reStructuredText Support in Trac

Trac supports using *reStructuredText* (RST) as an alternative to wiki markup in any context <u>WikiFormatting</u> 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.* "

Requirements

Note that to activate RST support in Trac, the python docutils package must be installed. If not already available on your operating system, you can download it at the <u>?RST Website</u>.

Install docutils using `easy_install docutils`. Do not use the package manager of your OS (e.g. `apt-get install python-docutils`), because Trac will not find docutils then.

More information on RST

- reStructuredText Website -- <u>?http://docutils.sourceforge.net/rst.html</u>
- RST Quick Reference -- <u>?http://docutils.sourceforge.net/docs/rst/quickref.html</u>

Using RST in Trac

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

<u>TracLinks</u> in reStructuredText

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

  Example:

  ```
  {{{
  #!rst
  This is a reference to |a ticket|
  ```

```
       .. |a ticket| trac:: #42
       }}}
```

- Trac allows an even easier way of creating <u>TracLinks</u> in RST, using the custom `:trac:` role.

  Example:

  ```
  {{{
  #!rst
  This is a reference to ticket `#12`:trac:

  To learn how to use Trac, see `TracGuide`:trac:
  }}}
  ```

  For a complete example of all uses of the `:trac:` role, please see <u>WikiRestructuredTextLinks</u>.

Syntax highlighting in reStructuredText

There is a directive for doing <u>TracSyntaxColoring</u> in RST as well. The directive is called code-block

Example

```
{{{
#!rst

.. code-block:: python

 class Test:

    def TestFunction(self):
        pass

}}}
```

Will result in the below.

```
.. code-block:: python

 class Test:

    def TestFunction(self):
        pass
```

Wiki Macros in reStructuredText

For doing <u>Wiki Macros</u> in RST you use the same directive as for syntax highlighting i.e code-block. To work you must use a version of trac that has <u>?#801</u> applied.

Wiki Macro Example

```
{{{
#!rst

.. code-block:: RecentChanges

   Trac,3

}}}
```

Will result in the below:

01/10/14

- ◆ <u>TracSupport</u> <u>(diff)</u>
- ◆ <u>TracUpgrade</u> <u>(diff)</u>
- ◆ <u>TracSyntaxColoring</u> <u>(diff)</u>

Or a more concise Wiki Macro like syntax is also available:

```
{{{
#!rst

:code-block:`RecentChanges:Trac,3`
}}}
```

Bigger RST Example

The example below should be mostly self-explanatory:

```
{{{
#!rst
FooBar Header
=============
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
}}}
```

Results in:

```
FooBar Header
=============
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
```

See also: WikiRestructuredTextLinks, WikiProcessors, WikiFormatting