

menu macro

The {menu}, {submenu}, {sub-submenu} and {menuitem} macros are all the same macro! The different names are merely to aid legibility of menu definitions by allowing semantically/heirarchically correct structures to be built in the following form:

- Menu
 - Submenu
 - Sub-Submenu
 - Menuitem

Each of these macros creates an item in your menu, grouping any sub-items (if found) in to a "pop-up" menu of their own, which can contain further items and pop-up menus.

There are a few other features provided by these macros, such as automatic hiding of items you don't have access to and auto-cleaning of unwanted separators, but we'll discuss those in the examples at the bottom of this page.

Requirements

This macro requires [Theme Builder 2.0](#) or above.

Usage


To recreate the hierarchical structure shown above, use the following notation:

```
{menu}Menu
  {submenu}Submenu
    {sub-submenu}Sub-Submenu
      {menuitem}Menuitem{menuitem}
    {sub-submenu}
  {submenu}
{menu}
```

We'll show examples of common structures in the examples at the bottom of this page.

Parameters

Property	Required	Default	Notes	
class	✗		The CSS class name to assign to an item in a menu	
subclass	✗		The CSS class name to assign to a pop-up menu that's associated with an item in a menu	
id	✗		The HTML ID (must be unique for the entire web page) associated with an item in the menu	
subid	✗		The HTML ID (must be unique for entire web page) associated to a pop-up menu that's associated with an item in a menu	
autohide	✗	true	The following values are permitted: <ul style="list-style-type: none">• <i>true</i> – the item will be removed if it doesn't contain a hyperlink• <i>false</i> – the item will always be shown regardless of whether it contains a hyperlink or not• <i>disabled</i> – the item will always be shown, but if it doesn't contain a hyperlink it will be "visually" disabled (e.g. grey text and faded icon)	
custom	✗	false	The following values are permitted: <ul style="list-style-type: none">• <i>false</i> – a normal menu item, which may contain a pop-up menu if there are hierarchically lower items found• <i>true</i> – allows you to embed just about anything in a menu item, defaults the "autohide" setting to <i>false</i> and prevents you from having a pop-up menu associated with the item	
flat	✗	false	The following values are permitted: <ul style="list-style-type: none">• <i>false</i> – a normal menu item designed for use in a hierarchical structure• <i>true</i> – a "flat" (non-hierarchical) menu item	
hideanon	✗	false	Hide the contents of the menu from anonymous users	
flag	✗		The macro is rendered only if one or more of the specified flags are set. See Working with Flags for more details.	3.3.6

notflag			The macro is rendered only if <i>none</i> of the specified flags are set. See Working with Flags for more details.	3. 3.6
---------	---	--	--	-----------

Examples

Basic Use

The most basic use of these macros is to output a single menu item:

```
{menuitem}[Home Page/Go to the Home Page]{menuitem}
```

We've used a wiki notation link in the example above to create a link to the Home page within the current space. You can use any valid wiki notation link inside a menu item so you could, for example, link to a news item, different space or even a user profile.

It should be noted that if you link to a page in a menu item, and then rename that page, the menu item will not update itself to use the new name of the page. You can get round this problem by using the [wikimenu macro](#) which allows you to create a normal bullet list of links on a page (which will be updated if you rename pages) and then import them in to your menu.

For all common locations within Confluence, we recommend using the [menulink macro](#) as it will be more reliable, especially if any of the common locations are changed in a later version of Confluence. For example, to create an ultra-reliable link to a home page, use:

```
{menuitem}{menulink:home/tooltip=Go to the Home Page}Home Page{menulink}{menuitem}
```

This might seem like overkill, but it really is far, far more reliable to link to common Confluence locations this way. Anyway...

You can have several items at the same level, for example:

```
{menuitem}{menulink:home}Home Page{menulink}{menuitem}
{menuitem}{menulink:dashboard}Dashboard{menulink}{menuitem}
```

As you can see, the items are merely output as an unordered list. To put them in to an actual menu bar, simply wrap them in the [menubar macro](#)¹ as shown below:

```
{menubar:id=demo1}
{menuitem}{menulink:home}Home Page{menulink}{menuitem}
{menuitem}{menulink:dashboard}Dashboard{menulink}{menuitem}
{menubar}
```

1. Remember to specify a unique ID for the menubar macro, otherwise it won't work!

As you can see, the items are shown next to each other because they are at the same hierarchical level. To create pop-up sub menus, you need to create a hierarchical structure...

Hierarchical Menus

To create hierarchical structures, you wrap menu items in the other macros as shown earlier, for example:

```

{menuitem}{menulink:home}Home Page{menulink}{menuitem}
{menuitem}{menulink:dashboard}Dashboard{menulink}{menuitem}
{menu}Level 1
  {menuitem}{menulink:home}Home Page{menulink}{menuitem}
  {menuitem}{menulink:dashboard}Dashboard{menulink}{menuitem}
  {submenu}Level 2
    {menuitem}{menulink:home}Home Page{menulink}{menuitem}
    {menuitem}{menulink:dashboard}Dashboard{menulink}{menuitem}
    {sub-submenu}Level 3
      {menuitem}{menulink:home}Home Page{menulink}{menuitem}
      {menuitem}{menulink:dashboard}Dashboard{menulink}{menuitem}
    {sub-submenu}
  {submenu}
{menu}

```

Here's another quick example to show that you can "indent" the menu structure at any point, not just the last item in a level:

```

{menuitem}{menulink:home}Home Page{menulink}{menuitem}
{menu}Level 1
  {menuitem}{menulink:home}Home Page{menulink}{menuitem}
{menu}
{menuitem}{menulink:dashboard}Dashboard{menulink}{menuitem}

```

CSS Customisation

See [Customising Menu Styles](#).