Blog

The extension *BlueSpiceSocialBlog* is part of *BlueSpiceSocial*, the communications module of BlueSpice. In addition to the type blog, there are other types of so-called social entities, such as *comment*, *discussion topic*, *attachment*, or *microblog*.
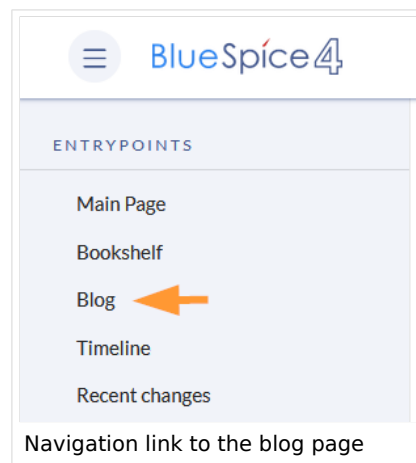
Contents

About the blog

Basically, a blog is a list of comments, questions, or feedback from different users. The blog is often sorted chronologically. Lively communication can arise when blog entries are further commented, recommended, linked with related articles, or placed on a personal watchlist. The blog is ready to use on the special page *Special:Blog*. This page is linked directly from the main navigation in every new wiki installation. So you can start using the blog right away.

Navigation link to the blog page

Characteristics and features

- A blog entry consists of a title and content.
- **Comments:** Users can leave comments to each blog entry.
- **Actions:**
  - **Delete:** Editors can delete their own posts. Administrators can delete all posts. Deleted posts are no longer displayed. Exception: Administrators see deleted items with a red background and can restore them.
  - **Recommend:** Blog posts can be recommended. It is possible to generate a list of posts with the most recommendations via the timeline filter. (See: Embedding the blog on a page)
  - **Linked page:** Every blog entry is connected to a wiki page by the system. By default, this is the main page of the wiki. However, other pages can be added as "Linked page". Blog entries can then be filtered for linked pages via the *Special:Timeline* page.
  - **Watch:** Own blog posts and comments are automatically added to the watch list.
- **Permissions:** It is possible to allow users with read permissions on the wiki to comment on blog entries. See: BlueSpiceSocial Permissions.

Adding the blog to a page

It is also possible to include the blog functionality on any wiki page. Social contributions of many types can be consolidated and displayed chronologically in a timeline and filtered and sorted as needed.

All blog entries

If you want to include a view of your blog on another page of your wiki, for example your homepage, you need to insert the necessary codeblock in the source code of your page.

To include a blog view:

1. **Go to** the page where you want to insert the blog or create a new page.
2. **Open the page** in *Source editing* mode.
3. **Copy the following code block** to the section in your page where you want to show the blog.

```
<bs:timeline>
 {
     "showentitylistmenu": true,
     "preloadtitles": {
         "blog": "Template:Blog"
     },
     "preloadedentities": [{
         "type": "blog"
     }],
     "headlinemessagekey": "Community-Blog",
      "showheadline": true,
     "usemorescroll": false,
     "morelink": "Special:Blog",
     "limit": 5,
     "sort": [ {
         "property": "timestampcreated",
         "direction": "DESC"
     }],
     "lockedfilternames": [
         "type"
     ],
     "filter": [ {
         "type": "list",
         "property": "type",
         "value": [ "blog" ],
         "comparison": "ct"
     }]
 }
</bs:timeline>
```

After saving the page, you will see a list of existing blog entries based on the filter and sorting criteria you provided.

Blog entries of a linked page

If you want to create a blog that only displays the posts of the blog posts created on this page, copy the following timeline tag. Here, under the preloadedentities, new blog entries are already linked to the page and filtered. Replace Blog1 under preloadedentities and in the filter section with the appropriate page name:

```
<bs:timeline>
 {
     "showentitylistmenu": true,
     "preloadtitles": {
         "blog": "Template:Blog"
     },
      "preloadedentities": [{
         "type": "blog",
         "tags":["Blog1"]
     }],
```

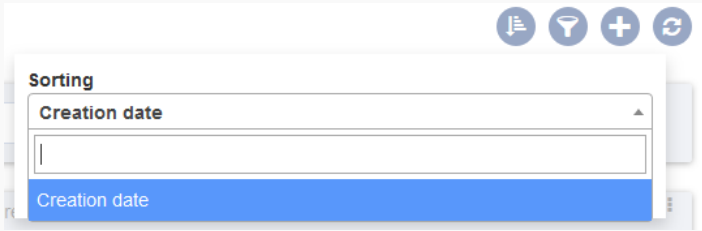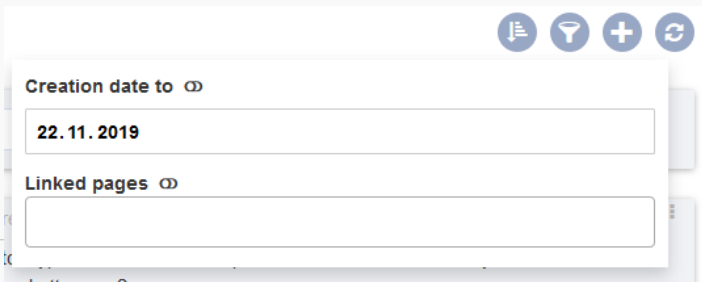```
    "headlinemessagekey": "Community-Blog",
     "showheadline": true,
    "usemorescroll": false,
    "morelink": "Special:Blog",
    "limit": 5,
    "sort": [{
        "property": "timestampcreated",
        "direction": "DESC"
    }],
    "lockedfilternames": [
        "type"
    ],
 "filter": [{
        "property": "type",
        "value": ["blog"],
        "comparison": "ct",
        "type": "list"},
       {
        "property": "tags",
        "value": ["Blog1"],
        "comparison": "ct",
        "type": "list" },
       {
        "property":"archived",
        "value":false,
        "type":"boolean",
        "comparison":"eq"}
   ]

 }
</bs:timeline>
```
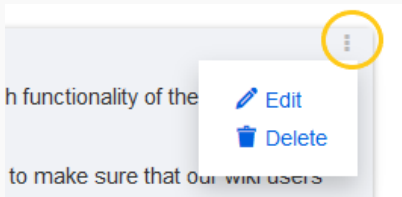
## Blog parameters

### Output format

The following table shows the blog elements and the necessary paramaters to create the output.



Blog output

| Element | Parameters | Parameter and element description | Type | Def... va... |
|---|---|---|---|---|
| **1 - header** | headlinemessagekey | Header text | string | *Tim...* |
| | showheadline | Show the header text | boolean | *true...* |
| **2 - blog menu** | showentitylistmenu | Show or hide the blog menu | boolean | *true...* |
| | showentityspawner | Show the menu item for creating a blog entry | boolean | *true...* |
| | availablesorterfields | Determines which data fields are included as sorting options in the sort menu. Example: allow sorting only by date created or by the user who created a blog entry:<br><br>`"availablesorterfields": ["timestampcreated","ownerid"]`<br><br><br><br>List of typical sort fields | array | all |
| | lockedoptionnames | Determines which settings for sorting the user cannot change. | array | [] |
| | availablefilterfields | Determines which data fields are included as filter options in the filter menu. Example: allow filtering based on Creation date and Linked pages:<br><br>`"availablefilterfields": ["timestampcreated","tags"]`<br><br> | array | all |
| | | Determines which settings for filtering the user cannot change. | | |

| Element | Parameters | Parameter and element description | Type | Def va |
|---|---|---|---|---|
| | lockedfilternames | Example: a user cannot change the filter values for *Linked pages.* The text field is greyed out:<br><br>`"lockedfilternames": ["tags"]`<br><br>**Creation date to** ⏣<br>22.11.2019<br>**Linked pages** ⏣ | array | [] |
| | persistsettings | Initial settings and custom sort and filter settings are included in the page url. The url can then be used to create a link to the page with the same filter and sort settings.<br><br>`"persistsettings":true`<br><br>myblog?sort=[{"property":"timestampcreated","direction":"DESC"}]&filter=[{"type":"list","property":"type","value":["l | boolean | *fals* |
| 3 - new blog entry | preloadentities | Shows the text field for creating a new blog entry. If it is not set, the text field is not available.<br><br>To show the text field:<br><br>`"preloadedentities": [{"type":"blog"}]`<br><br>If the wiki page that has a `<bs:timeline>` tag should only show its own blog entries, then all blog entries need to be automatically tagged with the current page name:<br><br>`"preloadedentities": [{`<br>`        "type": "blog",`<br>`        "tags":["Blog1"]`<br>`    }]` | array of objects | - |

| Element | Parameters | Parameter and element description | Type | Def<br>va |
|---|---|---|---|---|
| | | In the example above, `Blog1` is the associated page for all blog entries that are created on this page. The blog filter has to be adjusted accordingly to take the associated page into consideration. | | |
| 4 - existing blog entry | outputtypes | It defines the display format of the blog entry.<br><br>Standard types are: *List, Short, Default, Page.* (see examples)<br><br>Each blog entry is a "social entity". | object | *Det* |
| | limit | Defines how many social entities are loaded initially. | object | *Det* |
| 4a - avatar | - | Shows the avatar of the user who created the blog entry or comment. It is displayed in all output types except *List*. | - | - |
| 4b - info area | - | Link to the comments and "last edit" info. | - | - |
| 4c - blog entry content | - | Shows the text body of a blog entry. The output type *Default* collapses the blog entry content after the first paragraph with a "More" link. | - | - |
| 4d - actiion menu | - | Actions: *edit, delete, recommend, Linked page, watch*<br><br> | - | - |
| 5 - new comment | - | Text field for creating a new comment. | - | - |
| 6 - existing comment | - | Comments include the social tag *Recommend.*<br><br>Each comment is a "social entity". | - | - |
| | morelink | Link target of the "more" button. For blogs, it makes sense to link to the page *Special:Blog*. | str | *Spe*<br>*Tim* |
| | | More blog entries are automatically loaded after the user | | |

| Element | Parameters | Parameter and element description | Type | Default value |
|---|---|---|---|---|
| 7 - "more" button | useendlessscroll | scrolls to the end of the blog. | boolean | *true* |
| | showentitylistmore | Shows a "more" button below the blog which loads more blog entries on click. This is only possible if *useendlessscroll* is set to *false*. | boolean | *fals* |

## Sorting

Most commonly, a blog shows the latest blog entries first. However, there are many different options for sorting the list of blog entries.

### Syntax

```
[{
"property": "timestamptouched",
"direction": "DESC"
}]
```

Properties

There are some common properties of a blog entry that are useful for sorting:

Property	Description
timestampcreated	creation date
timestamptouched	last edit date
commentcount	number of comments
ratingcount	number of ratings
ownerid	username of the creator of a blog entry
ownerrealname	real name of the user

Direction

Blog entries can be sorted in descending and ascending order. Descending is the default order.

Property	Description
DESC	sorts the blog entries in descending order
ASC	sorts the blog entries in ascending order

Filtering

Without setting the filter parameter, there will be no output on the page.

To show the blog entries of your wiki, include the following codeblock like shown in the code example above:

```
"filter": [ {
        "type": "list",
        "property": "type",
        "value": [ "blog" ],
        "comparison": "ct"
    }]
```

This filter includes all social entities which match the type *blog*. Since we only want to show social entities of type *blog*, we only include *blog* entries in our filter. If you want to create a social timeline rather than a blog view, you can also add other types of social entities. The following example shows a timeline view that includes blog entries, discussion entries from wiki pages, and user profile information:

```
"filter": [ {
        "type": "list",
        "property": "type",
        "value": ["blog","topic","profile"],
        "comparison": "ct"
    }]
```

Filtering associated pages

If an embedded blog should only show posts that are associated with a paritcular page name (e. g., Blog1), this must be defined in the blog parameters (see *preloadedentities*) and then specified in the filter:

```
"filter": [ {
        "property": "type",
        "value": ["blog"],
        "comparison": "ct",
        "type": "list"},
      {
        "property": "tags",
        "value": ["Blog1"],
        "comparison": "ct",
        "type": "list" },
      {
```

```
        "property":"archived",
        "value":false,
        "type":"boolean",
        "comparison":"eq"}
    ]
```
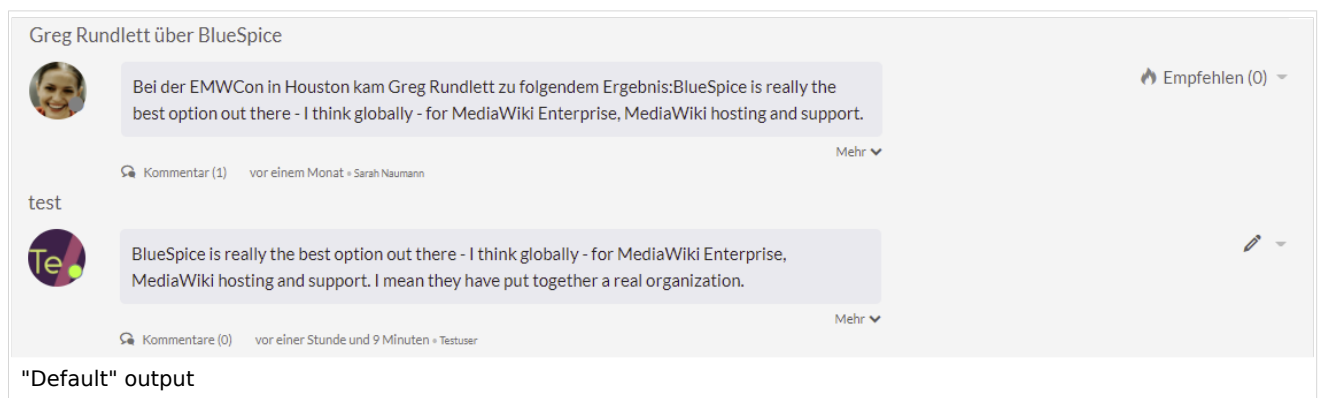
Output type examples

The output type for a blog is added using the *outputtypes* parameter. Don't forget to include a comma before and after this parameter (unless this is the last parameter in your code):

```
"outputtypes": {
        "blog": "Default"
    }
```

Default

The output type *Default* shows the content of a blog entry truncated after the first paragraph. To view the full text, users can click on the *More* link.
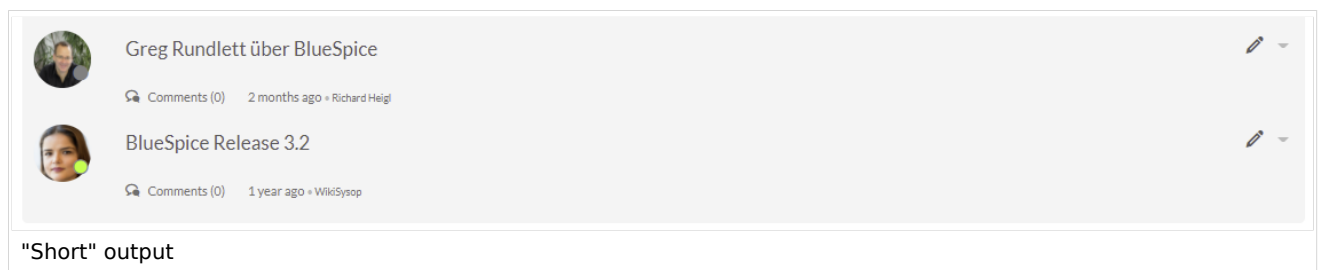
"Default" output

Short

The output type *Short* does not show the content of the blog entry. To view the content, a user has to click on the title of a blog entry. This loads the actual page for the selected blog entity.
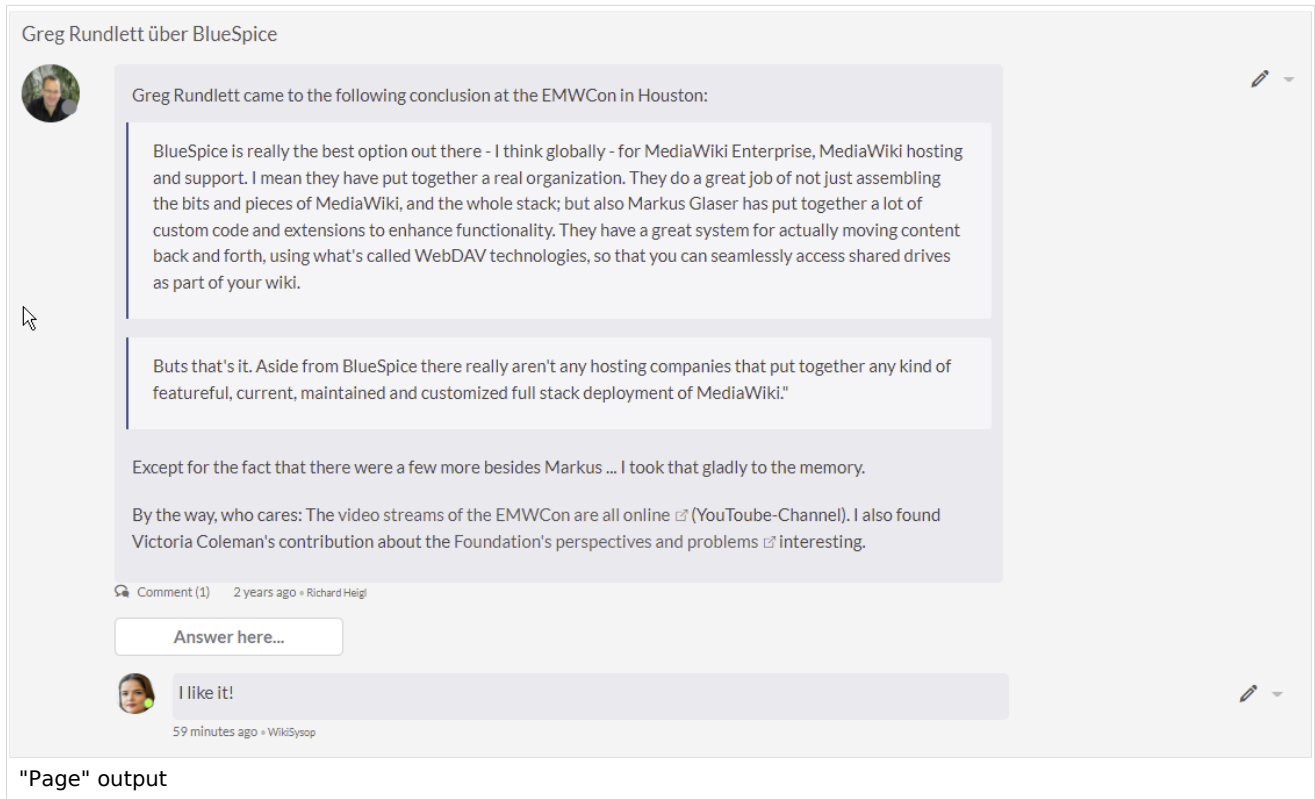
"Short" output

Page

The output type *Page* shows all blog elements, including the comments, in an expanded view.

"Page" output

List

The output type *List* only shows the title and the timestamp.

"List" output

Setting blog permissions

Using the "commenter" role

In general, all users with edit rights can create blog entries and comments. However, the *commenter* role in the Permission manager does not allow you to create blog entries. This role only assigns the right to create comments on existing blog entries. See also BlueSpiceSocial permissions.

Using groups

> ✱ is available from BlueSpice 4.2.

If a blog is integrated into a page, the creation and editing of discussion posts and comments can be restricted to certain user groups. This option does not exist on the *Special:Blog* page itself.

Add the blog to a wiki page as follows. Then, modify the *editgroups*, c*ommentgroups*, *deletegroups,* and r*eadgroups* as needed.

> **Important!** Only set the groups that you really need, since individual rights are not inherited. Therefore, please read the following explanations carefully!

```
<bs:timeline>
{
    "showentitylistmenu": true,
      "preloadedentities": [{
       "type": "groupblog",
       "editgroups": ["blog_GF"],
       "commentgroups": ["blog_commenter"],
       "deletegroups": ["blog_delete"],
       "readgroups": ["blog_reader","blog_GF","blog_commenter","blog_delete"]
    }],
    "headlinemessagekey": "My Blog",
    "showheadline": true,
    "usemorescroll": true,
    "limit": 5,
    "sort": [ {
        "property": "timestampcreated",
        "direction": "DESC"
    }],
    "lockedfilternames": [
        "type"
    ],
    "filter": [ {
        "type": "list",
        "property": "type",
        "value": [ "groupblog" ],
        "comparison": "ct"
    }]
}
</bs:timeline>
```

The object *preloadedentities* defines the different permissions:

Name	Value (examples)	Description
type	groupblog	The type *groupblog* allows setting group-based permissions on a blog.
editgroups	["blog_GF"]	Permission to create blog entries
commentgroups	["blog_commenter"]	Permission to create comments

Name	Value (examples)	Description
deletegroups	["blog_delete"]	Permission to delete blog entries and comments
readgroups	["blog_reader"," blog_GF"," blog_commenter"," blog_delete"]	Read permission: All groups listed in editgroups, commentgroups and deletegroups must also be listed here explicitly if this group is defined.

Important notes:

1. If the blog already exists on a page and the group assignments are created later, the permissions will not be applied to existing blog entries.
2. There is no inheritance if a group definition is given explicitly. E.g., a user group that is in editgroups but not in commentgroups (if exists) cannot add comments.

"editgroups" exists:

- Only users in these groups see the button to start a new blog entry.

"commentgroups" exists:

- Only users in these groups see the "answer here" button and can add comments.

"deletegroups" exists:

- Only users in these groups see the delete link and can delete o**ther users'** posts. Note: "sysop" users can always delete or restore a post. See below for sysop user read permissions.

"readgroups" exists:

- Only set if necessary, as there is no inheritance. If present, all user groups of the other blog groups must be added here.

Examples:

- If the *blog_GF* user group is in *editgroups* but not in *readgroups*, the create button is not displayed despite explicit editing rights. If the *blog_commenter* user group is in *commentgroups* but not in *readgroups*, the new blog entries that have this setting applied will not be visible. If the user group *blog_delete* is in *deletegroups* but not in *readgroups*, the new blog entries are not visible and therefore cannot be deleted.

user group "sysop":

- If *readgroups* is not defined, group sysop can delete all blog entries
- If *readgroups* is defined and sysop is not part of this group, sysop cannot see blog posts and therefore cannot delete them from the page. But with a direct link to the actual blog page, the sysop user can delete/restore the entry from there.

> **Important!** The default user group "sysop" cannot be added as a user group to a definition. For example, to assign read rights to an admin user (if "readgroups" exists), the user must be added to a user group defined in "readgroups" (e.g. blog_reader)

more"-button:

To avoid switching to the default *Special:Blog* page at the end of the blog, the m*ore*-button must be set as a scroll button instead of linking to the special page. To do this, set (as already indicated in the example):

```
"usemorescroll": true
```

Configuration

In the Config manager, you can change the following settings:

- **Use teaser text for blog entries in the timeline:** When you create a blog entry, you can create a teaser text for it by clicking the cogwheel symbol in the blog entry form. If this setting is active, the teaser text will be shown on the page *Special:Timeline* instead of the first part of the content.

blog teaser text

- **Show entrypoint "Blog":** Shows a link to the page *Special:Blog* in the main navigation.

Notifications

If users want to receive notifications about blog posts in the wiki or by email, the setting "Entity actions on watched pages" must be activated in the user settings ("Notifications" tab).

Notification settings

Show a list of recent blog entries

If you want to display the latest blog entries of a specific blog (one that is associated with a particular page), for example on the main page, the following steps are necessary:

1. Embed the timeline tag on the main page
2. Define formatting instructions in *MediaWiki: Common.css*

Recent blog posts

Embed the timeline

Copy the following timeline tag to the main page (or any other wiki page)

```
<div class="bloglist"><bs:timeline>
 {
    "showentitylistmenu": false,
    "preloadtitles": {
     "blog": "Vorlage:Blog"
    },
    "limit": 5,
    "sort": [ {
        "property": "timestampcreated",
        "direction": "DESC"
    }],
    "showentitylistmore": false,
    "availablefilterfields": [],
    "lockedfilternames": [
        "type"
    ],
    "outputtypes": {
        "blog": "List"
```

```
      },
    "filter": [ {
      "property": "type",
      "value": ["blog"],
      "comparison": "ct",
      "type": "list"},
      {
      "property": "tags",
      "value": ["Blog1"],
      "comparison": "ct",
      "type": "list" },
      {
      "property":"archived",
      "value":false,
      "type":"boolean",
      "comparison":"eq"}
  ]
 }
</bs:timeline>
... [[Blog1|all entries]]
</div>
```

Note that the value **Blog1** in the filter and after the timeline tag as a link to the actual blog page must be replaced with the real name of your blog page. For the value *Blog1* or the name of your blog pageto be automatically generated for all blog posts on the actual blog page, it must be specified in the *<bs:timeline>* tag under the preloadedentities object as the value for the key "tags":

```
"preloadedentities": [{
    "type": "blog",
    "tags":["Blog1"]
}]
```

Define formatting instructions

Copy the following css-declarations into *MediaWiki:Common.css*:

```
/* Simple list view of timeline items */
.bloglist .bs-social-entity .bs-social-entity-right .bs-social-entity-actions, .bs-
social-entity-timecreated {display:none}
.bloglist  ul.bs-social-entitylist {background-color:transparent!important; list-style-
type:disc; padding:0!important}
.bloglist  ul.bs-social-entitylist li {margin:0!important; border-bottom:1px dotted #74
747488}
.bloglist  .bs-social-entity div.bs-social-entity-right {margin: 0}
.bloglist  .bs-social-entity-actions {display:none!important}
.bloglist  .bs-social-entity .bs-social-entity-title h3 {font-size: 1em!important; marg
in:0;}
.bloglist  .bs-social-entity-title {width:100%; margin:0!important}
#content .bloglist  .bs-social-entity-right a, #content .bloglist .bs-social-entity-
right a::before {color:#08529d!important}
```

If you don't see the styles applied as in the screenshot above, you might have to do a page refresh (Ctrl+F5)

Related info

- [Micro-Blog](#)
- [Reference:BlueSpiceSocial](#)

➡ [Technical Reference: BlueSpiceSocialBlog](#)