```
/***
    USAGE:

    TagDirectory(pages, tagPrefix, columnCount, listPagesOptions)
        build a multi-column tag directory for a list of pages

    PARAMETERS:

    (optional) pages : list/map/str
        list/map of pages to list; if pages is a str, then it is used as
a path to a parent page to list all subpages;
        defaults to list of subpages of current page

    (optional) tagPrefix : str
        only use tags with given prefix (e.g. 'category:dog'); defaults
to using only unprefixed tags

    (optional) columnCount : num
        number of columns to display; defaults to 2

    (optional) listPagesOptions : map
        options passed into the ListPages template for listing each
column entry;
        defaults to { sort: 'viewcount', reverse: true, style: 'bullets' }

***/

var pages = $0 ?? $pages ?? page.subpages;
if(pages is str) let pages = wiki.getpage(pages).subpages;
if(pages is map) let pages = map.values(pages);
var tagprefix = $1 ?? $tagprefix;
var columns = num.max($2 ?? $columnCount ?? 2, 1);
var listPagesOptions = { sort: 'viewcount', reverse: true, style:
'bullets' } .. ($4 ?? $listPagesOptions ?? { });

// build map of all tags in pages
var tagmap = { };
foreach(var p in pages) {
    var tags = p.tags;

    // check if page has no tags; if so make up a default list
    if(!#tags) {
        let tags = [ { name: (tagprefix ? tagprefix .. ':' : '') ..
'(unclassified)', type: 'text' } ];
    }

    // for each tag on the page, append the page to that tag's list
    foreach(var t in tags where t.type == 'text') {
```

```
        // check if either the tag prefix matches or there is no tag
prefix
        var parts = string.split(t.name, ':', 2);
        if(tagprefix && (#parts == 2) && (string.compare(parts[0],
tagprefix, true) == 0)) {
            let tagmap ..= { (parts[1]) : tagmap[parts[1]] .. [ p ] };
        } else if(!tagprefix && (#parts == 1)) {
            let tagmap ..= { (parts[0]) : tagmap[parts[0]] .. [ p ] };
        }
    }
}

if(#tagmap) {

    // count how many pages each tag has
    var tag_count = [ { tag: tag, count: #tagmap[tag] } foreach var tag
in map.keys(tagmap) ];

    // balance columns so that their heights are as equal as possible
    var column_tags = list.new(columns, [ ]);
    var column_sums = list.new(columns, 0);

    foreach(var t in list.sort(tag_count, 'count', true)) {

        // find shortest column
        var column = list.indexof(column_sums, list.min(column_sums));

        // update column
        let column_tags = list.splice(column_tags, column) .. [
column_tags[column] .. [ t.tag ] ] .. list.splice(column_tags, 0, column
+ 1);
        let column_sums = list.splice(column_sums, column) .. [
column_sums[column] + (listPagesOptions.limit ?? t.count) + 2 ] ..
list.splice(column_sums, 0, column + 1);
    }

    // emit the table with N columns
    <table width="100%" cellspacing="0" cellpadding="5" border="0"
style="table-layout: fixed;">
        <tr valign="top">

            // loop over each column
            foreach(var column in column_tags) {
                <td style="padding-right: 20px;">

                    // loop over all tags in column, sorted alphabetically
                    foreach(var tag in list.sort(column)) {
                        <h5> string.tocamelcase(tag) </h5>
                        template("MindTouch/Controls/ListPages",
listPagesOptions .. { pages: tagmap[tag] });
```

```
                    }
                </td>
            }
        </tr>
    </table>
}
```