

reStructuredText 简介

本章节介绍 reStructuredText (reST) 的概念和语法，为文档生成者提供足够的信息。reST 被认为是简单，实用的标记语言，因此学习它不会花太多时间。

See also

读物 [reStructuredText User Documentation](#). 文档内“ref”链接指向reST的分类参考文献.

段落

段落 (`:duref:ref <paragraphs>`) 是reST 文件的基本模块. 段落是由空行分隔的一段文本. 和Python一样, 对齐也是reST的操作符, 因此同一段落的行都是左对齐的.

内联标记

标准的reST 内联标记相当简单:

星号: `*text*` 是强调 (斜体),
双星号: `**text**` 重点强调 (加粗),
反引号: ``text`` 代码样式.

星号及反引号在文本中容易与内联标记符号混淆, 可使用反斜杠符号转义.

标记需注意的一些限制:

不能相互嵌套,
内容前后不能由空白: 这样写 ``` text``` 是错误的,
如果内容需要特殊字符分隔. 使用反斜杠转义, 如: `thisis\ *one*\ word`.

这些限制在未来版本可能会被改善.

reST 也允许自定义“文本解释角色”, 这意味着可以以特定的方式解释文本. Sphinx以此方式提供语义标记及参考索引, 操作符为 `:rolename:content``.

标准reST 提供以下规则:

`:durole:emphasis`` – 写成 `*emphasis*`
`:durole:strong`` – 写成 `**strong**`
`:durole:literal`` – 写成 ```literal```
`:durole:subscript`` – 下标
`:durole:superscript`` – 上标
`:durole:title-reference`` – 书、期刊等材料的标题

详情请查看 [内联标记](#).

列表与引用

列表标记 (`:duref:ref <bullet-lists>`) 的使用最自然: 仅在段落的开头放置一个星号和一个缩进. 编号的列表也可以;也可以使用符号 # 自动加序号:

```
* 这是一个项目符号列表.
* 它有两项,
  第二项使用两行.

1. 这是个有序列表.
2. 也有两项.

#. 是个有序列表.
#. 也有两项.
```

列表可以嵌套, 但是需跟父列表使用空行分隔

```
* 这是
* 一个列表
```

- * 嵌套列表
- * 子项
- * 父列表继续

定义列表 (:duref: ref <definition-lists>`)

术语 (term 文本开头行)
定义术语，必须缩进

可以有多段组成

下一术语 (term)
描述。

一行仅能写一个术语.

引用段落 (:duref: ref <block-quotes>`) 仅使用缩进 (相对于周围段落) 创建.

行模块 (:duref: ref <line-blocks>`) 可以这样分隔

| 这些行
| 在源文件里
| 被分隔的一模一样。

还有其他有用的模块:

- 字段列表 (:duref: ref <field-lists>`)
- 选项列表(:duref: ref <option-lists>`)
- 字面引用模块 (:duref: ref <quoted-literal-blocks>`)
- 文档测试模块 (:duref: ref <doctest-blocks>`)

源代码

字面代码块 (:duref: ref <literal-blocks>`) 在段落的后面使用标记 :: 引出. 代码块必须缩进(同段落，需要与周围文本以空行分隔):

这是一段正常文本。下一段是代码文字::

它不需要特别处理，仅是
缩进就可以了。

它可以有多行。

再是正常的文本段。

这个 :: 标记很优雅:


- 如果作为独立段落存在,则整段都不会出现在文档里.
- 如果前面有空白，则标记被移除.
- 如果前面是非空白，则标记被一个冒号取代.

因此上面的例子第一段文字将变为"下一段是代码文字":.

表格

支持两种表格. 一种是 网格表格 (:duref: ref <grid-tables>`), 可以自定义表格的边框. 如下:

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	

简单表格 (:duref: `ref <simple-tables>`) 书写简单, 但有一些限制: 需要有多行, 且第一列元素不能分行显示, 如下:

====	====	=====
A	B	A and B
====	====	=====
False	False	False
True	False	False
False	True	False
True	True	True
====	====	=====

超链接

外部链接

使用 `链接文本` <http://example.com/>`_` 可以插入网页链接. 链接文本是网址, 则不需要特别标记, 分析器会自动发现文本里的链接或邮件地址. 可以把链接和标签分开 (:duref: `ref <hyperlink-targets>`), 如下:

```
段落里包含 `a link`_.

.. _a link: http://example.com/
```

内部链接

内部链接是Sphinx特定的reST角色, 查看章节 [交叉索引的位置](#).

章节

章节的标题 (:duref: `ref <sections>`) 在双上划线符号之间 (或为下划线), 并且符号的长度不能小于文本的长度:

```
=====
This is a heading
=====
```

通常没有专门的符号表示标题的等级, 但是对于Python 文档, 可以这样认为:

- # 及上划线表示部分
- * 及上划线表示章节
- =, 小章节
- , 子章节
- ^, 子章节的子章节
- ", 段落

当然也可以标记 (查看 reST 文档), 定义章节的层次, 但是需要注意输出格式(HTML, LaTeX)所支持的层次深度 .

显式标记

显式标记"Explicit markup" (:duref: `ref <explicit-markup-blocks>`) 用在那些需做特殊处理的reST结构中, 如尾注, 突出段落, 评论, 通用指令. 显式标记以 .. 开始, 后跟空白符, 与下面段落的缩进一样. (在显式标记与正常的段落间需有空行, 这听起来有些复杂, 但是写起来会非常直观.)

指令

指令 (:duref: `ref <directives>`) 是显式标记最常用的模块. 也是reST 的扩展规则, 在 Sphinx 经常被用到.

文档工具支持以下指令:

警告: `:dudir:`attention``, `:dudir:`caution``, `:dudir:`danger``, `:dudir:`error``, `:dudir:`hint``, `:dudir:`important``, `:dudir:`note``, `:dudir:`tip``, `:dudir:`warning`` 及通用标记 `:dudir:`admonition``. (大多数模式仅支持 “note” 及 “warning”)

图像:

`:dudir:`image`` (详情可看下面的 [图像](#))
`:dudir:`figure`` (有标题及可选说明的图像)

额外的主体元素:

`:dudir:`contents <table-of-contents>`` (本地, 仅是当前文件的内容表格)
`:dudir:`container`` (自定义容器, 用来生成HTML的 `<div>`)
`:dudir:`rubric`` (和文档章节无关的标题)
`:dudir:`topic``, `:dudir:`sidebar`` (高亮显示的主体元素)
`:dudir:`parsed-literal`` (支持内联标记的斜体模块)
`:dudir:`epigraph`` (可选属性行的摘要模块)
`:dudir:`highlights``, `:dudir:`pull-quote`` (有自己的类属性的摘要模块)
`:dudir:`compound`` (复合段落)

专用表格:

`:dudir:`table`` (有标题的表格)
`:dudir:`csv-table`` (CSV自动生成表格)
`:dudir:`list-table`` (列表生成的表格)

专用指令:

`:dudir:`raw`` (包含原始格式的标记)
`:dudir:`include`` (包含reStructuredText标记的文件) – 在Sphinx中,如果包含绝对文件路径, 指令会以源目录地址做为参照
`:dudir:`class`` (将类属性指派给下一个元素) [\[1\]](#)

HTML 特性:

`:dudir:`meta`` (生成HTML `<meta>` 标签)
`:dudir:`title`` (覆盖文档标题)

影响标记:

`:dudir:`default-role`` (设置新的默认角色)
`:dudir:`role`` (创建新的角色)

如果仅有一个文件, 最好使用 `:confval:`default_role``.

设置不使用指令 `:dudir:`sectnum``, `:dudir:`header`` 及 `:dudir:`footer``.

Sphinx 新增指令可查阅 [Sphinx标记的组成](#).

指令有名字, 参数, 选项及内容组成. (记住这些, 在下面一小节中自定义指令里会用到). 来看一个例子:

```
.. function:: foo(x)
    foo(y, z)
:module: some.module.name

返回用户输入的一行文本.
```

function 是指令名字. 在第一行和第二行给出了两个参数, 及一个选项 module (如你所见, 选项在参数后给出, 由冒号引出). 选项必须与指令有一样的缩进.

指令的内容在隔开一个空行后, 与指令有一样缩进.



图像

reST 支持图像指令 (`:dudir:`ref <image>``), 如下:

```
.. image:: gnu.png
(选项)
```

这里给出的文件名 (gnu.png) 必须是源文件的相对路径, 如果是绝对路径则以源目录为根目录. 例如, 在文件 sketch/spam.rst 引用图像 images/spam.png, 则使用 `../images/spam.png` 或者 `/images/spam.png`.

Sphinx 会自动将图像文件拷贝到输出目录的子目录里, (输出HTML时目录为 `_static`)

图像的大小选项 (width 及 height): 如果没有单位或单位为像素, 给定的尺寸信息仅在输出通道支持像素时才有用 (如输出LaTeX 没  v: latest  输出(如 pt)HTML、LaTeX 时被用到.

Sphinx 延伸了标准的文档化行为，只需在后面加星号：

```
.. image:: gnu.*
```

上面这样写，Sphinx 会搜索所有名字匹配的图像，而不管图像类型。每个生成器则会选择最合适的图像。一般，在源文件目录里文件名 `gnu.*` 会含有两个文件 `gnu.pdf` 和 `gnu.png`，LaTeX 生成器会选择前者，而HTML 生成器则匹配后者。

Changed in version 0.4: 添加对文件名以星号结束的支持。

Changed in version 0.6: 图像路径可以是绝对路径。

尾注

尾注 (`:duref:ref <footnotes>`)，使用 `[#name]_` 标记尾注的位置，尾注的内容则在文档底部红色标题“Footnotes”的后面，如下：

```
Lorem ipsum [#f1]_ dolor sit amet ... [#f2]_

.. rubric:: Footnotes

.. [#f1] 第一条尾注的文本。
.. [#f2] 第二条尾注的文本。
```

你也可以使用数字尾注 (`[1]_`) 或使用自动排序的 (`[#]_`)。

引用

支持标准的reST 引用 (`:duref:ref <citations>`)，且新增了“global”特性，所有参考文献不受所在文件的限制。如：

```
Lorem ipsum [Ref]_ dolor sit amet.

.. [Ref] 参考文献，书，URL 等。
```

引用的使用同尾注很相近，但是它们没有数字标签或以 `#` 开始。

替换

reST 支持替换 “substitutions” (`:duref:ref <substitution-definitions>`)，有一小段文本或标记被关联到 `|name|`。定义与尾注一样需有明确的标记块，如下：

```
.. |name| replace:: replacement *text*
```

或者：

```
.. |caution| image:: warning.png
   :alt: Warning!
```

详情查看 `:duref:reST reference for substitutions <substitution-definitions>`。

如果想在所有文档中使用这些替换，需把它们放在 `:confval:rst_prolog` 或一个单独文件里，然后在使用它们的文档文件里包含这个文件，包含指令 `include`。（请给出包含文件的扩展名，已区别于其他的源文件，避免Sphinx将其作为独立的文档文件。）

Sphinx 定义了一些默认的替换，请查看 [替换](#)。

评论

有明确标记块但又不是有效的结构标记的标记（像上面的尾注）都被视为评论 (`:duref:ref <comments>`)。例如：

```
.. 这是一个评论。
```

可以通过缩进产生多行评论：

```
..
    这整个缩进块都是
    一个评论。

    仍是一个评论。
```

源编码

在reST使用Unicode字符可以容易的包含特殊字符如破折号，版权标志. Sphinx 默认源文件使用UTF-8 编码; 你可以通过 `:confval:source_encoding`` 的配置值改变编码.

常见问题

具体使用中可能会遇到一些问题:

内联标记的分离 如上面所讲，内联标记需与周围的文本使用空格分隔, 内联标记内部则使用反斜线转义空格. 查看详情: [the reference](#) .

内联标记不能嵌套 像这样写 `*see :func:`foo`*` 是不允许的.

Footnotes

[1] 当默认主域里包含指令 `class` , 这个指令将被隐藏 因此, Sphinx使用 `rst-class`.