Wikiprint Book

Title: Trac マクロ

Subject: SilverFrost - WikiMacros

Version: 3

Date: 06/04/26 07:07:14

SilverFrost 目次

Trac マクロ

Trac マクロとは、 Python で書かれた 'カスタム関数' によって Trac の Wiki エンジンを拡張するプラグインです。 WikiFormatting エンジンが利用可能なあらゆるコンテキストにおいて、マクロを使用することによって、動的な HTML データが挿入されます。

もう 1 種類のマクロは WikiProcessors です。これは通常、 Wiki 以外のマークアップ形式と表示を取り扱うために使用し、多くは、(ソースコードハイライトのような) より大きいブロックに使用します。

マクロの利用

マクロ呼び出しは、二つの 角括弧 (square brackets) で括られた箇所です。 Python 関数のように、マクロは引数を取ることができ、括弧 (parenthesis) の中に、カンマで区切ったリストで表記します。

詳細なヘルプを見るには

マクロの一覧と完全なヘルプは、 下記のマクロ一覧 にある MacroList マクロを使用してみることができます。

簡単なマクロ一覧は `[[MacroList(*)]]` や `[[?]]` で見ることができます。

特定のマクロの詳細なヘルプを参照したい場合は、 MacroList マクロに引数渡すことによって参照することができます。 例) `[[MacroList(MacroList)]]` 。もしくは、便宜上、 `[[MacroList?]]` のようにマクロ名にクエスチョンマーク ('?') をつけることでヘルプをみることができます。

利用例

'Trac' で始まる Wiki ページの最近の変更履歴 3 件分を表示するマクロです:

Wiki マークアップ	表示
`[[RecentChanges(Trac,3)]]`	2012/10/20 • TracIni (diff) • TracRepositoryAdmin (diff) • TracSearch (diff)

`[[RecentChanges?(Trac,3)]]`	`[[RecentChanges]]`
	最近更新されたページを最終更新の日付でグループ化して一覧表示します。
	このマクロは2つのパラメータと1つの名前付き引数を受け取ります。名前付き引数は、任意の順番で指定することができます。
	1つ目はプレフィックス文字列で、指定している場合、その結果はそのプレフィックスで始まるページ名のみとなります。このパラメータを省略した場合はすべてのページを表示します。
	2つ目のパラメータは、結果のページ数に対する上限値になります。
	`group` パラメータによって一覧をどのように表示を行うかが決まります。
	`group=date`
	ページを日付ごとに箇条書きで表示します (デフォルト)。
	`group=none`
	ページを1つの箇条書きにて表示します。
	ヒント: プレフィックス文字列でフィルタを行わずに項目の最大数のみを指定したい場合には、1つ目のパラメータを空にします。例えば `[[RecentChanges(,10,group=none)]]` とします。

`[[?]]`	`[[Image]]`
	画像を Wiki 形式のテキストに組み込みます。 1 番目の引数は、ファイル名を指定します。ファイルの指定は添付ファイルなど …
	`[[InterTrac]]`
	既知の InterTrac プレフィックスをリスト形式で表示します。
	`[[InterWiki]]`
	既知の InterWiki プレフィックスに関する概要のリストを表示します。
	`[[KnownMimeTypes]]`
	WikiProcessors で処理できる既知の mime-type を表示します。 引数が与えられた場合は、 mime-type ...
	etc.

マクロ一覧

Note: 以下に示すリストはマクロドキュメントを含むものだけです。 *-OO* による最適化や、 mod_python での *PythonOptimize* オプションが設定されていると表示されません。

[[AddComment]]

A macro to add comments to a page. Usage:

```
[[AddComment]]
```

The macro accepts one optional argument that allows appending to the wiki page even though user may not have modify permission:

```
[[AddComment(appendonly)]]
```

[[ChangeLog]]

Write repository change log to output.

The ChangeLog macro writes a log of the last changes of a repository at a given path. Following variants are possible to use:

```
1. [[ChangeLog([reponame:]path)]]
2. [[ChangeLog([reponame:]path@rev)]]
3. [[ChangeLog([reponame:]path@rev, limit)]]
4. [[ChangeLog([reponame:]path@from-to)]]
5. [[ChangeLog([reponame:]path, limit, rev)]]
```

1. Default repository is used if reponame is left out. To show the last five changes of the default repository:

   ```
   [[ChangeLog(/)]]
   ```

   To show the last five changes of the trunk folder in a named repo:

   ```
   [[ChangeLog(otherrepo:/trunk)]]
   ```

2. The ending revision can be set. To show the last five changes up to revision 99:

   ```
   [[ChangeLog(otherrepo:/trunk@99)]]
   ```

3. The limit can be set by an optional parameter. To show the last 10 changes, up to revision 99:

   ```
   [[ChangeLog(otherrepo:/trunk@99, 10)]]
   ```

4. A range of revisions can be logged.

   ```
   [[ChangeLog(otherrepo:/trunk@90-99)]]
   ```

   To lists all changes:

   ```
   [[ChangeLog(otherrepo:/trunk@1-HEAD)]]
   ```

   HEAD can be left out:

   ```
   [[ChangeLog(otherrepo:/trunk@1-)]]
   ```

5. For backwards compatibility, revision can be stated as a third parameter:

   ```
   [[ChangeLog(otherrepo:/trunk, 10, 99)]]
   ```

limit and rev may be keyword arguments.

```
[[ChangeLog(otherrepo:/trunk, limit=10, rev=99)]]
```

[[ExtractUrl]]

Provides test macro for the `tracextracturl.extract_url` function.

This macro is intended for code testing by the developers of the above function and has no real usage for normal Trac users.

Macro usage: `[[ExtractUrl(traclink)]]`
Result: The URL extracted by `extract_url`

$Id: macro.py 8545 2010-08-30 21:57:33Z martin_s $

Description for extract_url()

Extracts an URL from an Wiki link, e.g. to used in macro produced HTML code.

Website: http://trac-hacks.org/wiki/ExtractUrlPlugin

`$Id: extracturl.py 8545 2010-08-30 21:57:33Z martin_s $`

Description

Returns an (possible relative) URL which can be used in HTML code.

If `raw` is true the returned link will point to a downloadable version of the linked resource otherwise the same link is returned which would be used in the resulting Wiki page.

The raw links are also usable as online resouces, e.g. if the link target is to be used as input for a flash application etc.

Usage

General:

```
from tracextracturl import extract_url
# ...
 url = extract_url (env, context, wikilink, raw=False)
```

Inside WikiMacros:

```
from tracextracturl import extract_url

def MyMacro(WikiMacroBase):
 def expand_macro (self, formatter, name, content):
    # e.g. wikilink = 'wiki:WikiStart' or 'attachment:file.ext'
    url = extract_url(self.env, formatter.context, wikilink)
    rawurl = extract_url(self.env, formatter.context, wikilink, True)
```

Example

Inside a Trac macro called from the wiki page 'ExamplePage' of project 'project1' on a multi-project trac server:

```
    extract_url(self.env, formatter, 'attachment:file.js', True)
```

will return `/project1/raw-attachment/wiki/ExamplePage/file.js`, which could be directly accessed by the browser inside some javascript or flash HTML object code produced by the macro.

[[FootNote]]

Collates and generates foot-notes. Call the macro with the foot-note content as the only argument:

```
[[FootNote(This is a footnote)]]
```

Foot-notes are numbered by the order in which they appear. To create a reference to an existing foot-note, pass the footnote number as argument to the macro:

```
[[FootNote(1)]]
```

In addition, identical foot-notes are coalesced into one entry. The following will generate one footnote entry with two references:

```
Some text[[FootNote(A footnote)]] and some more text [[FootNote(A footnote)]].
```

A list of footnotes generated by one or more of the above commands is produced by calling the macro without arguments:

```
[[FootNote]]
```

Once a set of footnotes has been displayed, a complete new set of footnotes can be created. This allows multiple sets of footnotes per page.

[[Image]]

Wiki 形式のテキストに画像を埋め込むことができます。

最初の引数にはファイル名を指定します。ファイルの指定には3つの方法で添付ファイルを参照することができます。

- `module:id:file` の場合は、module には wiki や ticket のどちらかを指定でき、指定している Wiki ページやチケットにある file という名前の添付ファイルを参照します。
- `id:file`: 上記と同じですが、id にはチケットの省略形式、もしくは Wiki ページ名のどちらかになります。
- `file` の場合は、ローカルの 'file' という名前の添付ファイルを参照します。これは Wiki ページやチケットでだけ機能します。

また、`source:file` 形式 (`source:file@rev` でも) を使ってリポジトリのファイルを参照することもできます。

直接 URL を記述してアクセスすることもできます。`/file` はプロジェクトの相対 URL、`//file` はサーバの相対 URL、`http://server/file` はファイルの完全 URL となります。

残りの引数は省略可能で表示する `<img>` 要素の属性とスタイルを設定することができます。

- 数値と単位付きの数値は画像に対するサイズだと解釈します (例 120, 25%)。
- `right`、`left`、`center`、`top`、`bottom`、`middle` は画像の位置合わせに使います。(別の方法としては、最初の3つは `align=...` を使って、最後の3つは `valign=...` を使って指定することができます)
- `link=some TracLinks...` は TracLinks を使って指定のリンクで画像へのリンクを置き換えることができます。値を指定していない場合は、単にリンクを削除します。
- `nolink` は画像へのリンクなしを意味します (非推奨、`link=` を使ってください)。
  `key=value` 形式は HTML 属性または CSS スタイルの指定だと解釈します。有効なキーは次のものです。
  - align、valign、border、width、height、alt、title、longdesc、class、margin、margin-(left,right,top,bottom)、id、usemap
  - `border`、`margin`、`margin-*` には1つの数値だけが指定できます。
  - `margin` は `center` がマージンとして使う auto より優先します。

例:

```
 [[Image(photo.jpg)]]                    # ■■■■■■
[[Image(photo.jpg, 120px)]]              # ■■■■■■■■
[[Image(photo.jpg, right)]]              # ■■■■■■■■■■
[[Image(photo.jpg, nolink)]]             # ■■■■■■■■■■■■■
[[Image(photo.jpg, align=right)]]        # ■■■■■■■
```

他のページやチケット、他のモジュールの画像を使うことができます。

```
    [[Image(OtherPage:foo.bmp)]]    # ■■■■■■■■■■ Wiki ■■■
   [[Image(base/sub:bar.bmp)]]     # ■■■■■■■■ Wiki ■■■■■■
   [[Image(#3:baz.bmp)]]           # #3 ■■■■■■■■■■■■■■
   [[Image(ticket:36:boo.jpg)]]
   [[Image(source:/images/bee.jpg)]] # ■■■■■■■■■■
   [[Image(htdocs:foo/bar.png)]]   # ■■■■■■■■ htdocs ■■■■■■■■■■■■■■■■
```

Shun-ichi Goto <gotoh@…> が作成した Image.py マクロが元になっています。

[[InterTrac]]

設定済みの InterTrac プレフィックスの一覧です。

[[InterWiki]]

設定済みの InterWiki プレフィックスの一覧です。

[[KnownMimeTypes]]

WikiProcessors として使用できる mime-type の一覧です。

引数を指定すると mime-type をフィルタリングすることができます。

[[Lineno]]

Wiki processor that prints line numbers for code blocks.

Example:

```
{{{
#!Lineno
#!java
class A {
   public static void main(String[] args) {
   }
}
}}}
```

行	
1	class A {
2	public static void main(String[] args) {
3	}
4	}

[[ListPage]]

現在のページ、または指定されたページのサブページの一覧を表示します。

使い方:

```
[[ListPage([■■■■■■■[, ■■[, ■■■■■■■■]]])]]
```

引数:

1. ■■■■■■■ (string) - 元にするページ。現在のページを指定するには空。トップを指定するには /。
2. ■■ (int) - 表示する深さ。現在の階層のみの場合は 0。全てを指定する場合(デフォルト)は *。
3. ■■■■■■■■ (bool) - Wiki ページ内の =XXX= で指定された文字列をタイトルとして使用するかどうか。デフォルトは False。

例:

```
[[ListPage]]
[[ListPage(/,1)]]
```

[[ListTagged]]

タグ付けされているリソースを表示します。

使用方法:

```
[[ListTagged(query)]]
```

クエリの構文については tags のドキュメントを参照してください。

[[MacroList]]

インストールされている Wiki マクロの一覧と利用可能ならドキュメントも表示します。

マクロの名前を引数に指定することができ、この場合はそのマクロのドキュメントだけを表示します。

※ このマクロは mod_python の PythonOptimize オプションを有効にしているとマクロのドキュメントを表示できません。

[[MilestonDescription]]

指定したマイルストーンの説明を表示します。

使い方:

```
[[MilestonDescription(■■■■■■■■)]]
```

引数:

1. ■■■■■■■■ (string) - 表示する対象のマイルストーン。

例:

```
[[MilestonDescription(1.0)]]
```

[[MilestoneCompact]]

Adds a wiki macro [[MilestoneCompact]] which lists and describes the project's milestones in a compact form.

[[MindMap]]

エイリアス: [[Mindmap]]

Website: http://trac-hacks.org/wiki/MindMapMacro

$Id: macro.py 9152 2010-09-26 20:59:50Z martin_s $

[[PageName]]

現在のページ名を表示します。

使い方:

```
[[PageName([■■■■■■■■■])]]
```

引数:

1. ■■■■■■■■■ (bool) - フルパスの場合(デフォルト)は True。それ以外の場合は False。

例:

```
[[PageName]]
[[PageName(False)]]
```

[[PageOutline]]

現在の Wiki ページのアウトライン構造を表示します。アウトラインの各項目は対応する見出しへのリンクとなります。

このマクロは4つの省略可能なパラメータを受け取ります。

- 1つ目は、数値または数値の範囲でアウトラインに表示する見出しの最小レベルと最大レベルを設定できます。 例えば、"1" を指定するとアウトラインにはトップレベルの見出しだけになります。 "2-3" を指定するとレベル2と3のすべての見出しがアウトラインに含まれるようになります。
- 2つ目のパラメータは、タイトルを指定することができます (デフォルトではタイトルはなしです)。
- 3つ目のパラメータは、アウトラインのスタイルを選択します。 `inline` または `pullout` のどちらかにすることができます (後者がデフォルトです)。 `inline` スタイルは通常のコンテンツの一部としてアウトラインを表示しますが、 `pullout` ではデフォルトでコンテンツの右にフロートしているボックスの中にアウトラインを表示します。
- 4つ目のパラメータはアウトラインを番号付きにするかどうかを指定します。`numbered` または `unnumbered` のどちらかにすることができます (前者がデフォルトです)。このパラメータは `inline` スタイルでのみ効果があります。

[[ProgressMeter]]

Progress meter wiki macro plugin for Trac

Usage instructions are available at:

http://trac-hacks.org/wiki/ProgressMeterMacro

[[QueryChart]]

Draw Chart from Query.

```
 {{{
[[QueryChart(args1,args2,...)]]
}}}

args:

* query: Search condition of ticket. The following three kinds of can be described.
  * [http://trac.edgewall.org/wiki/TracQuery#QueryLanguage Query language] notation of TicketQuery macro:[[BR]]
    Write the condition without applying ? to the head.
    Refer to the [http://trac.edgewall.org/wiki/TracQuery#QueryLanguage Query language] for details.[[BR]]
    !query:status=new|assigned&version;^=1.0
  * Notation displayed in URL with custom query:[[BR]]
    Write conditions delimited by & applying ? (Without forgetting :) to the head as follows. It might be
    good to put the part of URL specifying the condition on the screen of custom query.[[BR]]
    !query:?status=new&status;=assigned&version;=^1.0
  * Omitte:[[BR]]
    It is possible to omit it only when putting it on the column the explanation of the report made from
    custom query (which displayed in address field of a browser). It becomes a search condition specified the
    omission on the screen. Please omit this item (query:...) when omitting it.
* col: Targeted item. Please describe by col=xxx, col=yyy, and switching off the comma district when you
  specify the plural. The item name can specify both the field name (name of the field in Trac) and the label
  (displayed item name).
* per:(=day,week,free) Unit of total. Default is week.
* start: Day in left end of graph. If it is unspecification, it is the most past day of the ticket. yyyy/mm/dd form
* end: Day on right edge in graph. If it is unspecification, it is the most recent day of the ticket. yyyy/mm/dd form
* width: Width in graph. It specifies it by the unit of px. If it is unspecification, it is 536px.
* height: Height of graph. It specifies it by the unit of px. If it is unspecification,    [[BR]]it is 300px.
* upper: The improvement chart is written (bug settling curve etc.). Down chart when not specifying it.
```

[[RecentChanges]]

最近更新されたページを最終更新の日付でグループ化して一覧表示します。

このマクロは2つのパラメータと1つの名前付き引数を受け取ります。名前付き引数は、任意の順番で指定することができます。

1つ目はプレフィックス文字列で、指定している場合、その結果はそのプレフィックスで始まるページ名のみとなります。このパラメータを省略した場合はすべての

2つ目のパラメータは、結果のページ数に対する上限値になります。

`group` パラメータによって一覧をどのように表示を行うかが決まります。

`group=date`
: ページを日付ごとに箇条書きで表示します (デフォルト)。

`group=none`
: ページを1つの箇条書きにて表示します。

ヒント: プレフィックス文字列でフィルタを行わずに項目の最大数のみを指定したい場合には、1つ目のパラメータを空にします。例えば `[[RecentChanges(,10,group=none)]]` とします。

[[RepositoryIndex]]

利用可能なリポジトリの一覧を表示します。

次の名前付き引数を指定できます。

format
: 表示するフォーマットを選択します。
  - compact はリポジトリのプレフィックス名をカンマ区切りで表示します (デフォルト)
  - list はリポジトリのプレフィックス名を一覧表示します
  - table はリポジトリブラウザページのものと同じようなテーブル形式で表示します

glob
: リポジトリ名に対して glob 形式のフィルタリングを行います (デフォルトは '*' です)

order
: リポジトリを指定のカラム順で並べます ("name", "date" または "author" の1つ)

desc
: 1を指定すると降順で並べます

(0.12 以降)

[[TagCloud]]

タグクラウドを表示します。

クエリに該当するリソースからタグクラウドを表示します。

使用方法:

```
[[TagCloud(query,caseless_sort=<bool>,mincount=<n>)]]
```

caseless_sort
: 大文字小文字を区別しないでソートするかどうかを指定します。

mincount
: 短いタグを非表示にする際の閾値を指定します。(省略可)

クエリの構文については tags のドキュメントを参照してください。

[[TicketQuery]]

Wiki macro listing tickets that match certain criteria.

このマクロの引数はカンマ区切りのリストにした、 "key=value" 形式の キー付きパラメータを使用します。

key がフィールド名であった場合、 value は TracQuery#QueryLanguage で
定義されているような、フィルタを指定するシンタックスでなければなりません。 ? の文字で始まる query: リンク向けの簡素化した URL
シンタックス とは 異なります 。フィールドの値としてカンマ (,) そのものを含む場合は バックスラッシュ (,) でエスケープする必要があります。

引数 or で区切られたフィルタのグループは、 OR 条件で結合されます。

このほか、フィルタとしていくつか名前付きパラメータを使用できます。
これらは検索結果をどのように表示するかを制御できます。すべて非必須です。

format はチケットのリストがどのように表示されるかを決定します:

- list -- デフォルトの表示形式です。チケット ID と概要 (Summary) を 一覧表示します。 1 行ごとに 1 つのチケットを表示します。
- compact -- チケット ID の一覧をカンマ区切りで 表示します。
- count -- 条件に当てはまるチケットの件数のみが表示されます。
- table -- カスタムクエリービューと似た形式で表示されます (ただし コントロールは表示されません)。
- progress -- a view similar to the milestone progress bars

max は表示されるチケット数の上限値を指定します (デフォルトは 0 です。これは無制限を意味します)。

order はチケットを整列する列を指定します (デフォルトは id となっています)。

desc はチケットの整列を逆順に行うか指定します。 (デフォルトは false です)。

group はチケットをグループ化を指定します (デフォルトは何も設定されていません)。

groupdesc はグループの表示を逆順とするかを指定します (デフォルトは false となっています)。

verbose を true に設定すると、リストされたチケットの 各行にチケットの説明を表示します。これは table format 専用です。
このパラメータは廃止予定です。代わりに *rows* を使用してください。

rows パラメータは1行使って表示するフィールドを指定します。 rows=description|summary のように使用します。

Trac 0.10 との互換性のため、マクロは最終引数にキーなし引数が与えられた場合 format として解釈します。
また、フィールドセパレータに使用する "&" は (order を除いて) 現時点では動作しますが、この機能は廃止予定です。

[[TitleIndex]]

Wiki ページの一覧をアルファベット順で出力します。

引数としてプレフィックスになる文字列を受け取ります:
指定した場合、一覧はプレフィックスで始まるページ名だけになります。引数を省略した場合、すべてのページを表示します。プレフィックスを指定している場合、
hideprefix が指定でき、出力からそのプレフィックスのものが取り除かれます。

format と depth という名前のパラメータを指定することができます。

- format=compact: カンマ区切りのリンクでページ一覧を表示します。
- format=group: 共通のプレフィックスによりグループ化を行ってページ一覧を表示します。 またこのフォーマットは min=n
  引数をサポートしており、グループに対する最小ページ数になります。
- format=hierarchy: ページ名の階層により構造化を行ってページ一覧を表示します。 またこのフォーマットは min=n
  引数をサポートしており、n 以上の階層がフラットになります。
- depth=n: ページ一覧の深さを制限します。 0を指定した場合はトップレベルのページだけを表示し、
  1を指定した場合は直接の子になるページまでを表示するなどとなります。
  指定なし、または-1を指定した場合は、すべてのページを階層化して表示を行います。
- include=page1:page*2: コロン区切りで並べたページ名に該当するものだけになります。 リストが空、または include
  引数がない場合はすべてのページを含みます。
- exclude=page1:page*2: コロン区切りで並べたページ名に該当するものは除くようになります。

include と exclude はシェル形式のパターンを受け付けます。

[[TracAdminHelp]]

trac-admin コマンドのヘルプを表示します。

例:

```
[[TracAdminHelp]]               # ■■■■■■■■■
[[TracAdminHelp(wiki)]]         # ■■■■■ wiki ■■■■■
[[TracAdminHelp(wiki export)]]  # "wiki export" ■■■■■
[[TracAdminHelp(upgrade)]]      # upgrade ■■■■■
```

[[TracGuideToc]]

Trac ガイドの目次を表示します。

このマクロはやっつけな方法で Help/Guide の目次を作成します。 目次は Trac* と WikiFormatting ページを含んでいてカスタマイズできません。
目次のカスタマイズには TocMacro を検索してください。

[[TracIni]]

Trac 設定ファイルのドキュメントを生成します。

これは通常 TracIni
ページで使います。引数を指定するとセクションとオプションの名前のフィルタになり、そのフィルタの内容で始まるセクションとオプションのみを出力します。

[[Tsvn]]

TortoiseSVN 用のリンクを表示します。

使い方:

```
[[Tsvn(■■■■■■■URL)]]
```

引数:

1. ■■■■■■■URL (string) - リンク先のリポジトリのURL。

例:

```
[[Tsvn(http://yourhost/svn/project)]]
```

[[WorkTimeReport]]

以下のようにWikiにマクロを記載することで、作業時間のレポートを表示します。
また、オプションを設定することで、レポートの表からグラフを生成することもできます。

```
[[WorkTimeReport(args1,args2,...)]]
```

args:

- workers:
  表示するユーザIDを指定します。コロンに続けて、カンマ区切りで指定します。デフォルトでは全ユーザ(ただし、ログインユーザに権限によっては絞られます)
- showSummaryChart:
  表示する全ユーザの作業時間を集計グループ単位に円グラフで出力するかどうか指定します。未指定の場合は、trueとなります。
  - true
    円グラフを表示します。
  - false
    円グラフを表示しません。
- cutoffDay: 月単位の集計を行う場合の、締め日を指定します。指定可能な値は、1～31です。月末としたい場合は、[last]と指定します。
- monthlyReportCount:
  月単位の集計を行う場合に、表示する月数を指定します。例えば、3を指定することで直近3ヶ月の集計結果を表示することができます。デフォルトでは1となりま
- fromDay: 集計開始日を指定します。指定可能な値は、YYYY-mm-DD形式です。
- toDay: 集計終了日を指定します。指定可能な値は、YYYY-mm-DD形式です。
- title: タイトル文字列を指定することができます。コロンに続けてタイトル文字列を指定します。

- groupField: 集計に利用するチケットのフィールドを指定します。未指定の場合は、id(チケットID)となります。
- condition: 集計に利用する条件を指定します。コロンに続けて、条件を指定します。
  例えば、typeがtaskのみのチケットを条件としたい場合は、type=taskと指定します。
  複数の条件を指定したい場合は、&で区切ります。
- unit: 作業時間の集計値の形式を指定します。コロンに続けて、以下の記載が可能です。
  - hours: 時間
  - ratio: 割合
- graph: グラフも合わせて表示する場合に指定します。コロンに続けて、以下の記載が可能です。
  - lines
    折れ線グラフを表示します。
  - bars
    棒グラフを表示します。
- stack: グラフ表示時に、スタック(積み上げ)グラフとするかどうか指定します。コロンに続けて、以下の記載が可能です。
  - true
    スタック(積み上げ)グラフとして表示します。
  - false
    スタック(積み上げ)グラフとして表示しません。
- legendLoc: グラフの凡例の表示位置を指定します。コロンに続けて、方角を示す nw, n, ne, e, se, s, sw, w のいずれかを指定します。
- legendXOffset: グラフの凡例位置の横方向のオフセットを指定します。未指定の場合は12となります。
- legendYOffset: グラフの凡例位置の縦方向のオフセットを指定します。未指定の場合は12となります。
- width: グラフの幅をpx単位で指定します。未指定の場合は536pxとなります。
- height: グラフの高さをpx単位で指定します。未指定の場合は300pxとなります。
- table: テーブルを出力を制御できます。コロンに続けて、以下の記載が可能です。
  - hide
    テーブルを表示しません。グラフのみを表示する場合に使用します。
- async: 非同期でレポート、グラフを描画するかどうか指定します。コロンに続けて、以下の記載が可能です。未指定の場合は、trueとなります。
  - true
    非同期で描画します。
  - false
    非同期で描画しません。

例:

- 2009-11-01～2009-11-30のデータを表示します。

```
[[WorkTimeReport(fromDay:2009-11-01,toDay:2009-11-30)]]
```

- 2009-11-01～2009-11-30のデータを表示します。グラフ化もします。

```
[[WorkTimeReport(fromDay:2009-11-01,toDay:2009-11-30,graph:bars,stack:true)]]
```

[[Workflow]]

ワークフローを描画します。

このマクロは TracWorkflow 設定を受け取って有向グラフの状態と遷移を描画します。
パラメータを指定しない場合は、現在のワークフローを描画します。 Wiki プロセッサモードでは `width` と `height` を引数に指定することができます。

(デフォルト: `width = 800`, `heigth = 600`)

例:

```
    [[Workflow()]]

   [[Workflow(go = here -> there; return = there -> here)]]
```

```
{{{
#!Workflow width=700 height=700
leave = * -> *
leave.operations = leave_status
leave.default = 1

accept = new,assigned,accepted,reopened -> accepted
accept.permissions = TICKET_MODIFY
accept.operations = set_owner_to_self

resolve = new,assigned,accepted,reopened -> closed
resolve.permissions = TICKET_MODIFY
resolve.operations = set_resolution

reassign = new,assigned,accepted,reopened -> assigned
reassign.permissions = TICKET_MODIFY
reassign.operations = set_owner

reopen = closed -> reopened
reopen.permissions = TICKET_CREATE
reopen.operations = del_resolution
}}}
```

[[graphviz]]

The GraphvizPlugin (http://trac-hacks.org/wiki/GraphvizPlugin) provides a plugin for Trac to render graphviz (http://www.graphviz.org/) graph layouts within a Trac wiki page.

[[graphviz.dot/png]]

エイリアス: [[graphviz.dot/jpg]] [[graphviz.dot/gif]] [[graphviz.dot/svg]] [[graphviz.dot/svgz]] [[graphviz.dot]] [[graphviz.neato/png]] [[graphviz.neato/jpg]] [[graphviz.neato/gif]] [[graphviz.neato/svg]] [[graphviz.neato/svgz]] [[graphviz.neato]] [[graphviz.twopi/png]] [[graphviz.twopi/jpg]] [[graphviz.twopi/gif]] [[graphviz.twopi/svg]] [[graphviz.twopi/svgz]] [[graphviz.twopi]] [[graphviz.circo/png]] [[graphviz.circo/jpg]] [[graphviz.circo/gif]] [[graphviz.circo/svg]] [[graphviz.circo/svgz]] [[graphviz.circo]] [[graphviz.fdp/png]] [[graphviz.fdp/jpg]] [[graphviz.fdp/gif]] [[graphviz.fdp/svg]] [[graphviz.fdp/svgz]] [[graphviz.fdp]] [[graphviz/png]] [[graphviz/jpg]] [[graphviz/gif]] [[graphviz/svg]] [[graphviz/svgz]]

ドキュメントが見つかりません

[[plantuml]]

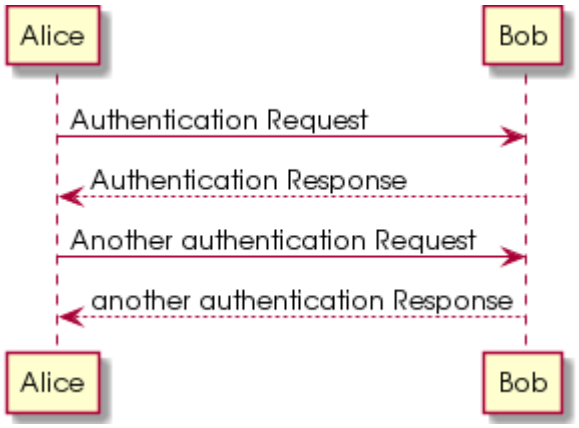
エイリアス: [[PlantUml]] [[PlantUML]]

A wiki processor that renders PlantUML diagrams in wiki text.

Example:

```
{{{
#!PlantUML
@startuml
Alice -> Bob: Authentication Reque
st
Bob --> Alice: Authentication Response
Alice -> Bob: Another authentication Request
Alice <-- Bob: another authentication Response
@enduml
}}}
```

Results in:



世界のマクロを共有

Trac Hacks というサイトは、コミュニティに寄稿されたマクロと プラグイン を収集し提供しています。新しいマクロを探している、共有したいマクロを作成した、などの場合は遠慮なく Trac Hacks のサイトを訪問してください。

カスタムマクロを開発する

マクロは、 Trac 本体と同様 Python で書かれています。そして TracPlugins の一種として開発します。

マクロの開発についての詳しい情報は リソースの開発 を参照してください。

Trac 0.11 でマクロを作成する簡単な例を 2 つ紹介します。

古いマクロと新しいマクロの違いを示す例は Timestamp.py を参照してください。また、古いマクロから新しいマクロに移行するための情報は macros/README を参照してください。

引数なしのマクロ

下記のソースコードをテストするためには、このソースコードを timestamp_sample.py として保存し、 TracEnvironment の plugins/ に配置しなければなりません。

```
from datetime import datetime
# Note: since Trac 0.11, datetime objects are used internally

from genshi.builder import tag

from trac.util.datefmt import format_datetime, utc
from trac.wiki.macros import WikiMacroBase

class TimeStampMacro(WikiMacroBase):
    """Inserts the current time (in seconds) into the wiki page."""

    revision = "$Rev$"
    url = "$URL$"

    def expand_macro(self, formatter, name, text):
        t = datetime.now(utc)
        return tag.b(format_datetime(t, '%c'))
```

引数付きのマクロ

下記のソースコードをテストするためには、このソースコードを helloworld_sample.py として保存し、 TracEnvironment の plugins/ に配置しなければなりません。

```
from genshi.core import Markup

from trac.wiki.macros import WikiMacroBase

class HelloWorldMacro(WikiMacroBase):
   """Simple HelloWorld macro.

   Note that the name of the class is meaningful:
    - it must end with "Macro"
    - what comes before "Macro" ends up being the macro name

   The documentation of the class (i.e. what you're reading)
   will become the documentation of the macro, as shown by
   the !MacroList macro (usually used in the WikiMacros page).
   """

   revision = "$Rev$"
   url = "$URL$"

   def expand_macro(self, formatter, name, text, args):
       """Return some output that will be displayed in the Wiki content.

       `name` is the actual name of the macro (no surprise, here it'll be
       `'HelloWorld'`),
       `text` is the text enclosed in parenthesis at the call of the macro.
         Note that if there are ''no'' parenthesis (like in, e.g.
         [[HelloWorld]]), then `text` is `None`.
       `args` are the arguments passed when HelloWorld is called using a
       `#!HelloWorld` code block.
       """
       return 'Hello World, text = %s, args = %s' % \
           (Markup.escape(text), Markup.escape(repr(args)))
```

Note: expand_macro **は 第4パラメータに、** *args* **を任意に取ることもできます。 このマクロが** WikiProcessor **として呼ばれたとき、** key=value **形式の** プロセッサパラメータ **を渡すことも可能です。もし、このパラメータを指定したとき、これらの値は、ディクショナリの中に保存され、追加の** args **パラメータによって渡されます。一方で、マクロとして呼び出されたときは、** args **パラメータは、** None **として扱われます (0.12 以降) 。**

例として、このように記述した場合:

```
{{{#!HelloWorld style="polite" -silent verbose
<Hello World!>
}}}

{{{#!HelloWorld
<Hello World!>
}}}

[[HelloWorld(<Hello World!>)]]
```

結果はこのようになります:

```
Hello World, text = <Hello World!> , args = {'style': u'polite', 'silent': False, 'verbose': True}
Hello World, text = <Hello World!> , args = {}
Hello World, text = <Hello World!> , args = None
```

Note: expand_macro **が返す値は、 HTML がエスケープされて いない ことに注意して下さい。期待する戻り値によっては、あなた自身でエスケープする必要があります (**return Markup.escape(result)**を使用できます)。また、戻り値として HTML が返ってくると分かっているならば、結果を (**return Markup(result)**) という風に Genshi が提供している Markup (**from genshi.core import Markup**) オブジェクトでラップすることもできます。**

また、text を Wiki としてマークアップする場合、 Wiki Formatter (from trac.wiki import Formatter)
オブジェクトも再帰的に使用することができます。以下がサンプルです:

```
from genshi.core import Markup
from trac.wiki.macros import WikiMacroBase
from trac.wiki import Formatter
import StringIO

class HelloWorldMacro(WikiMacroBase):
       def expand_macro(self, formatter, name, text, args):
               text = "whatever '''wiki''' markup you want, even containing other macros"
               # Convert Wiki markup to HTML, new style
               out = StringIO.StringIO()
               Formatter(self.env, formatter.context).format(text, out)
               return Markup(out.getvalue())
```