Wiki プロセッサ

プロセッサは Wiki マクロ の一種です。 Trac Wiki エンジン 以外のマークアップフォーマットを使用するために設計されています。プロセッサはユーザが編集したテキストを処理するマクロ機能 と考えることができます。

Trac 内の全ての Wiki テキストにおいて、 Wiki プロセッサを使用することができます。 例えば、以下に述べる様々な用途で使用することができます:

- シンタックスハイライト またはテキストを逐語的にレンダリングする,
- ブロックや や <td>, <th> のテーブルのセルの内容など コンテキスト内の Wiki のマークアップをレンダリングする,
- HTML や Restructured テキスト または textile のシンタックスをマークアップの代替手段として使用する

プロセッサを使用する

テキストブロック上に Wiki プロセッサを適用するには、 最初のデリミタとして Wiki の コードブロック を使用します:

```
{{{
■■■■■■■
■■■■■■■■■■■...
}}}
```

{{{ のすぐ後または直後の行に, #! に続けて プロセッサ名 を追加します。.

```
{{{
#!■■■■■■■
■■■■■■■■
■■■■■■■■■■■...
}}}
```

これはほとんどの UNIX ユーザにはおなじみである "shebang" を用いた表記法です。

これらの内容に加えて、いくつかの Wiki プロセッサでは、 プロセッサ名に続けて同じ行に、 key=value のペアを書くことによって、 パラメータを許容するものがあります。もし、 value にスペースが含まれるケースがたびたび起こるようであれば、
クオテーションで囲んだ文字列を使用することができます (key="■■■■■■■■■■■■■■■■■") 。

いくつかのプロセッサは Wiki のマークアップを処理することを意味するので、 プロセッサのブロックを ネスト させることができます。
可読性を向上させるためにネストしたブロックの内容をインデントさせたいと思うかもしれません。
この種のインデントは中身を処理する際に無視されます。

例

Wiki マークアップ	表示
例 1: HTML を挿入する	
{{{ HTML ■■■ }}}	HTML テスト
例 2: ブロック内のカスタマイズしたスタイルシートでハイライトした Python のコード	

{{{#!div style="background: #ffd; border: 3px ridge"
 	
■■■ ■■■■■■■ "■■■" ■■■■■■■■■:
 	
{{{
#!python
def hello():
return "world"
}}}
 	
}}}
	これは 埋め込まれた "コード" ブロックの例です:
	
	def hello():
	return "world"

例 3: Wiki ページでチケットのキーワード検索を行なう

```
{{{
#!html
<form action="/query" method="get"><div>
<input type="text" name="keywords" value="~" size="30"/>
<input type="submit" value="■■■■■■■■■"/>
<!-- To control what fields show up use hidden fields
<input type="hidden" name="col" value="id"/>
<input type="hidden" name="col" value="summary"/>
<input type="hidden" name="col" value="status"/>
<input type="hidden" name="col" value="milestone"/>
<input type="hidden" name="col" value="version"/>
<input type="hidden" name="col" value="owner"/>
<input type="hidden" name="col" value="priority"/>
<input type="hidden" name="col" value="component"/>
-->
</div></form>
}}}
```

プロセッサー覧

以下のプロセッサが Trac の配布に含まれています:

`#!default`

フォーマット済みのテキストブロック内でテキストを逐語的に表示する。 これはプロセッサ名を指定 しない のと同じです。 (さらに `#!` を指定しないのと同じです)

`#!comment`

このセクション内のテキストを処理しません(つまり、このセクションの内容は プレインテキスト内にのみ存在します - レンダリングしたページには表示されません)。

HTML 関連

`#!html`

Wiki ページにカスタムした HTML を挿入する。

`#!htmlcomment`

Wiki ページに HTML のコメントを挿入する (0.12 以降) 。

`#!html` ブロックは 自己完結型 でなければなりません。 つまり、1つ目のブロックである HTML 要素を開始し、後に記述する2つ目のブロックでその要素を閉じるということはできません。これに類似した効果を得るためには、下記に述べるプロセッサを使用し

`#!div`

 要素内の任意の Wiki の内容をラップする (0.11 以降).

`#!span`

 要素内の任意の Wiki の内容をラップする (0.11 以降).

`#!td`

<td> 要素内の任意の Wiki の内容をラップする (0.12 以降)

`#!th`

<th> 要素内の任意の Wiki の内容をラップする (0.12 以降)

`#!tr`

`#!td` や `#!th` ブロックをラッピングしたり、グルーピングした列を見やすくするために属性を指定する際、オプションとして使用することができます。 (0.12 以降)

これらのプロセッサについての詳細と使用例については、 WikiHtml を参照して下さい。

その他のプロセッサ

`#!rst`

Trac における Restructured テキストのサポート. WikiRestructuredText 参照。

#!textile

Textile がインストール されている場合のサポート。 a Textile リファレンス 参照。

コードのハイライトのサポート

Trac はインラインのシンタックスハイライトを提供するプロセッサを同梱しています:

#!c (C), #!cpp (C++), #!python (Python), #!perl (Perl), #!ruby (Ruby), #!php (PHP), #!asp (ASP), #!java (Java), #!js (Javascript), #!sql (SQL), #!xml (XML または HTML), #!sh (Bourne/Bash シェル), など。

Trac では、シンタックスのカラーリングについて外部のソフトウェア 例えば Pygments に依存しています。

TracSyntaxColoring にどの言語のシンタックスハイライトがサポートされているかについての情報と
その他の言語のシンタックスハイライトのサポートを有効にする方法について書かれています。

MIME
タイプをプロセッサ名として使用することによって、ソースコードをブラウズするときに、サポートしている言語についてはシンタックスハイライトをすることが可

```
{{{
#!text/html
<h1>■■■■</h1>
}}}
```

この結果はシンタックスハイライトされた HTML コードになります:

```
<h1>text</h1>
```

これは、対応しているすべての 対応している MIME タイプ に対して有効です。

ユーザによるプロセッサマクロの開発やコントリビュート:

- ProcessorBazaar
- MacroBazaar
- Trac Hacks コミュニティサイト

プロセッサは Wiki マクロを開発するのと変わりはありません。 実際、開発方法は同じであり、シンタックスの使用方法のみが異なります。
詳細情報については、 カスタムマクロを開発する を参照して下さい。

See also: WikiMacros, WikiHtml, WikiRestructuredText, TracSyntaxColoring, WikiFormatting, TracGuide