reStructuredText Support in Trac

Introduction

Trac supports [?reStructuredText (RST)](#) as an alternative to wiki markup where [WikiFormatting](#) is used.

From the reStucturedText webpage:

> "reStructuredText is an easy-to-read, what-you-see-is-what-you-get plaintext markup syntax and parser system. It is useful for in-line program documentation (such as Python docstrings), for quickly creating simple web pages, and for standalone documents. reStructuredText is designed for extensibility for specific application domains."

If you want a file from your Subversion repository to be displayed as reStructuredText in the Trac source browser, set `text/x-rst` as the value for the Subversion property `svn:mime-type`, or add the extension `rst` to the filename. See [?this example](#).

The examples will only be rendered as reStructuredText if docutils is installed. If Pygments is installed but docutils is not installed, the examples will be syntax-highlighted rather than rendered as reStructuredText.

Requirements

To activate RST support in Trac, install the python docutils package with the command `easy_install docutils`, or through your operating system package manager. If not already available on your operating system, you can download it from [?PyPI](#).

More information on RST

- [?reStructuredText Website](#)
- [?RST Quick Reference](#)

Using RST in Trac

To specify that a block of text should be parsed using RST, use the *rst* processor.

TracLinks in reStructuredText

- Trac provides a custom RST directive `trac::` to allow TracLinks from within RST text.

Wiki Markup	Display

`{{{#!rst`
`This is a reference to |a ticket|`
 	
`.. |a ticket| trac:: #42`
`}}}`
	This is a reference to |a ticket|
	
	  #42

- You can also use the custom `:trac:` role to create TracLinks in RST.

Wiki Markup	Display
```{{{#!rst` `This is a reference to ticket `#12`:trac:` `To learn how to use Trac, see `TracGuide`:trac:` `}}}```	This is a reference to ticket To learn how to use Trac, see

For a complete example of all uses of the :trac: role, see WikiRestructuredTextLinks.

Syntax highlighting in reStructuredText

There is a directive for doing TracSyntaxColoring in RST as well. The directive is called code-block:

Wiki Markup	Display
```{{{#!rst`  `.. code-block:: python`  ` class Test:`  ` def TestFunction(self):` ` pass`  `}}}```	

Note the need to indent the code at least one character after the `.. code-block` directive.

Wiki Macros in reStructuredText

To enable Wiki Macros in RST, you use the same `code-block` directive as for syntax highlighting:

Wiki Markup	Display
```{{{#!rst`  `.. code-block:: RecentChanges`  ` Trac,3`  `}}}```	

Or use the `:code-block:` role for a more concise Wiki Macro-like syntax:

Wiki Markup	Display
```{{{` `#!rst`  `:code-block:`RecentChanges:Trac,3`` `}}}```	

Bigger RST Example

The example below should be self-explanatory:

Wiki Markup	Display
```{{{#!rst` `FooBar Header`	reStructuredText is . It has its own webp

```
=============
reStructuredText is **nice**. It has its own webpage_.

A table:

=====  =====  ======
   Inputs     Output
------------  ------
  A      B    A or B
=====  =====  ======
False  False  False
True   False  True
False  True   True
True   True   True
=====  =====  ======

RST TracLinks
-------------

See also ticket `#42`:trac:.

.. _webpage: http://docutils.sourceforge.net/rst.html
}}}
```
```
A table:

=====  =====  ======
   Inputs     Output
------------  ------
  A      B    A or B
=====  =====  ======
False  False  False
True   False  True
False  True   True
True   True   True
=====  =====  ======



See also ticket .

  http://docutils.sourceforge.net/rst.htm
```

See also: WikiRestructuredTextLinks, WikiProcessors, WikiFormatting