Inline Markup

Inline markup allows words and phrases within text to have character styles (like italics and boldface) and functionality (like hyperlinks).

`*emphasis*`	*emphasis*
`**strong emphasis**`	**strong emphasis**
`` `interpreted text` ``	The rendering and meaning of interpreted text is domain- or application-dependent.
`` ``inline literal`` ```	`inline literal`
`reference_`	reference
`` `phrase reference`_ ``	phrase reference
`anonymous__`	anonymous
`_`inline internal target``	inline internal target
`|substitution reference|`	The result is substituted in from the substitution definition.
`footnote reference [1]_`	footnote reference [1]
`citation reference [CIT2002]_`	citation reference [CIT2002]

`http://docutils.sf.net/`	http://docutils.sf.net/

Escaping with Backslashes

reStructuredText uses backslashes ("\") to override the special meaning given to markup characters and get the literal characters themselves. To get a literal backslash, use an escaped backslash ("\\"). For example:

`*escape* ``with`` "\"`	*escape* with ""
`*escape* \``with`` "\\"`	*escape* ``with`` "\"

Lists

```
- This is item 1. A blank line before the first
  and last items is required.
- This is item 2

- Item 3: blank lines between items are optional.
- Item 4: Bullets are "-", "*" or "+".
  Continuing text must be aligned after the bullet
  and whitespace.
```

- This is item 1. A blank line before the first and last items is required.
- This is item 2
- Item 3: blank lines between items are optional.
- Item 4: Bullets are "-", "*" or "+". Continuing text must be aligned after the bullet and whitespace.

```
3. This is the first item
4. This is the second item
5. Enumerators are arabic numbers,
   single letters, or roman numerals
6. List items should be sequentially
   numbered, but need not start at 1
   (although not all formatters will
   honour the first index).
#. This item is auto-enumerated
```

3. This is the first item
4. This is the second item
5. Enumerators are arabic numbers, single letters, or roman numerals
6. List items should be sequentially numbered, but need not start at 1 (although not all formatters will honour the first index).
7. This item is auto-enumerated

```
what
  Definition lists associate a term with
  a definition.

how
  The term is a one-line phrase, and the
  definition is one or more paragraphs or
  body elements, indented relative to the
  term. Blank lines are not allowed
  between term and definition.
```

what
Definition lists associate a term with a definition.

how
The term is a one-line phrase, and the definition is one or more paragraphs or body elements, indented relative to the term. Blank lines are not allowed between term and definition.

```
:Authors:
    Tony J. (Tibs) Ibbs,
    David Goodger

    (and sundry other good-natured folks)

:Version: 1.0 of 2001/08/08
:Dedication: To my father.
```

Authors:	Tony J. (Tibs) Ibbs, David Goodger (and sundry other good-natured folks)
Version:	1.0 of 2001/08/08
Dedication:	To my father.

```
-a            command-line option "a"
-b file       options can have arguments
              and long descriptions
--long        options can be long also
--input=file  long options can also have
              arguments
/V            DOS/VMS-style options too
```

`-a`	command-line option "a"
`-b file`	options can have arguments and long descriptions
`--long`	options can be long also
`--input=file`	long options can also have arguments
`/V`	DOS/VMS-style options too

Section Structure

```
Title
=====

Titles are underlined (or over- and underlined) with
a nonalphanumeric character at least as long as the
text.

A lone top-level section is lifted up to be the
document's title.

Any non-alphanumeric character can be used, but
Python convention is:

* ``#`` with overline, for parts
* ``*`` with overline, for chapters
* ``=``, for sections
* ``-``, for subsections
* ``^``, for subsubsections
* ``"``, for paragraphs
```

Title

Titles are underlined (or over- and underlined) with a nonalphanumeric character at least as long as the text.

A lone top-level section is lifted up to be the document's title.

Any non-alphanumeric character can be used, but Python convention is:

- `#` with overline, for parts
- `*` with overline, for chapters
- `=`, for sections
- `-`, for subsections
- `^`, for subsubsections
- `"`, for paragraphs

Blocks

```
This is a paragraph.

Paragraphs line up at their left edges, and are
normally separated by blank lines.
```

This is a paragraph.

Paragraphs line up at their left edges, and are normally separated by blank lines.

```
A paragraph containing only two colons indicates
the following indented or quoted text is a literal
block or quoted text is a literal block.

::

  Whitespace, newlines, blank lines, and  all kinds of
  markup (like *this* or \this) is preserved here.

You can also tack the ``::`` at the end of a
paragraph::

   It's very convenient to use this form.

Per-line quoting can also be used for unindented
blocks::
```

A paragraph containing only two colons indicates that the following indented or quoted text is a literal block.

```
Whitespace, newlines, blank lines, and
all kinds of markup (like *this* or
\this) is preserved by literal blocks.
```

You can also tack the `::` at the end of a paragraph:

```
It's very convenient to use this form.
```

Per-line quoting can also be used for unindented blocks:

```
> Useful for quotes from email and
> for Haskell literate programming.
```

> Useful for quotes from email and
> for Haskell literate programming.

```
| Line blocks are useful for addresses,
| verse, and adornment-free lists.
|
| 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 spaces in place of vertical bars.
```

Line blocks are useful for addresses,
verse, and adornment-free lists.

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 spaces in place of vertical bars.

```
Block quotes are just:

    Indented paragraphs,

        and they may nest.
```

Block quotes are just:

> Indented paragraphs,
>
> > and they may nest.

```
Doctest blocks are interactive
Python sessions. They begin with
"``>>>``" and end with a blank line.

>>> print "This is a doctest block."
This is a doctest block.
```

Doctest blocks are interactive Python sessions. They begin with " `>>>` " and end with a blank line.

```
>>> print "This is a doctest block."
This is a doctest block.
```

```
A transition marker is a horizontal line
of 4 or more repeated punctuation
characters.

------------

A transition should not begin or end a
section or document, nor should two
transitions be immediately adjacent.
```

A transition marker is a horizontal line of 4 or more repeated punctuation characters.

A transition should not begin or end a section or document, nor should two transitions be immediately adjacent.

Tables

There are two syntaxes for tables in reStructuredText. Grid tables are complete but cumbersome to create. Simple tables are easy to create but limited (no row spans, etc.).

```
+------------+------------+-----------+
| Header 1   | Header 2   | Header 3  |
+============+============+===========+
| body row 1 | column 2   | column 3  |
+------------+------------+-----------+
| body row 2 | Cells may span columns.|
+------------+------------+-----------+
| body row 3 | Cells may  | - Cells   |
+------------+ span rows. | - contain |
| body row 4 |            | - blocks. |
+------------+------------+-----------+
```

Header 1	Header 2	Header 3
body row 1	column 2	column 3
body row 2	Cells may span columns.	
body row 3	Cells may span rows.	• Cells • contain • blocks.
body row 4		

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

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

Explicit Markup

Explicit markup blocks are used for constructs which float (footnotes), have no direct paper-document representation (hyperlink targets, comments), or require specialized processing (directives). They all begin with two periods and whitespace, the "explicit markup start".

```
Footnote references, like [5]_.
Note that footnotes may get
rearranged, e.g., to the bottom of
the "page".

.. [5] A numerical footnote. Note
   there's no colon after the ``]``.
```

Footnote references, like [5]. Note that footnotes may get rearranged, e.g., to the bottom of the "page".

[5]	A numerical footnote. Note there's no colon after the `]`.

```
Autonumbered footnotes are
possible, like using [#]_ and [#]_.

.. [#] This is the first one.
.. [#] This is the second one.

They may be assigned 'autonumber
labels' - for instance,
[#fourth]_ and [#third]_.

.. [#third] a.k.a. third_

.. [#fourth] a.k.a. fourth_
```

Autonumbered footnotes are possible, like using [1] and [2].

[1]	This is the first one.

[2]	This is the second one.

They may be assigned 'autonumber labels' - for instance, [4] and [3].

[3]	a.k.a. third

[4]	a.k.a. fourth

```
Auto-symbol footnotes are also
possible, like this: [*]_ and [*]_.

.. [*] This is the first one.
.. [*] This is the second one.
```

Auto-symbol footnotes are also possible, like this: [*] and [†].

[*]	This is the first one.

[†]	This is the second one.

```
Citation references, like [CIT2002]_.
Note that citations may get
rearranged, e.g., to the bottom of
the "page".

.. [CIT2002] A citation
   (as often used in journals).

Citation labels contain alphanumerics,
```

Citation references, like [CIT2002]. Note that citations may get rearranged, e.g., to the bottom of the "page".

[CIT2002]	*(1, 2)* A citation (as often used in journals).

```
underlines, hyphens and fullstops.
Case is not significant.

Given a citation like [this]_, one
can also refer to it like this_.

.. [this] here.
```

Citation labels contain alphanumerics, underlines, hyphens and fullstops. Case is not significant.

Given a citation like [this], one can also refer to it like this.

[this]	here.

```
External hyperlinks, like Python_.

.. _Python: http://www.python.org/
```

External hyperlinks, like Python.

```
External hyperlinks, like `Python
<http://www.python.org/>`_.
```

External hyperlinks, like Python.

```
Internal crossreferences, like example_.

.. _example:

This is an example crossreference target.
```

Internal crossreferences, like example.

This is an example crossreference target.

```
Python_ is `my favourite
programming language`__.

.. _Python: http://www.python.org/

__ Python_
```

Python is my favourite programming language.

```
Titles are targets, too
=======================

Implict references, like `Titles are targets, too`_.
```

Titles are targets, too

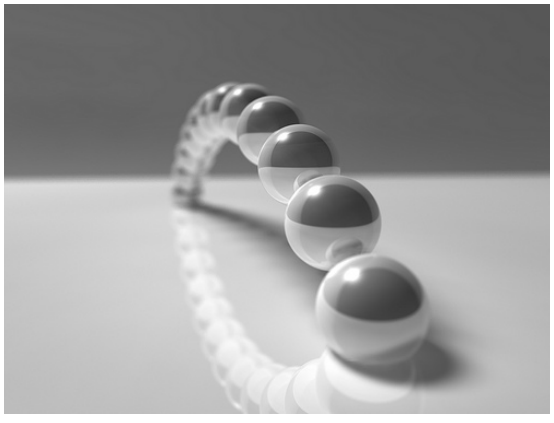
Implict references, like Titles are targets, too.

Directives are a general-purpose extension mechanism, a way of adding support for new constructs without adding new syntax. For a description of all standard directives, see reStructuredText Directives (http://is.gd/2Ecqh).

```
For instance:

.. image:: magnetic-balls.jpg
   :width: 40pt
```

For instance:



Substitutions are like inline directives, allowing graphics and arbitrary constructs within text.

The ☣ symbol must be used on containers used to

```
The |biohazard| symbol must be used on containers used to
dispose of medical waste.

.. |biohazard| image:: biohazard.png
   :align: middle
   :width: 12
```

dispose of medical waste.

Any text which begins with an explicit markup start but doesn't use the syntax of any of the constructs above, is a comment.

```
.. This text will not be shown
   (but, for instance, in HTML might be
   rendered as an HTML comment)
```

```
An "empty comment" does not
consume following blocks.
(An empty comment is ".." with
blank lines before and after.)

..

        So this block is not "lost",
        despite its indentation.
```

An "empty comment" does not consume following blocks. (An empty comment is ".." with blank lines before and after.)

> So this block is not "lost", despite its indentation.

Credits

CP Font from LiquiType:	http://www.liquitype.com/workshop/type_design/cp-mono
Magnetic Balls V2 image by fdecomite:	http://www.flickr.com/photos/fdecomite/2926556794/
Sponsored by Net Managers	http://www.netmanagers.com.ar
Typeset using rst2pdf	http://rst2pdf.googlecode.com

Based on quickref.txt from docutils

Non-Commercial

Share Alike