

Tips for Wiki Editors

Useful tips for editing wiki content

- [Keyboard Shortcuts](#)
- [Reusing Page Content](#)
- [Searching Content](#)

Keyboard Shortcuts

The following shortcuts are available in the WYSIWYG Editor:

Shortcut (Windows & Linux/Mac)	Description
Ctrl+S / Cmd+S	Save Draft
Ctrl+Enter / Cmd+Enter	Save Page & Continue
Ctrl+B / Cmd+B	Bold
Ctrl+I / Cmd+I	Italic
Ctrl+U / Cmd+U	Underline
Ctrl+1 / Cmd+1 Ctrl+2 / Cmd+2 Ctrl+3 / Cmd+3 Ctrl+4 / Cmd+4	Header Large Header Medium Header Small Header Tiny
Ctrl+5 / Cmd+5 Ctrl+D / Cmd+D	Normal Paragraph
Ctrl+6 / Cmd+6 Ctrl+Q / Cmd+Q	Blockquote
Ctrl+7 / Cmd+7 Ctrl+E / Cmd+E	Code Block
Ctrl+8 / Cmd+8 Ctrl+Shift+E / Cmd+Shift+E	Inline Code
Ctrl+9 / Cmd+9	Callout (Keep pressing to toggle through styles)

Reusing Page Content

How to use include tags to reuse pages and page content within other pages.

You may find that you want to include the same block of content in multiple places. You could copy and paste that content into multiple pages but then, if that content needs to be changed, you'd have to update it multiple times across different pages. By using include tags you can include entire other pages and also include single blocks of content from other pages.

Note that the include behaviour is non-recursive so including will only work to a single level. This ensures performance and prevents users from breaking pages by creating include loops.

Include Tags

To include the content of a page within another you can use the following syntax:

```
{{@page_id}} or {{@page_id#content_id}}
```

You simply enter that in the page editor and then, when the page is viewed, the content will be dynamically be fetched. This means you can update the referenced content and it will show the same changes on any page the content is included within.

Easily Grabbing the Syntax

Finding the id of a page can be difficult so a quick and easy way to find include tags is built-in. View a page with some content you want to re-use. Use your mouse to select some content and you will see a popup appear. By default this contains a direct link to that content. If you click the link icon on the left it will switch to show you the include tag for that content. Simply copy that and place it in another page. Remove the `#` and content id to include the whole content of the page instead of just a single block.

Forcing Content IDs

When including a specific block of content the id of the block is used (part after the hash). By default these are generated when you save a page. If you are using the WYSIWYG editor the id will stay the same unless the block is removed or re-formatted. In the markdown editor it changes with every save based on the content so while it may remain the same there's a good chance an ID can change therefore it's recommended to force an ID if you're using the content for includes.

Note: All ID values must start with "bkmrk" to avoid being overwritten.

WYSIWYG Editor

In the WYSIWYG editor you can force an ID by going into the source code view and adding it manually:

BackToggle header

Set ChangelogSave Page

Chapter child page

FormatsB I

Some text i want

Playing with balls happened, find er innocent. I am the but gets sympathy) cat, little cat purr didn't give me any rainbow while flyi always looking da in. Flee in terror a hang jaw half ope balcony gives ow then plays in gras the cat knocks ov water but claws in

Continual woody sociables seabed welsh raunchy eig supernova. Monta twitched burdenir fictional sorest pe incarnated prefat bluebottles luxuriously depoliticise updater titbit.

Source code

<p id="bkmrk-important-text-to-include">Some text i want to include on another page.</p><p id="bkmrk-playing-with-balls-o">Playing with balls of wool chase the pig around the house throwup on your pillow attack the dog then pretend like nothing happened, find empty spot in cupboard and sleep all day refuse to leave cardboard box rub whiskers on bare skin act innocent. I am the best if it smells like fish eat as much as you wish missing until dinner time, yet fall over dead (not really but gets sympathy) spit up on light gray carpet instead of adjacent linoleum then cats take over the world. Purrr purr littel cat, little cat purr purr sweet beast run outside as soon as door open yet destroy couch as revenge for my slave human didn't give me any food so i pooped on the floor and please stop looking at your phone and pet me. Lick sellotape pooping rainbow while flying in a toasted bread costume in space so poop on grasses, or leave fur on owners clothes. Tuxedo cats always looking dapper purr for no reason. Sit on human sit by the fire russian blue but have secret plans or meow to be let in. Flee in terror at cucumber discovered on floor. Swat at dog hide when guests come over, yet sniff other cat's butt and hang jaw half open thereafter. Meow destroy couch sun bathe sleep on dog bed, force dog to sleep on floor jumps off balcony gives owner dead mouse at present then poops in litter box snatches yarn and fights with dog cat chases laser then plays in grass finds tiny spot in cupboard and sleeps all day jumps in bathtub and meows when owner fills food dish the cat knocks over the food dish cat slides down the water slide and into pool and swims even though it does not like water but claws in your leg. Sit and stare.</p><p id="bkmrk-continual-woody-suns">Continual woody sunset proudly stranglehold optical intuited ninepins guinean. Cockles dumping hungering austerest sociables seabeds outgrow dimmers setups distrustful outsets. Smackers genomes yachtsman marty arty wavelets welsh raunchy eighteenth plymouth commercialises. Hysterectomies necktie alkaloids blockaded quilters ferret supernova. Montague alcoholics belabouring ventriloquists proboscides compatriots nonstandard. Frenchman unwisely twitched burdening rewrite. Rampage kissing mcclain practicing imply stockades. Renew badmouthing ye potpourri fictional sorest penknife couplets dionysus daring mumbled marigolds. Mealest hearkening schoolchildren income incarnated prefabs jarred voldemort quondam wilful yachtzee nashville tarpaulins. {{@ 15#bkmrk-cat-cat-cat-attack}} bluebottles luxuriously depoliticise updater titbit.</p>

OkCancel

Searching Content

The ability to search your documentation is vital to day-to-day use.

There are a few locations within BookStack where you can search for your content. Below is a list of search functions within BookStack:

- **Header Search Bar** - The search bar/link in the header of every page allows you to search from anywhere. This search is a global search which will look across all books, chapters and pages in your system. After performing a search in this box you'll be led to a search page that includes options and features that can help you build a more advanced search.
- **Book/Chapter Search Bar** - When viewing a book or chapter a search bar can be found in the top of the left sidebar. These searches will look across all child items.
- **Move & Link Selection** - When choosing to move a page/chapter or when selecting a page/chapter/book to link to within the editor the most popular items are shown but you also have the ability to search.

Advanced Search Syntax

All of the above search locations within BookStack share the ability to use advanced search syntax. An easy way to see this syntax in action is to use the global search in BookStack then play with the search filters which will update the search term with the below syntax. Below are details of the different types of syntax that can be used:

Search Type	Syntax	Examples	Description
Normal Searches	<term_a> <term_b>	london meeting	Normal word searches across the name and description or body of your content. When multiple terms are searched only one term has to match your content but content containing both terms will be higher in the results.
Exact Searches	"<term>"	"london meeting"	Exact matches will require that the whole string within quotes exists in your content in exactly the same format. Use this if you're looking for an exact phrase containing or if you need to search for a term with spaces in.
Tag Searches	[<name>] [<operator><value>] [<name><operator><value>]	[location] [=london] [location=london] [attendees>5]	Tag searches allow you to find pages which have specific tags applied. You can search by tag name, by tag value or by both name and value. When searching by tag value an operator must be used to define the match type. You can use <code>=</code> , <code>!=</code> , <code><</code> , <code>></code> , <code><=</code> , <code>>=</code> or <code>like</code> as operators. When using the <code>like</code> operator you can use <code>%</code> symbols to represent wildcards in your search.
Filter Searches	{<filter_name>} {<filter_name>:<filter_value>}	See below	Filters perform additional advanced functionality to make your searches even more powerful. Some filters take values but some don't need to. See below for a full list of filters available.

Available Filters

Filters are set advanced search features that can be used in your search term. The below table shows all the filters available in BookStack and how they can be used.

Syntax	Examples	Description
Date Filters		
{updated_after:<date>}	{updated_after:2016-12-30}	Adds the condition that the content must have been last updated after the given date. The date should be in the format YYYY-MM-DD
{updated_before:<date>}	{updated_before:2016-12-30}	Adds the condition that the content must have been last updated before the given date. The date should be in the format YYYY-MM-DD
{created_after:<date>}	{created_after:2016-12-30}	Adds the condition that the content must have been created after the given date. The date should be in the format YYYY-MM-DD
{created_before:<date>}	{created_before:2016-12-30}	Adds the condition that the content must have been created before the given date. The date should be in the format YYYY-MM-DD
User Filters		
{updated_by:<user_slug me>}	{updated_by:barry} {updated_by:me}	Adds the condition that the content must have been last updated by the user of the given slug. If 'me' is used in place of a slug then it will find content that was last updated by the current logged-in user.
{created_by:<user_slug me>}	{created_by:barry} {created_by:me}	Adds the condition that the content must have been created by the user of the given slug. If 'me' is used in place of a slug then it will find content that was created by the current logged-in user.
{owned_by:<user_slug me>}	{owned_by:barry} {owned_by:me}	Adds the condition that the content must have be actively owned by the user of the given slug. If 'me' is used in place of a slug then it will find content that is owned by the current logged-in user.
Content Filters		
{in_name:<search>}	{in_name:London Meetings} {in_name:Meetings}	Will require the content to have the given <search> term in the name rather than the name or content body.
{in_body:<search>}	{in_body:London Meetings} {in_body:Meetings}	Will require the content to have the given <search> term in the body rather than both the name or content body.
Option Filters		
{is_restricted}	{is_restricted}	Will require the content to have content-level permissions active. Does not return items with only inherited asset permissions.
{viewed_by_me}	{viewed_by_me}	Will require the content to have been viewed by the current user at least once.
{not_viewed_by_me}	{not_viewed_by_me}	Will not return any content that has been viewed by the current user.

<code>{type:<content_types>}</code>	<code>{type:page chapter book}</code> <code>{type:page chapter}</code> <code>{type:book}</code>	Restricts the types of content that will be in the search results. Use of this will depend on the type of search. For example, in a chapter search only pages are shown so this has no effect.
---	---	---

Search Examples

Below are some examples of using the above syntax and filters with descriptions:

- `"my cat" {viewed_by_me} {updated_after: 2017-01-24}`
 - `"my cat"` - Search for content containing the exact phrase 'my cat'
 - `{viewed_by_me}` - that has been viewed by me
 - `{updated_after: 2017-01-24}` - and was last updated after the 24th of Jan 2017.
- `textbook discussion [meeting] {type: page} {created_by: me}`
 - `textbook discussion` - Search content for the words `textbook` or `discussion`
 - `[meeting]` - only show content that has a `meeting` tag applied
 - `{type: page}` - only show pages, hide chapters and books
 - `{created_by: me}` - that was created by me.
- `{type: book| chapter} {created_by: me} {created_after: 2016-08-12} {created_before: 2017-02-18}`
 - `{type: book| chapter}` - Search all books and chapters
 - `{created_by: me}` - that were created by me
 - `{created_after: 2016-08-12}` - after the 12th of Aug 2016
 - `{created_before: 2017-02-18}` - but before the 18th of Feb 2017