



Template:TagCloud

/**

USAGE:

```
TagCloud()  
    build a tag cloud from a search query.
```

PARAMETERS:

```
query: str  
    query string for selecting pages on which the tag could is built
```

```
showCount: bool  
    show number of occurrences of the tag (default: true)
```

```
max: num  
    limit tag cloud to top X tags (default: 10)
```

```
order: str  
    order tags by name or frequency (either "count" or "name"; default: "count")
```

```
exclude: list  
    list of tags to exclude from tag cloud
```

```
fontSizeUnits: str  
    display units for font size (default: "%")
```

```
fontSizeMin: num  
    smallest display font (default: 80)
```

```
fontSizeMax: num  
    largest display font size (default: 240)
```

```
fontColorRMin: num  
    lowest color value for red (default: 50)
```

```
fontColorRMax: num  
    highest color value for red (default: 50)
```

```
fontColorGMin: num  
    lowest color value for green (default: 200)
```

```

    fontColorGMax: num
        highest color value for green (default: 100)

    fontColorBMin: num
        lowest color value for blue (default: 50)

    fontColorBMax: num
        highest color value for blue (default: 50)

***/

var searchString = $query ?? "";
var hitShow = $showcount ?? true;
var tagOrder = $order ?? "name";
var tagsLimit = $max ?? 10;
var tagsToExclude = $exclude ?? [ ];
var fontSizeUnits = $fontSizeUnits ?? "%";
var fontSizeMin = $fontSizeMin ?? 80;
var fontSizeMax = $fontSizeMax ?? 240;
var fontColorRMin = $fontColorRMin ?? 50;
var fontColorRMax = $fontColorRMax ?? 50;
var fontColorGMin = $fontColorGMin ?? 200;
var fontColorGMax = $fontColorGMax ?? 100;
var fontColorBMin = $fontColorBMin ?? 50;
var fontColorBMax = $fontColorBMax ?? 50;

// TODO: figure out what this is supposed to do
if(__request.args.q) {
    <div id='tagResults'>
        <b> "Pages tagged with "; <u> __request.args.q; </u> " in
        wiki.search("tag:"..string.quote(__request.args.q).."", 1
        <a href=("Special:Tags?tag=" .. web.UriEncode(__request.a

        var defined = __request.args.q;
        var found = wiki.getsearch("tag:" .. string.quote(defined
        if(#found) {
            var def = found[0].tags[defined].definition;
            if(def) {
                <br />;
                <a href=(def.uri)> "Go to the " .. __requ

            }
        }
    }
</div>
}

```

```

// convert list of excluded tags into a map (easier to check if tag is excluded)
let tagsToExclude = { (t): true foreach var t in tagsToExclude };

// compute map of tags based on search string and excluded tags (note: on
var myResults = wiki.getsearch(searchString, 1000, _, "type:wiki");
var tagMap = { };
foreach(var p in myResults, if __index < 250, var t in map.keys(p.tags ?? {})) {
    var tag = string.startswith(t, "define:") ? string.substr(t, 7) : t;
    let tagMap += { (tag) : 1 + (tagMap[tag] ?? 0) };
}

// compute list of tags sorted by occurrence count
var tagListofMaps = [ { name: t, count: tagMap[t] } foreach var t in map.keys(tagMap)];
var tagsTotal = #tagListofMaps;
let tagListofMaps = list.sort(tagListofMaps, 'count', true);
if ((tagsLimit > 0) && !__request.args.alltags) {
    let tagListofMaps = list.splice(tagListofMaps, tagsLimit);
}
var maxValue = tagListofMaps[0].count;
var minValue = tagListofMaps[#tagListofMaps-1].count;

// sort display of tags by frequency or name
let tagListofMaps = (tagOrder == "name" ? list.sort(tagListofMaps, 'name') : tagListofMaps);

// emit tag cloud
<div id="tagCloud">
    <ul>
        foreach (var tm in tagListofMaps) {
            var tag = string.tocamelcase(tm.name);
            var tagURI = page.uri & { q: tag } .. '#tagResults';
            var count = tm.count;
            var weight = ( num.log(count) - num.log(minValue) ) / ( num.log(maxValue) - num.log(minValue) );
            var fontSize = fontSizeMin + num.round((fontSizeMax - fontSizeMin) * weight);
            var fontColorR = fontColorRMin + num.round((fontColorRMax - fontColorRMin) * weight);
            var fontColorG = fontColorGMin + num.round((fontColorGMax - fontColorGMin) * weight);
            var fontColorB = fontColorBMin + num.round((fontColorBMax - fontColorBMin) * weight);

            <li style=("padding-left: 1ex; display: inline-block; vertical-align: top; margin-right: 1ex;">
                var rgb = "rgb(" .. fontColorR .. "," .. fontColorG .. "," .. fontColorB .. ")";
                <a href=(tagURI) style=("color:" .. rgb) >{tag}</a>
                if(hitShow) {
                    <sup style=("color:" .. rgb .. ")>{count}</sup>
                }
            </li>
        }
    </ul>
</div>

```

```
        if (!__request.args.alltags && (tagsLimit > 0) && (tagsTo
            <li style="display:block">
                "Showing top " .. tagsLimit .. " tags. ";
                <a href=(page.uri & { alltags: _ } .. "#t
                    " »";
            </li>
        }
    </ul>
</div>
```