

# reStructuredText Demonstration

*Optional Subtitle*

**David Goodger**

This document is a demonstration of the reStructuredText markup language, containing examples of all basic reStructuredText constructs and many advanced constructs.

January 03, 2012



# 1 Structural Elements

## 1.1 Section Title

That's it, the text just above this line.

## 1.2 Transitions

Here's a transition:

---

It divides the section.



## 2 Body Elements

### 2.1 Paragraphs

A paragraph.

#### 2.1.1 Inline Markup

Paragraphs contain text and may contain inline markup: *emphasis*, **strong emphasis**, inline literals, standalone hyperlinks (<http://www.python.org>), external hyperlinks (Python <sup>1</sup>), internal cross-references (*example*), external hyperlinks with embedded URIs (Python web site), footnote references (manually numbered <sup>2</sup>, anonymous auto-numbered <sup>3</sup>, labeled auto-numbered <sup>4</sup>, or symbolic <sup>5</sup>), citation references ([CIT2002]), substitution references (<sup>6</sup>), and inline hyperlink targets (see *Targets* below for a reference back to here). Character-level inline markup is also possible (although exceedingly ugly!) in *reStructuredText*. Problems are indicated by `|problematic|` text (generated by processing errors; this one is intentional).

The default role for interpreted text is *Title Reference*. Here are some explicit interpreted text roles: a PEP reference (PEP 287); an RFC reference (RFC 2822); a `subscript`; a `superscript`; and explicit roles for *standard inline* markup.

Let's test wrapping and whitespace significance in inline literals: This is an example of  
--inline-literal                      --text,                      --including                      some--  
strangely--hyphenated-words.      Adjust-the-width-of-your-browser-window  
to see how the text is wrapped.   -- ---- -----      Now note              the  
spacing              between the              words of              this sentence              (words should  
be grouped              in pairs).

If the `--pep-references` option was supplied, there should be a live link to PEP 258 here.

### 2.2 Bullet Lists

- A bullet list
  - Nested bullet list.
  - Nested item 2.
- Item 2.
  - Paragraph 2 of item 2.
    - Nested bullet list.
    - Nested item 2.
    - Third level.

---

<sup>1</sup> <http://www.python.org/>

<sup>2</sup> A footnote contains body elements, consistently indented by at least 3 spaces.  
This is the footnote's second paragraph.

<sup>3</sup> This footnote is numbered automatically and anonymously using a label of "#" only.

<sup>4</sup> Footnotes may be numbered, either manually (as in <sup>2</sup>) or automatically using a "#" -prefixed label. This footnote has a label so it can be referred to from multiple places, both as a footnote reference (<sup>4</sup>) and as a hyperlink reference (*label*).

<sup>5</sup> Footnotes may also use symbols, specified with a "\*" label. Here's a reference to the next footnote: <sup>6</sup>.

<sup>6</sup> This footnote shows the next symbol in the sequence.

[CIT2002] Citations are text-labeled footnotes. They may be rendered separately and differently from footnotes.

- Item 2.
- Nested item 3.

### 2.3 Enumerated Lists

1. Arabic numerals.
  1. lower alpha)
    1. (lower roman)
      1. upper alpha.
        1. upper roman)
2. Lists that don't start at 1:
  1. Three
  2. Four
  1. C
  2. D
  1. iii
  2. iv
3. List items may also be auto-enumerated.

### 2.4 Definition Lists

#### Term

Definition

#### Term : classifier

Definition paragraph 1.

Definition paragraph 2.

#### Term

Definition

### 2.5 Field Lists

**what** Field lists map field names to field bodies, like database records. They are often part of an extension syntax. They are an unambiguous variant of RFC 2822 fields.

**how arg1 arg2** The field marker is a colon, the field name, and a colon.

The field body may contain one or more body elements, indented relative to the field

marker.

## 2.6 Option Lists

For listing command-line options:

-a	command-line option "a"
-b <i>file</i>	options can have arguments and long descriptions
--long	options can be long also
--input= <i>file</i>	long options can also have arguments
--very-long-option	
	The description can also start on the next line.
	The description may contain multiple body elements, regardless of where it starts.
-x, -y, -z	Multiple options are an "option group".
-v, --verbose	Commonly-seen: short & long options.
-l <i>file</i> , --one= <i>file</i> , --two <i>file</i>	
	Multiple options with arguments.
/v	DOS/VMS-style options too

There must be at least two spaces between the option and the description.

## 2.7 Literal Blocks

Literal blocks are indicated with a double-colon (":") at the end of the preceding paragraph (over there -->). They can be indented:

```
if literal_block:
    text = 'is left as-is'
    spaces_and_linebreaks = 'are preserved'
    markup_processing = None
```

Or they can be quoted without indentation:

```
>> Great idea!
>
> Why didn't I think of that?
```

## 2.8 Line Blocks

This is a line block. It ends with a blank line.

Each new line begins with a vertical bar ("|").

Line breaks and initial indents are preserved.

Continuation lines are wrapped portions of long lines; they begin with a space in place of the vertical bar.

The left edge of a continuation line need not be aligned with the left edge of the text above it.

This is a second line block.

Blank lines are permitted internally, but they must begin with a "|".

Take it away, Eric the Orchestra Leader!

A one, two, a one two three four

Half a bee, philosophically,  
must, *ipso facto*, half not be.  
But half the bee has got to be,  
*vis a vis* its entity. D'you see?

But can a bee be said to be  
or not to be an entire bee,  
when half the bee is not a bee,  
due to some ancient injury?

Singing...

2.9 Block Quotes

Block quotes consist of indented body elements:

My theory by A. Elk. Brackets Miss, brackets. This theory goes as follows and begins now. All brontosauruses are thin at one end, much much thicker in the middle and then thin again at the far end. That is my theory, it is mine, and belongs to me and I own it, and what it is too.

—Anne Elk (Miss)

2.10 Doctest Blocks

```
>>> print 'Python-specific usage examples; begun with ">>>"'
Python-specific usage examples; begun with ">>>"
>>> print '(cut and pasted from interactive Python sessions)'
(cut and pasted from interactive Python sessions)
```

2.11 Tables

Here's a grid table followed by a simple table:

Header row, column 1 (header rows optional)	Header 2	Header 3	Header 4
body row 1, column 1	column 2	column 3	column 4
body row 2	Cells may span columns.		
body row 3	Cells may span rows.	<ul style="list-style-type: none"><li>• Table cells</li><li>• contain</li><li>• body elements.</li></ul>	
body row 4			
body row 5	Cells may also be empty: -->		

Inputs		Output
A	B	A or B
False	False	False
True	False	True
False	True	True
True	True	True



### *reStructuredText*

Figure 1. A figure is an image with a caption and/or a legend:

re	Revised, revisited, based on 're' module.
Structured	Structure-enhanced text, structuredtext.
Text	Well it is, isn't it?

This paragraph is also part of the legend.

## 2.12 Footnotes

## 2.13 Citations

Here's a reference to the above, [CIT2002], and a [\[nonexistent\]](#) citation.

## 2.14 Targets

This paragraph is pointed to by the explicit "example" target. A reference can be found under *Inline Markup*, above. *Inline hyperlink targets* are also possible.

Section headers are implicit targets, referred to by name. See *Targets*, which is a subsection of *Body Elements*.

Explicit external targets are interpolated into references such as "Python <sup>1</sup>".

Targets may be indirect and anonymous. Thus *this phrase* may also refer to the *Targets* section.

Here's a ``hyperlink reference without a target`_`, which generates an error.

### 2.14.1 Duplicate Target Names

Duplicate names in section headers or other implicit targets will generate "info" (level-1) system messages. Duplicate names in explicit targets will generate "warning" (level-2) system messages.

### 2.14.2 Duplicate Target Names

Since there are two "Duplicate Target Names" section headers, we cannot uniquely refer to either of them by name. If we try to (like this: ``Duplicate Target Names`_`), an error is generated.

## 2.15 Directives

These are just a sample of the many reStructuredText Directives. For others, please see <http://docutils.sourceforge.net/docs/ref/rst/directives.html>.

### 2.15.1 Document Parts

An example of the "contents" directive can be seen above this section (a local, untitled table of *contents*) and at the beginning of the document (a document-wide *table of contents*).

## 2 Body Elements

---

### 2.15.2 Images

An image directive (also clickable -- a hyperlink reference):

*reStructuredText*

A figure directive:

### 2.15.3 Admonitions

**Attention!** Directives at large.

**Caution!** Don't take any wooden nickels.

**!DANGER!** Mad scientist at work!

**Error:** Does not compute.

---

**Hint:**

It's bigger than a bread box.

---

**Important**

- Wash behind your ears.
  - Clean up your room.
  - Call your mother.
  - Back up your data.
- 

**Note:**

This is a note.

---

**Tip:**

15% if the service is good.

---

**Warning:** Strong prose may provoke extreme mental exertion. Reader discretion is strongly advised.

---

**And, by the way...**

You can make up your own admonition too.

---

### 2.15.4 Topics, Sidebars, and Rubrics

#### Sidebar Title

This is a sidebar. It is for text outside the flow of the main text.

#### This is a rubric inside a sidebar

Sidebars often appears beside the main text with a border and background color.

#### Topic Title

This is a topic.

This is a rubric

### 2.15.5 Target Footnotes

### 2.15.6 Replacement Text

I recommend you try Python, *the* best language around <sup>1</sup>.

### 2.15.7 Compound Paragraph


This paragraph contains a literal block:

```
Connecting... OK
Transmitting data... OK
Disconnecting... OK
```

and thus consists of a simple paragraph, a literal block, and another simple paragraph. Nonetheless it is semantically *one* paragraph.

This construct is called a *compound paragraph* and can be produced with the "compound" directive.

## 2.16 Substitution Definitions

An inline image () example:

(Substitution definitions are not visible in the HTML source.)

## 2.17 Comments

Here's one:

(View the HTML source to see the comment.)



## 3 Error Handling

Any errors caught during processing will generate system messages.

\*\*\* Expect 6 errors (including this one). \*\*\*

There should be six messages in the following, auto-generated section, "Docutils System Messages":

