Code Block Macro

The **Code Block Macro** allows you to display source code in your document with the appropriate syntax highlighting. The code block displays on the page as shown below.

```java
class App {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

On this page:
- Usage with the Macro Browser
- Usage with the Wiki Markup Editor
- Parameters
- Examples

Usage with the Macro Browser

To insert the code block macro into a page using the Macro Browser,

1. Open the Confluence page or blog post that you want to edit, then click the 'Edit' button.
2. Click the Macro Browser icon on the toolbar. The macro browser window will open.
3. Scroll through the list of macros to find the one you want. Alternatively, start typing the macro name into the search box at the top right of the macro browser. Macros with a matching name will appear in the main pane.
4. Click the desired macro.
5. Set the macro parameters to your requirements. If desired, you can preview these changes by clicking 'Refresh'.
6. Click 'Insert' to add the macro onto the page.

You can also insert macros via autocomplete. For more information, see Using Autocomplete in the Rich Text Editor.

Once you've found the code block macro, click 'Insert' to add it to your page.

Usage with the Wiki Markup Editor

<table>
<thead>
<tr>
<th>What you need to type</th>
<th>What you will get</th>
</tr>
</thead>
</table>
| `{code}
public static void main(String[] args)
{args}` | `public static void main(String[] args)` |

Parameters

Parameters are options that you can include in Confluence macros to control the content or format of the macro output. The table below lists relevant parameters for this macro.

Parameter names are different in the macro browser and in wiki markup. Below we show the macro browser parameter names in **bold** text, and the equivalent wiki markup parameters in *(bracketed)* text. If we do not show any parameter name for the wiki markup, then you should leave out the parameter name and simply include the parameter value as the first parameter, immediately after the colon (`).`

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Required?</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
</table>

Syntax highlighting

Specifies the language (or environment) for syntax highlighting. The default language is **Java** but you can also choose from one of the following other languages/environments:

- ActionScript3
- Bash
- C# (csharp)
- ColdFusion
- C++ (cpp)
- CSS
- Delphi
- Diff
- Erlang
- Groovy
- JavaFX
- JavaScript
- Perl
- PHP
- PowerShell
- Python
- Ruby
- Scala
- SQL
- Visual Basic (vb)
- XML (including HTML)
- ... or no syntax highlighting (none).

Title

Adds a title to the code macro. If specified, the title will be displayed in its own row at the top of your code macro box.

Collapsible

If true, the code macro's content will be collapsed upon visiting or refreshing the Confluence page. Clicking the 'expand source' link allows you to view this content. If false, the code macro's content is always displayed in full.

Show line numbers

If true, a line number will be shown to the left of each line of code, where each number is incremented by 1. If false, no line numbers are shown.

First line number

When Show line numbers (linenumbers) is true, this value defines the number of the first line of code.

Theme

Specifies the colour scheme used for displaying your code. Many of these themes are based on the default colour schemes of popular Integrated Development Environments (IDEs). The default theme is **Confluence** (also known as **Default**), which is typically black and coloured text on a blank background. However, you can also choose from one of the following other popular themes:

- Django
- Emacs
- FadeToGrey
- Midnight
- RDark
- Eclipse
- Confluence.

Be aware that any white space contained between the `{code}` commands is not manipulated in any way by the Code Block Macro. This is to provide the writer with flexibility over code indentation.

ℹ️ All the optional parameters of the `{panel}` macro are valid for the `{code}` macro as well.

### Examples

#### Example 1: Java

<table>
<thead>
<tr>
<th>What you need to type</th>
<th>What you will get</th>
</tr>
</thead>
</table>
| `{code}
public String getFoo()
{
    return foo;
}
` | public String getFoo()
{
    return foo;
} |

#### Example 2: XML

<table>
<thead>
<tr>
<th>What you need to type</th>
<th>What you will get</th>
</tr>
</thead>
</table>
| `{code}
<foo>
    <bar>
        <baz/>
    </bar>
</foo>` | `<foo>
    <bar>
        <baz/>
    </bar>
</foo>` |
<test>
    <another tag="attribute"/>
</test>

RELATED TOPICS

Working with Macros

Take me back to the Wikis Help Guide.