
<p><txp:linkdesctitle /></p>
```

Link, *title=Description*

113 linklist

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:linklist />
```

The *linklist* tag is a *single* or a *container* tag which is used to produce a list of links from the predefined list created on the [Links](#) tab.

If used as a container, it must be specified as an opening and closing pair of tags, like this:

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

113.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

category="category name(s)"

Restrict to links from specified category/ies.

Values: (comma separated list of) category name(s). Note: category names may be different to the Title you typed when you created the category, as the names are sanitized for URL use. Check the [Categories](#) tab to ensure you are using the correct names

Default: unset.

sort="sort value(s)"

How to sort the resulting list.

Values: id, linkname, url, description, category, date, linksort, rand() (**random**).

Default: linksort asc.

limit="integer"

Number of links to display.

Default: 0 (no limit).

offset="integer"

The number of links to skip.

Default: 0.

form="form name"

Use specified form.

Default: plainlinks.

label="text"

Label for the top of the list.

Default: unset.

labeltag="tag"

(X)HTML tag (without brackets) to wrap around label.

Default: unset.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around list.

Default: unset.

break="value"

(X)HTML tag (without brackets) or string used to separate list items.

Default: unset.

class="class name"

(X)HTML class attribute to be applied to wraptag.

Default: linklist.

113.2 Examples

113.2.1 Example 1: List of links from specified category

```
<txp:linklist form="Links" category="general" limit="10" sort="linksort" wraptag="p" />
```

113.2.2 Example 2: Links as a selectable ordered list

This example uses the displayed page's category as the criterion for choosing the linklist's category.

```
<txp:if_category name="100">
<txp:linklist label="First Floor" category="First" wraptag="ol" break="li" />
</txp:if_category>
```

```
<txp:if_category name="200">
<txp:linklist label="Second Floor" category="Second" wraptag="ol" break="li" />
</txp:if_category>
```

Other tags used: [if_category](#)

113.2.3 Example 3: Links Form (default Links)

```
<txp:link /><br />
<txp:link_description /><br />
<txp:linkdesctitle />
```

The form is repeated for each link provided by *linklist*.

113.3 Genealogy

113.3.1 Version 4.0.7

- Can be used as a container tag.

113.3.2 Version 4.0.6

- support added for comma separated list for category attribute

114 Category:List Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

List Tags are a subcategory of the [Tag Reference](#).

Download [Category:List_Tags](#) book (see [here](#) for more details)

115 Category:Markup Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Markup Tags are a subcategory of the [Tag Reference](#). They are tags that are used to add (mostly) meta information to a page through or to html elements.

[Download Category:Markup_Tags book](#) (see [here](#) for more details)

116 meta author

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:meta_author />
```

Used in the head of an individual article page template, the **meta_author** tag is a *single* tag. Textpattern will replace this tag with an HTML meta tag as follows:

```
<meta name="author" content="Article author's name" />
```

116.1 Attributes

This tag takes no attributes.

116.2 Examples

116.2.1 Example 1: Use article author for meta tag content

Article's **author** name: *sysop*

Tag: In the head of an individual article page `<txp:meta_author />`

Output: `<meta name="author" content="sysop" />`

117 meta keywords

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:meta_keywords />
```

Used in the head of an individual article page template, the **meta_keywords** tag is a *single* tag. Textpattern will replace this tag with an HTML meta tag as follows:

```
<meta name="keywords" content="keywords as set in your article's keywords input area" />
```

117.1 Attributes

This tag takes no attributes.

117.2 Examples

117.2.1 Example 1: Use article's keywords for meta tag content

Keywords field contains: *sauce,caramel,sugar*

Tag: In the head of an individual article page `<txp:meta_keywords />`

Output: `<meta name="keywords" content="sauce,caramel,sugar" />`

Note: Tag returns nothing if no keywords are set for an article.

118 modified

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:modified />
```

The **modified** tag is a *single* tag which is used to return the modification date of the article being displayed. The format is determined by the settings specified in the Date Format, or Archive Date Format, fields on the [Basic Preferences](#) tab.

118.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

format="format string"

Override the default date format set in the preferences.

Values: any valid `strftime()` string values, `since`, `iso8601`, `w3cdtf`, or `rfc822`.

Default: unset (date format set via preferences).

gmt="boolean"

Return either local time according to the set time zone preferences or GMT time.

Default: 0 (local time set in preferences).

lang="ISO language code"

Format time string suitable for the specified language (locale).

Values: locales adhere to [ISO-639](#).

Default: unset (time format set via preferences).

118.2 Examples

118.2.1 Example 1: Display "since" format date setting

```
<p>modified: <txp:modified format="since" /></p>
```

would result in:

```
<p>modified: 29 Days ago</p>
```

118.2.2 Example 2: Display custom date format

```
<p>modified: <txp:modified format="%b %d, %Y" /></p>
```

would result in:

```
<p>modified: May 28, 2005</p>
```

118.3 Genealogy

118.3.1 Version 4.0.7

- tag added.

119 Category:Navigation Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Navigation Tags are a subcategory of the [Tag Reference](#). These are all tags that (can) create hyperlinks.

Download [Category:Navigation_Tags](#) book (see [here](#) for more details)

120 newer

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

`<txp:newer>`

The **newer** tag is both a *single* tag and a *container* tag. Should be used in a page after an article tag.

Textpattern will replace this tag with an XHTML link to the next list of articles in the sort order. The container tags wrap the text or tag assigned to the link. As a single tag it outputs the URL for the next list page.

An article list consists of the assigned number of articles set by the article tag. If there are no articles available having **Newer** status (articles ranked higher, or newer, in the present sort criteria than the present top of page article) `<txp:newer>` will not display unless the `showalways` attribute is set to 1. It is normally seen used in tandem with `older`.

Given a `<txp:article limit = "5" />` tag on the page in question, `<txp:newer>` will page up five articles at a time from the oldest post forward in time to the most recently posted article.

Note: This tag is context-sensitive, meaning it will only grab content from the section or category being viewed.

120.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

`showalways="boolean"`

Show wrapped value even when no newer page exists.

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

Default: 0

120.2 Examples

120.2.1 Example 1: Container tag: link with text

```
<txp:newer>Newer</txp:newer>
```

120.2.2 Example 2: Single tag: link with image

```
<a href="<txp:newer />"><txp:image name="right-arrow.gif" /></a>
```

120.2.3 Example 3: Container tag: link with image

```
<txp:newer><txp:image name="right-arrow.gif" /></txp:newer>
```

The difference between examples 2 and 3 is that the tags in example 2 will display the image even if there are no newer articles, those used in example 3 won't.

Other tags used: [image](#)

120.2.4 Example 4: Container tag: link with div wrapper

```
<txp:newer>
  <div class="nav-link">
    <a href="<txp:newer />"><txp:text item="newer" /></a> &raquo;
  </div>
</txp:newer>
```

A few things to note about this example:

- The entire container is only output if there is newer content
- The link and div wrapper are constructed manually
- The anchor only encompasses the textual part, leaving the arrow outside

Other tags used: [text](#)

121 next title

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:next_title />
```

The ***next_title*** tag is a *single* tag which Textpattern will replace with the title of the next article in the sort order.

The container tag [link_to_next](#) wraps the text or tag and assigns the link.

121.1 Attributes

This tag takes no attributes.

121.2 Examples

121.2.1 Example 1: Link to next article by its title

```
<txp:link_to_next><txp:next_title /></txp:link_to_next>
```

Other tags used: [link_to_next](#)

122 older

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

`<txp:older>`

The **older** tag is both a *single* tag and a *container* tag. Should be used in a page after an article tag.

Textpattern will replace this tag with an XHTML link to the next list of articles in the sort order. The container tags wrap the text or tag assigned to the link. As a single tag it outputs the URL for the previous list page.

An article list consists of the assigned number of articles set by the article tag. If there are no articles available having **Older** status (articles ranked lower, or later, in the present sort criteria than the present bottom of page article) `<txp:older>` will not display unless the `showalways` attribute is set to 1. It is normally seen used in tandem with [newer](#).

Given a `<txp:article limit = "5" />` tag on the page in question `<txp:older>` will page down five articles at a time from the most recent post back in time to the oldest.

Note: This tag is context-sensitive, meaning it will only grab content from the section or category being viewed.

122.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

`showalways="boolean"`
Show wrapped value even when no newer page exists.
Values: 0 (no) or 1 (yes).
Default: 0

122.2 Examples

122.2.1 Example 1: Container tag: link with text

```
<txp:older>Older</txp:older>
```

122.2.2 Example 2: Single tag: link with image

```
<a href="<txp:older />"><txp:image name="left-arrow.gif" /></a>
```

122.2.3 Example 3: Container tag: link with image

```
<txp:older><txp:image name="left-arrow.gif" /></txp:older>
```

The difference between examples 2 and 3 is that the tags in example 2 will display the image even if there are no older articles, those used in example 3 won't.

Other tags used: [image](#)

122.2.4 Example 4: Older/newer nav system with div wrapper

```
<div id="navigation">
<txp:older>
  <div class="nav-link prev">
    &laquo; <a href="<txp:older />"><txp:text item="older" /></a>
  </div>
</txp:older>

<txp:newer>
  <div class="nav-link next">
    <a href="<txp:newer />"><txp:text item="newer" /></a> &raquo;
  </div>
</txp:newer>
</div>
```

A few things to note about this example:

- The older and newer containers are only output if there is older/newer content
- The main `navigation` div is output regardless if there are any older/newer pages so it can remain styled on the page as a placeholder
- The link and div wrappers are constructed manually and each div is classed so they can be treated differently if required
- The anchor link only encompasses the textual part, leaving the arrows outside

Other tags used: `text`, `newer`

123 output form

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:output_form>
```

The **output_form** tag can be used as a *single* or a *container* tag. Textpattern will replace this tag with the content resulting from the form called by the tag.

For the container tag usage, see the [yield](#) tag.

123.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

form="form name"

Use specified form.

Default: unset (no output).

123.2 Examples

123.2.1 Example 1: Display static text at the head of a column

```
<txp:output_form form="headtext" />
```

123.2.2 Example 2: Manage header for all pages

Suppose you want to manage the DOCTYPE and the <head> section of your page template as a single-sourced block of content. You can create a form called **page-header** and save it as type **misc**. The content of the form might look like this (just one example):

```
<tt><?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
<head>
<title><txp:page_title /></title>
<txp:css />
</head></tt>
```

Then in each of your pages, you insert the header using...

```
<txp:output_form form="haut_page" />
```

...which will add the header to all the pages automatically.

The advantage of this is that when you edit your page header, you can do so once in the form and it will update all instances of use in your different pages at the same time.

123.2.3 Example 3: Manage small pieces of static text

You can use this tag in combination with a form to create small pieces of text that would not otherwise be managed as a regular article. For example you might define the copyright conditions of content on your site in a form and add that to one or more places via the output_form tag. Name the form **copyright**, save it as type **misc** and call the form using the tag structure...

```
<txp:output_form form="copyright">
```

123.3 Genealogy

123.3.1 Version 4.2.0

- Can be used as a container tag.

124 page title

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:page_title />
```

The **page_title** tag is a *single* tag that displays text depending on the context it is used. Its primary purpose is for outputting information suitable for the HTML `<title>` tag.

Results appear as follows:

Article List

Your site name

Articles by Category

Your site name : Category title

Search Results page

Your site name : Search results: Search term

Single Article page

Your site name : Article name

Comments display

Comments on: Article name

124.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

separator="character(s)"

The character sequence you want between each piece of information.

Default: : (a colon).

124.2 Examples

124.2.1 Example 1: Show page titles with custom separator

```
<title><txp:page_title separator=" &raquo; " /></title>
```

125 page url

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:page_url />
```

The **page_url** tag is a *single* tag. It is used to return a particular component of the url from the current page being displayed.

125.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

type="type"

Specifies which component of the current page's url will be returned.

Values:

`request_uri`: current article's URL—title including any query string
`id`: current article's id on a single article page
`s`: current page's section
`c`: current page's category
`q`: search query string
`pg`: current page number in article list mode
`month`: current page's month on time based article lists
`author`: current page's author on article lists filtered by author
`status`: HTTP error response (200, 404)

Default: `request_uri`.

125.2 Examples

125.2.1 Example 1: Show the current article's ID, HTTP status and section

```
<p>Article ID: <txp:page_url type="id" /><br />  
From section: <txp:page_url type="s" />  
(Result: <txp:page_url type="status" />)</p>
```

126 password protect

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:password_protect />
```

The **password_protect** tag is a *single* tag. When Textpattern encounters the password protect tag it causes the user to be prompted for username and password, if these match the attributes set in the tag, the user is allowed access to the site. The tag can go anywhere, from page template, to article and forms.

126.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

login="text"

The username the user has to enter.

Default: unset.

pass="text"

The password the user has to enter.

Default: unset.

126.2 Examples

126.2.1 Example 1: Cause Textpattern to prompt the user for a login

```
<txp:password_protect login="theuser" pass="thepassword" />
```

Note: It is not adequate to protect a single section. This is not due to the tag itself, but rather because of how Textpattern handles URLs. By changing the URL an article can be rendered with a different section template, which would mean that the tag in the protected section would not be rendered and could not protect the article – only page requests that would be rendered in that section would be protected.

127 permalink

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:permalink>
```

The **permalink** can be used as a *single* tag or a *container* tag to return the permanent url of the article being displayed.

If used as a container tag, the HTML required to output a hyperlink is returned; if used as a single tag, only the url itself is returned.

127.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

id="article id"

The article ID to link.
Default: unset (current article).

class="class name"

CSS class attribute.
Default: unset.

style="style rule"

Inline CSS style definition. Recommended that you use CSS via `class` attribute instead.
Default: unset.

title="text"

HTML title attribute.
Default: unset.

127.2 Examples

127.2.1 Example 1: Container tag

```
<txp:permalink><txp:title /></txp:permalink>
```

would result in something like:

```
<a href="http://example.com/index.php?id=2">Article title</a>
```

Other tags used: [title](#)

127.2.2 Example 2: Single Tag

```
<txp:permalink />
```

would result in something like:

```
http://example.com/index.php?id=2
```

127.2.3 Example 3: Customizing Permanent Links

By default `permalink` returns only a very basic link, which doesn't allow for customizing the link title, or adding a CSS class, etc. Using the tag in its single tag capacity opens up a lot more possibilities.

For example, to have the permanent link have an HTML title attribute of the article's title, and also apply a class to it named "orange":

```
<a href="<txp:permalink />" title="<txp:title />" class="orange"><txp:title /></a>
```

Other tags used: [title](#)

128 php

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:php>
```

The **php** tag is a *container* tag that is the textpattern equivalent of `<?php //Code goes here... ?>`.

Use it like this:

```
<txp:php>
// Code goes here...
</txp:php>
```

Control over where this tag is allowed to appear (i.e. the privileges required to allow it to appear in pages and forms) are governed by settings in the [Advanced Preferences](#) tab.

128.1 Attributes

This tag has no attributes.

128.2 Examples

128.2.1 Example 1: Display PHP server library information

```
<txp:php>
    phpinfo();
</txp:php>
```

128.2.2 Example 2: Show the current linked category title

```
<txp:php>
    echo "The current TXP category is: "
        . category(array(
            'title' => '1',
            'link' => '1',
            'wraptag' => 'div'
        ));
</txp:php>
```

Other tags used: [category](#)

Note that all TXP tags have an equivalent programmatic name which is exactly the same as its tag name. e.g. `<txp:recent_articles />` is `recent_articles()`.

Also note that you must pass an array to all tag functions, even if there are no attributes to set. For tags that require no attributes or those that you do not wish to modify the defaults, pass an empty array, viz: `category1(array());`

129 popup

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:popup />
```

The **popup** tag is a *single* tag. Textpattern will replace this tag with a popup selector for browsing by section or category.

129.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

label="text"

Label for the top of the list.

Default: Browse.

type="type"

Section or Category.

Values: s (section) or c (category).

Default: c

wraptag="text"

(X)HTML tag (without brackets) to wrap around list.

Default: unset.

section="section_name"

Jump to the selected category for the named section.

Default: unset.

this_section="boolean"

Jump to the selected category for the currently active section.

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

Default: 0.

129.2 Examples

129.2.1 Example 1: Browse by category popup selector

```
<txp:popup type="c" wraptag="p" />
```

129.2.2 Example 2: Popup selector with custom label

```
<txp:popup label="Browse this site" type="c" wraptag="p" />
```

130 posted

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:posted />
```

The **posted** tag is a *single* tag which is used to return the publish date of the article being displayed. The format is determined by the settings specified in the Date Format, or Archive Date Format, fields on the [Basic Preferences](#) tab.

130.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

format="format string"

Override the default date format set in the preferences.

Values: any valid `strftime()` string values, `since`, `iso8601`, `w3cdtf`, or `rfc822`.

Default: unset (date format set via preferences).

gmt="boolean"

Return either local time according to the set time zone preferences or GMT time.

Default: 0 (local time set in preferences).

wraptag="tag"

HTML tag surrounding the posted date, without brackets.

Default: unset.

class="class name"

CSS class name which will be applied to the `wraptag` element.

Default: unset.

lang="ISO language code"

Format time string suitable for the specified language (locale).

Values: locales adhere to [ISO-639](#).

Default: unset (time format set via preferences).

130.2 Examples

130.2.1 Example 1: "since" format date setting

```
<p>Posted: <txp:posted format="since" /></p>
```

would result in:

```
<p>Posted: 29 Days ago</p>
```

130.2.2 Example 2: Custom format date setting

```
<p>Posted: <txp:posted format="%b %d, %Y" /></p>
```

would result in:

```
<p>Posted: Sep 18, 2008</p>
```

130.2.3 Example 3: Extended custom format date setting

```
<p>Posted:
<txp:posted format="%Y" wraptag="span" class="year" />
<txp:posted format="%B" wraptag="span" class="month" />
<txp:posted format="%e" wraptag="span" class="day" />
</p>
```

would result in:

```
<p>Posted:
<span class="year">2008</span>
<span class="month">Sep</span>
<span class="day">18</span>
</p>
```

This provides styling hooks for each date part.

130.3 Genealogy

130.3.1 Version 4.0.4

- `class` and `wraptag` added

131 prev title

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:prev_title />
```

The ***prev_title*** tag is a *single* tag which Textpattern will replace with the name (text) of the previous article in the sort order.

The container tag [link_to_prev](#) wraps the text or tag and assigns the link.

131.1 Attributes

This tag takes no attributes.

131.2 Examples

131.2.1 Example 1: Display a link to the previous article when displaying individual articles

```
<txp:link_to_prev><txp:prev_title /></txp:link_to_prev>
```

Other tags used: [link_to_prev](#)

132 Category:Programmer Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Programmer Tags are a subcategory of the [Tag Reference](#). They are tags that help developers and people who can write PHP to customise Textpattern beyond the core tags.

[Download Category:Programmer_Tags](#) book (see [here](#) for more details)

133 recent articles

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

<txp:recent_articles />

The **recent_articles** tag is a *single* tag which is used to produce a list of permanent links to recent articles by title.

133.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

section="section_name"

Restrict to articles from specified section(s).
Values: (comma separated list of) section name(s).
Default: unset.

category="category name"

Restrict to articles from specified category/ies. Note: category names may be different to the Title you typed when you created the category. Check the [Categories](#) tab to ensure you are using the correct names.
Values: (comma separated list of) category name(s).
Default: unset.

sort="sort value(s)"

How to sort resulting list.
Values:
ID (article id#)
AuthorID (author)
LastMod (date last modified)
LastModID (author of last modification)
Posted (date posted)
Title
Category1
Category2
comments_count
Status
Section
Keywords
Image (article image id#)
url_title
custom_1 through custom_10
(From 4.2.0 on: custom_n)
rand() (random).
Default: Posted desc.

limit="integer"

Number of articles to display.
Default: 10.

no_widow="boolean"

Whether to inhibit line breaks in titles which would leave just a single word on the last line (*widows*).
Values: 0 single words allowed; 1 single words not allowed on their own line
Default: Whatever is set in [Advanced Preferences](#)

label="text"

Label for the top of the list.
Default: Recent Articles.

labeltag="tag"

(X)HTML tag (without brackets) to wrap around label.
Default: unset.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around list.
Default: unset.

break="value"

(X)HTML tag (without brackets) or string used to separate list items.
Default: br.

class="class name"

(X)HTML class attribute to be applied to wraptag.
Default: recent_articles.

133.2 Examples

133.2.1 Example 1: Labelled list of recent articles

```
<txp:recent_articles label="Latest and Greatest" limit="5" />
```

133.2.2 Example 2: List of recent articles by category

```
<txp:recent_articles label="Latest" break="br" wraptag="p" category="code" sort="Section desc" />
```

133.2.3 Example 3: Styled recent article list

```
<txp:recent_articles label="Recently" break="li" wraptag="ul" />
```

Styles could go this way

```
.recent_articles {  
    list-style-type:none;  
}
```

133.3 Genealogy

133.3.1 Version 4.0.6

- support added for comma separated lists for section and category attributes

134 recent comments

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:recent_comments />
```

The **recent_comments** tag is a *single* or a *container* tag. Textpattern will replace this tag with a list of permanent links to recent comments. This list will be displayed with the format

User's Name (Article Name)

If used as a container, the tag must be specified as an opening and closing pair, like this:

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

134.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

sort="sort value(s)"

How to sort the resulting list.

Values:

- discussid (comment ID#)
- parentid (article ID#)
- name
- email
- web
- ip (IP address)
- posted
- message
- rand() (random)

Default: posted asc.

limit="integer"

Number of comments to display.

Default: 10.

offset="integer"

Number of comments to skip.

Default: unset.

label="label text"

Label for the top of the list.

Default: unset.

labeltag="tag"

(X)HTML tag (without brackets) to wrap around label.

Default: unset.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around list.

Default: unset.

break="value"

(X)HTML tag (without brackets) or string used to separate list items.

Default: br.

class="class name"

(X)HTML class attribute to be applied to wraptag.

Default: recent_comments.

134.2 Examples

134.2.1 Example 1: Labelled list of recent comments

```
<txp:recent_comments label="Recent Comments" limit="25" wraptag="p" break="br" />
```

134.2.2 Example 2: Recent comments as an unordered list

```
<txp:recent_comments label="Recent Comments" wraptag="ul" break="li" />
```

Style for default class could go this way:

```
.recent_comments {  
    list-style-type:none;  
}
```

134.3 Genealogy

134.3.1 Version 4.0.7

- Can be used as a container tag.
- `offset` attribute added.

135 related articles

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:related_articles>
```

The **related_articles** tag can be used as either a *single* tag or a *container* tag, and is used to produce a list of related (by category) articles.

When used as a *container* tag, it must be specified as an opening and closing pair of tags, like this:

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

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

Related matches are selected as follows:

If a match to Cat1 or Cat2 of the individual article being displayed is found in either Cat1 or Cat2 of any other article in the database, it will cause that article to be listed as related.

If Cat1 of the individual article being displayed is left blank and Cat2 is not blank, then all other articles are selected as being related. If both Categories are left blank, then no articles are selected.

135.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

section="section name(s)"

Restrict to articles from specified section(s).
Values: (comma separated list of) section name(s).
Default: unset.

match="category number(s)"

Restrict to articles related by specified category.
Values: "Category1", "Category2", "Category1,Category2".
Default: "Category1,Category2".

sort="sort value(s)"

How to sort resulting list.
Values:
ID (article id#)
AuthorID (author)
LastMod (date last modified)
LastModID (author of last modification)
Posted (date posted)
Title
Category1
Category2
comments_count
Status
Section
Keywords
Image (article image id#)
url_title
custom_1 through custom_10
(From 4.2.0 on: custom_n)
rand() (random).
Default: Posted desc.

limit="integer"

Number of articles to display.
Default: 10.

label="text"

Label for the top of the list.
Default: unset.

labeltag="tag"

(X)HTML tag (without brackets) to wrap around label.
Default: unset.

wraptag="tag"

(X)HTML tag (without brackets) to wrap around list.

Default: unset.

break="tag"
 (X)HTML tag (without brackets) or string used to separate list items.
 Default: br.

class="class name"
 CSS class name to be applied to the wraptag.
 Default: related_articles.

form="form name"
 Use specified form.
 Default: unset. If left empty, the permlinked article title(s) will be displayed.

no_widow="boolean"
 Whether to inhibit line breaks in titles which would leave just a single word on the last line (widows).
 Values: 0 single words allowed; 1 single words not allowed on their own line
 Default: Whatever is set in [Advanced Preferences](#)

135.2 Examples

135.2.1 Example 1: Labelled list of related articles

```
<txp:related_articles label="Related" limit="5" />
```

135.2.2 Example 2: Related articles as an unordered list

```
<txp:related_articles label="Related" limit="10" break="li" wraptag="ul" />
```

Styles could go this way

```
.related_articles {
  list-style-type:none;
}
```

135.2.3 Example 3: Used as a container tag

```
<txp:related_articles label="Related" labeltag="h3" limit="10" break="li" wraptag="ul">
  <txp:permlink><txp:title /></txp:permlink> by <txp:author />
</txp:related_articles>
```

Other tags used: [permlink](#), [title](#), [author](#).

135.3 Genealogy

135.3.1 Version 4.0.6

- support added for comma separated list for section attribute

135.3.2 Version 4.0.7

- Can be used as a container tag.
- *form* and *no-widow* attributes added.

136 rsd

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

<txp:rsd />

The **rsd** tag is a *single* tag which is used to insert a [Really Simple Discoverability](#) link element helping XML-RPC client programs to configure themselves.

136.1 Attributes

None.

136.2 Examples

136.2.1 Example 1: To be written

TBW.

136.3 Genealogy

136.3.1 Version 4.0.7

- Added as a new tag.

137 search input

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:search_input />
```

The **search_input** tag is a *single* tag. This tag will provide a text entry field for search parameters and an optional button to initiate the search.

137.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

section="section name"

Use the specified section as the destination page that will display the search results.

Default: unset (use the front page)

html_id="id"

The HTML id attribute assigned to the search form.

Default: unset.

label="text"

Label for the top of the field.

Default: Search.

button="text"

Creates and labels a button to initiate the search.

Default: unset (no button is created)

size="integer"

Sets the size of the text field.

Default: 15.

wraptag="tag"

HTML tag to wrap the search field with.

Default: p.

form="form name"

Use specified form. (Yes, a TXP-form to build a customized X/HTML form.)

Default: search_input.

137.2 Examples

137.2.1 Example 1: Display a search input form

```
<txp:search_input label="Search" button="Search" size="20" wraptag="div" />
```

137.2.2 Example 2: Elements required for building a customized (X)HTML search form

You can build your own custom search form (in a TXP form, like you do with e.g. the article tag) by specifying **form**="txpformname" inside the `<txp:search_input />` tag.

The following is the content of *txpformname* and shall depict the absolute minimum of tags and attributes required:

```
<form action="<txp:site_url />">
<input type="text" name="q" />
</form>
```

If you use a customized TXP form Textpattern doesn't automatically wrap the (X)HTML form output with form tags, so you need a pair of those to enclose your code. The **name**="q" attribute/value pair inside the text input tag is required for even initiating a search query.

Other tags used: [site_url](#)

Textpattern, as of this writing, will use a user defined form named *search_results*, or an internally defined default form if no search result form is defined by you.

137.3 Genealogy

137.3.1 Version 4.0.7

- `html_id` attribute added.

138 search result count

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:search_result_count />
```

The **search_result_count** tag is a *single* tag that returns the number of articles returned by an article tag. Use [if_search](#) to count search results or use in regular page after the [article](#) tag.

138.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

text="text"

Text to display with the number of matches.

Default: `articles found`

138.2 Examples

Note: The [if_search](#) conditional tag is required to recognize actual search results, without them the number of articles is returned by default.

138.2.1 Example 1: Display a number of matches

```
<txp:if_search>
<txp:article />
<txp:search_result_count />
</txp:if_search>
```

What this does...

If the visitor is searching for articles, it will show the number of articles that matched the search term (e.g. 5) as follows: "5 articles found".

Other tags used: [if_search](#), [article](#)

138.2.2 Example 2: Number of matches with custom text

```
<txp:if_search>
<txp:search_result_count text="Hits" />
</txp:if_search />
```

What this does...

It displays the number of articles returned (e.g. 5) as follows: "5 Hits".

Other tags used: [if_search](#)

138.2.3 Example 3: Search result count above results

```
<txp:if_search>
<txp:article pgonly="1" limit="20" />
<txp:search_result_count text="Hits" />
<txp:article limit="20" />
</txp:if_search />
```

Note: The "pgonly" attribute sets the article tag to return pagination statistics without rendering the article list. Care must be taken to remain consistent with article tag attributes to keep statistics accurate.

Other tags used: [article](#), [if_search](#)

139 search result date

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:search_result_date />
```

The ***search_result_date*** tag is a *single* tag. This tag will provide the article posted date as returned by the search function.

139.1 Attributes

This tag has no attributes.

139.2 Examples

139.2.1 Example 1: Displays the posting date of an article

Used in a search results form, this offers a search result entry comprising a hyperlinked article title, the date that article was posted and a permanent link to the article.

```
<h3><txp:permalink><txp:title /></txp:permalink></h3>
<p><txp:search_result_date /><br/>
<small><txp:permalink><txp:permalink /></txp:permalink> .
<txp:posted /></small></p>
```

Other tags used: [title](#), [search_result_date](#), [permalink](#), [posted](#)

140 search result excerpt

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:search_result_excerpt />
```

The ***search_result_excerpt*** tag is a *single* tag. The tag will show the occurrence of the search term with some surrounding context.

140.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

limit="integer"

Maximum number of search match excerpts per search result.

Default: 5.

highlight="tag"

HTML tag to be used for search term matches in excerpt text, without brackets.

Default: `strong`

break="tag"

Trailing string.

Default: ellipsis (—).

140.2 Examples

140.2.1 Example 1: Display up to 15 search excerpts with a search results form

```
<h3><txp:permalink><txp:title /></txp:permalink></h3>
<p><txp:search_result_excerpt highlight="p" limit="15" />
<small><txp:permalink><txp:permalink /></txp:permalink> · <txp:posted /></small></p>
```

140.3 Genealogy

140.3.1 Version 4.0.6

- `break` added

141 search result title

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:search_result_title />
```

The ***search_result_title*** tag is a *single* tag. This tag will provide a hyperlinked title to an article as returned by the search function.

141.1 Attributes

This tag has no attributes.

141.2 Examples

141.2.1 Example 1: Display a hyperlinked title to an article

In a search results form, this shows the title of an article that matched the visitor's search results, and its posted date.

```
<h3><txp:search_result_title /> <txp:search_result_date /></h3>
```

Other tags used: [search_result_date](#)

142 search result url

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:search_result_url />
```

The ***search_result_url*** tag is a *single* tag. This tag will provide a hyperlinked URL to an article as returned by the search function.

142.1 Attributes

This tag has no attributes.

142.2 Examples

142.2.1 Example 1: Display a hyperlinked URL to an article

Used within a search results form to allow visitors to click on the link and be taken to the article that matched their search results.

```
<p><txp:search_result_url /></p>
```

143 Category:Search Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Search Tags are a subcategory of the [Tag Reference](#). They are tags that are used to react to and display information regarding visitor-submitted search results.

[Download Category:Search_Tags](#) book (see [here](#) for more details)

144 search term

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:search_term />
```

The **search_term** tag is a *single* tag which returns the expression the user searched for through the full text search form.

144.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

escape="html"

When set to html, HTML entities are used for <, >, and & in the output text.

Default: html

144.2 Examples

144.2.1 Example 1: Display the search term on the search results page

```
<txp:if_search>
<h3>Search results</h3>
<p>You searched for <strong><txp:search_term /></strong>.</p>
<txp:article />
</txp:if_search>
```

144.3 Genealogy

144.3.1 Version 4.0.6

- Tag support added

145 section

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:section>
```

The **section** tag can be used as either a *single* tag or *container* tag. It will display information about the section as defined by either the `name` attribute, the section currently being viewed, or the section of the article being displayed (if used within an [article form](#), or an [if_individual_article](#) conditional tag).

When used as a containing tag, it will turn the contents into a link to that section. Otherwise, it will return plain text.

145.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

title="boolean"

Display either the section name or its title.

Values: 0 or 1

Default: 0 (name)

link="boolean"

Display as plain text or a link.

Values: 0 or 1

Default: 0 (plain text)

wraptag="tag"

HTML tag name to be used as the wraptag, without brackets.

Default: unset.

class="class name"

CSS class name to apply to the wraptag. If no wraptag is supplied (and `link="1"`), the class is applied to the anchor instead.

Default: unset.

name="section name"

Sets the link to the named section.

Default: unset (sets the link to the current section)

url="boolean"

Display plain URL or full link.

Values: 0 or 1

Default: 0 (display title or full link, depending on `link`)

145.2 Examples

145.2.1 Example 1: Display the current section name

```
<txp:section />
```

145.2.2 Example 2: Display hyperlinked section title

```
<txp:section link="1" title="1" />
```

What this does...

In an article form, it displays the article's section title as a hyperlink to the section home page.

Otherwise, it displays the title of the section currently being viewed as a hyperlink to the section home page.

145.2.3 Example 3: Display a link to a specified section

```
<txp:section link="1" title="1" wraptag="p" name="archive" />
```

What this does...

It displays a hyperlink to the 'archive' section home page, wrapped in `<p>` tags, using the section's title as link text.

145.2.4 Example 4: Container example

```
<txp:section name="archive">My Archive</txp:section>
```

What this does...

It displays the text "My Archive" as a hyperlink to the 'archive' section home page.

145.3 Genealogy

145.3.1 Version 4.0.7

- Applies `class` attribute to the `<a>` element when `wraptag` is empty.
- New attribute, `url` to output URL only.

146 section list

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:section_list />
```

The **section_list** tag is a *single* or a *container* tag which is used to produce a list of linked sections. When used as a container tag, it is used as an opening and closing pair, like this:

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

146.0.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

sections="section name(s)"

Comma-separated list of section names to include in the list, displayed in specified order (unless overridden by the sort attribute).
Default: unset (all sections)

exclude="section name(s)"

Comma-separated list of section names to exclude from the list. Sections takes precedence over exclude.
Default: unset (none)

include_default="boolean"

Whether to include "default" section in section list.
Default: 0 (no)

sort="sort value(s)"

How to sort the resulting list.
Values:

name
page
css
is_default
in_rss
on_frontpage
searchable
title
rand() (random)

Default: name asc

form="form name"

Use the specified form to process each included section.

label="text"

Label for the top of the list.
Default: unset

default_title="text"

Text used as a title for the "default" section when `include_default` is set to 1.
Default: [Site Name](#).

labeltag="tag"

(X)HTML tag (without brackets) to wrap around label.
Default: unset

wraptag="tag"

(X)HTML tag (without brackets) to wrap around list.
Default: unset.

break="tag"

(X)HTML tag (without brackets) or string used to separate list items.
Default: `br`.

class="class name"

(X)HTML class attribute to be applied to the specified wraptag.
Default: `section_list`.

active_class="class name"

(X)HTML class attribute to be applied to the "active" or current link in a list (only works in the *single* tag without the `form` attribute).
Default: unset

146.1 Examples

146.1.1 Example 1: Display a linked section list

Adding the label "Sections" and wrapping the output in a paragraph with each section on its own line:

```
<txp:section_list label="Sections" wraptag="p" break="br" />
```

146.1.2 Example 2: Display a styled section list

```
<txp:section_list break="li" wraptag="ul" />
```

Styles could go this way

```
.section_list {  
list-style-type: none;  
}
```

146.1.3 Example 3: Set active class using the container tag

This code will add `class="active"` to the `` element around the "current" section in the list.

```
<txp:section_list wraptag="ul" break="">  
<li<txp:if_section name='<txp:section />'> class="active"</txp:if_section>>  
<txp:section title="1" link="1" />  
</li>  
</txp:section_list>
```

146.2 Genealogy

146.2.1 Version 4.0.7

- Can be used as a container tag.
- `form` attribute added.

147 site name

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:site_name />
```

The **site_name** tag is a *single* tag that returns the site's name as defined under the [Basic Preferences](#) tab.

147.1 Attributes

This tag has no attributes

147.2 Examples

147.2.1 Example 1: Display the site's name

```
<h1><txp:site_name /></h1>
```

148 site slogan

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:site_slogan />
```

The **site_slogan** is a *single* tag which is used to output the site's tagline (labeled as *Site tagline* in the [Basic Preferences](#)).

The slogan is a brief (255 chars max) tagline or description of your site which can be used, for example, in XML feeds.

148.1 Attributes

This tag has no attributes.

148.2 Examples

148.2.1 Example 1: General display of Slogan

```
<p><txp:site_slogan /></p>
```

148.2.2 Example 2: As content filler

The slogan could be used for the content attribute of the description metadata element. Either whole...

```
<meta name="description" content="<txp:site_slogan />" />
```

or partial...

```
<meta name="description" content="<txp:site_slogan />. And the rest of your pithy description would go here." />
```

149 site url

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:site_url />
```

The **site_url** tag is a *single* tag which returns the full URL of the site (as defined in the [Basic Preferences](#)) as text.

If you maintain local development versions of your live sites and import databases between them, then this tag is extremely valuable for ensuring your domain links are never confused (thus broken) between the two locations (see example 1).

149.1 Attributes

This tag has no attributes.

149.2 Examples

149.2.1 Example 1: Maintain accurate domain paths

The idea is that you don't break URL paths after importing a database from local development to live, or visa versa. By using this tag it will automatically be relative to a given site and you'll never have to manually edit broken domain paths again. A classic example is with navigation links.

```
<ul id="navmenu">
<li class="articles"><a class="nopad" href="<txp:site_url />articles" title="Articles">Articles</a></li>
<li class="photos"><a href="<txp:site_url />photos" title="Photographs">Photographs</a></li>
</ul>
```

149.2.2 Example 2: HTML header paths

In your Head section of your HTML pages you might have a variety of links to locations relative to the local server, such as CSS files, Javascript files, a favicon and so forth. The relevance is similar to example #1, you want to ensure the paths are accurate relative to the server if a database has been imported from another location. Following is an example for the *shortcut icon* file (if you use one).

```
<link rel="shortcut icon" href="<txp:site_url />favicon.ico" />
```

149.2.3 Example 3: Display a hyperlink to download a text file

```
<a href="<txp:site_url />download.txt">Download</a>
```

150 Category:Structural Tags

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

Structural Tags are a subcategory of the [Tag Reference](#). They are tags that are used to 'group' content.

Download [Category:Structural_Tags](#) book (see [here](#) for more details)

151 text

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:text />
```

The **text** tag is a *single* tag which is primarily used to return localized language strings from the txp_lang database table.

151.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

item="text"

Piece of text to display. If the given item matches a key in the txp_lang table, its contents will be returned. Otherwise, whatever you supply in the *item* tag is returned verbatim.

151.2 Examples

151.2.1 Example 1: Display some localized text

```
<txp:older><txp:text item="older" /></txp:older>
```

What it does...

Outputs the text 'older' inside the <txp:older /> tag, respecting the current TXP language.

Why you might use this...

Instead of using the tag like this: <txp:older>older</txp:older> which would always render the English text 'older', it replaces the contents with the value assigned to the name 'older' in the current language. So you would see a link with the word 'älter' if you were using German (de) as the TXP site language.

Other tags used: [older](#)

152 thumbnail

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:thumbnail />
```

The **thumbnail** tag is a *single* tag that Textpattern will replace with the `` HTML tag matching the thumbnail image of the numeric `id` assigned by Textpattern when the parent image was uploaded via the TXP [Images](#) tab.

152.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

id="integer"

Specifies the `id` assigned at upload of the image to display. Can be found on the [Images](#) tab. If both `name` and `id` are specified, `name` is used while `id` is ignored.

name="image name"

Specifies which image thumbnail to display by its image name as shown on the [Images](#) tab.

escape="html"

Escape HTML entities such as `<`, `>` and `&` for the image's `alt` and `title` attributes.

Values: `html` or `unset`.

Default: `html`.

wraptag="tag text"

HTML tag to be used to wrap the `img` tag, specified without brackets.

Default: `unset`.

class="class name"

CSS `class` attribute to apply to the image (or to the `wraptag`, if set).

Default is `unset`.

html_id="id"

The HTML `id` attribute assigned to the image (or to the `wraptag`, if set).

Default: `unset`.

style="style rule"

Inline CSS style rule.

Default: `unset`.

align^{deprecated}="HTML value"

HTML `align` attribute for the `img` tag. Recommended that you use CSS via `class` or `html_id` attribute instead.

Default: `unset`.

poplink="boolean"

If set, the image will be rendered in a popup window.

Default is `0`.

link="boolean"

If set, the thumbnail will be rendered as a (non-javascript) URL link to the full-size image.

Default: `0`.

link_rel="relation"

Value for the HTML `rel` attribute.

Default: `unset`.

152.2 Examples

152.2.1 Example 1: Display the given thumbnail

```
<txp:thumbnail id="23" />
```

What this does...

Displays the image thumbnail for the image uploaded as ID #23.

152.3 Genealogy

152.3.1 Version 4.2.0

- attribute `align` deprecated

152.3.2 Version 4.0.7

- default value for attribute `escape` changed from `unset` to `html`

152.3.3 Version 4.0.6

- `link` and `link_rel` added

152.3.4 Version 4.0.4

- `html_id`, `escape` and `wraptag` added

153 title

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:title />
```

The **title** tag is a *single* tag which is used to return the title of the article being displayed. It is usually used in an article form.

153.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

no_widow="boolean"

Controls **widows** .

Values:

0 – allow the last word in the title to appear on its own line, i.e. the title content is rendered unchanged

1 – ensure the last word is not left on its own line. Textpattern inserts an invisible code -- a non-breaking space -- between the last two words

Default: As set by *Prevent widowed words in article titles* in [Advanced Preferences](#)

153.2 Examples

153.2.1 Example 1: Display an article title

```
<h2><txp:title /></h2>
<div class="post">
  <txp:author /> @ <txp:posted /><br /><txp:body />
</div>
```

What this does...

Shows the current article title as the page heading, a few other pieces of information such as the article's author and posted date, then the article body itself.

Other tags used: [author](#), [posted](#)

153.2.2 Example 2: Display a hyperlinked title

```
<txp:permalink><txp:title /></txp:permalink>
```

What this does...

Wraps a permanent link to the current article around its title.

Other tags used: [permalink](#)

154 txp die

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:txp_die />
```

The **txp_die** tag is a *single* tag that will terminate normal page rendition and return the given status to the user agent (browser, search engine crawler, feed aggregator). An error page will also be returned to the user agent.

The status can be displayed by the [error_status](#) tag. A textual message can be associated with the error status and retrieved with the [error_message](#) tag. See also: [Custom Error Pages](#).

154.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

msg="message"

Textual representation of the error condition.

status="number"

Numerical representation of the error condition.

Typical values: 404, 403, 401, 408, 410, 304, 503, or any other [valid status code](#)

Default: 503.

154.2 Examples

154.2.1 Example 1: Force a 404 'not found' error

```
<txp:txp_die status="404" />
```

155 variable

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:variable />
```

The **variable** tag is both a *single* and a *container* tag which sets or returns a user-defined global variable.

If used as a container, the result of the contained statements are assigned to the given variable name.

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

155.1 Attributes

Tag will accept the following attributes (**case-sensitive**):

name="text"

The variable name for which you wish to assign a value. Valid variable names must not contain any single or double quotes.

value="value"

(optionally) define the value to which you wish to set the variable. Without this attribute, the tag returns the current value assigned to the named variable.

155.2 Examples

155.2.1 Example 1: Store sitewide constants

`<txp:variable />` allows a site/theme designer to define constants at a central location (a form or the location near the top of a page template) and use them elsewhere later on, for instance as the e-mail address for `zem_contact_reborn`'s `to` attribute or as the AdSense Publisher id for all three AdSense forms which are scattered throughout your template.

Somewhere at the very beginning of a template you would define names and values, just like you do on your desktop calculator's "memory" keys:

```
<txp:variable name="site-owner" value="john.doe@example.com" />
<txp:variable name="adsense-pub" value="pub-9999999" />
<txp:variable name="client-is-always-right" value="yesss" />
```

Later down the page template or in a separate form you can read the attribute values previously set:

```
<txp:zem_contact to='<txp:variable name="site-owner" />' />
```

Conditionals come in handy at times:

```
<txp:if_variable name="client-is-always-right">
  <txp:css n="fugly" />
</txp:if_variable>
```

Elsewhere :

```
<script type="text/javascript">
</script>
```

Other tags used: [if_variable](#), [css](#)

155.2.2 Example 2: Use any tag's value as a conditional expression

Example:

```
<txp:variable name="foo" value='<txp:permlink />' />

<txp:if_variable name="foo" value="http://example.com/bar/baz">

</txp:if_variable>
```

Other tags used: [if_variable](#)

155.2.3 Example 3: Build a table of article titles

Textpattern will build a three-column table where each row has the title of the article. If a number of articles is not divisible by 3 then empty cells will be inserted.

Article #1	Article #2	Article #3
Article #4	Article #5	Article #6
Article #7	Article #8	Article #9

First, use an [article_custom](#) or [article](#) tag somewhere:

```
<txp:article_custom limit="100" section="article" form="tables" />
```

Form **tables**:

```
<txp:if_first_article>
<table>
</txp:if_first_article>

<txp:if_variable name="trigger">
  <txp:if_variable name="trigger" value="2">
    <txp:variable name="trigger" value="3" />
  </txp:if_variable>

  <txp:if_variable name="trigger" value="1">
    <txp:variable name="trigger" value="2" />
  </txp:if_variable>

<txp:else />

  <txp:variable name="trigger" value="1" />

</txp:if_variable>

<txp:if_variable name="trigger" value="3">
  <td><txp:title /></td>
  </tr>
  <txp:variable name="trigger" value="1" />
</txp:if_variable>

<txp:if_variable name="trigger" value="2">
  <td><txp:title /></td>

  <txp:if_last_article>
    <td></td>
  </tr>
  </txp:if_last_article>
</txp:if_variable>

<txp:if_variable name="trigger" value="1">
  <tr>
  <td><txp:title /></td>

  <txp:if_last_article>
    <td></td><td></td>
  </tr>
  </txp:if_last_article>
</txp:if_variable>

<txp:if_last_article>
</table>
</txp:if_last_article>
```

In this case an article [title](#) was used but any article information could be inserted in the cells.

Other tags used: [if_variable](#), [if_first_article](#), [if_last_article](#), [title](#), [else](#)

155.3 Genealogy

155.3.1 Version 4.0.7

- Added as a new tag.

156 yield

Tag reference quick links:

- [Tag Reference Index](#)
- [Tag Basics](#)
- [Attributes Cross-reference](#)
- [Tags In Development](#)

```
<txp:yield />
```

The **yield** tag is a *single* tag which is used to return the inner content of the enclosing `<txp:output_form>` tag.

156.1 Attributes

This tag has no attributes.

156.2 Examples

156.2.1 Example: inner content

Given the following form named "example_form":

```
<div>
  This content is static and will be the same every time this form is invoked.
  <txp:yield />
</div>
```

We can invoke it twice with different inner content each time:

```
<txp:output_form form="example_form">
  Invoking "example_form" with some inner content.
</txp:output_form>

<txp:output_form form="example_form">
  Invoking "example_form" again, this time with different inner content.
</txp:output_form>
```

And the result will be:

```
<div>
  This content is static and will be the same every time this form is invoked.
  Invoking "example_form" with some inner content.
</div>

<div>
  This content is static and will be the same every time this form is invoked.
  Invoking "example_form" again, this time with different inner content.
</div>
```

Other tags used: [output_form](#)

156.3 Genealogy

156.3.1 Version 4.2.0

- Added as a new tag