sphinx Documentation

firewang

2020 年 03 月 20 日

Contents:

用于学习 sphinx 和 reStructureText

CHAPTER 1

sphinx 介绍

1.1 基本介绍

Sphinx 是一种文档工具，它可以令人轻松的撰写出清晰且优美的文档, 由 Georg Brandl 在 BSD 许可证下开发。

它最初是为 Python 文档创建的，它具有出色的工具，可用于各种语言的软件项目文档。

- 输出格式: HTML (包括 Windows HTML 帮助), LaTeX (适用于可打印的 PDF 版本), ePub, Texinfo, 手册页，纯文本
- 广泛的交叉引用: 语义标记和功能，类，引用，词汇表术语和类似信息的自动链接
- 分层结构: 轻松定义文档树，自动链接到平级，上级和下级
- 自动索引: 一般索引以及特定于语言的模块索引
- 代码处理: 使用 Pygments 荧光笔自动突出显示
- 扩展: 自动测试代码片段，包含 Python 模块（API 文档）中的文档字符串等
- 贡献的扩展: 用户在第二个存储库中贡献了 50 多个扩展; 其中大多数可以从 PyPI 安装

Sphinx 使用 reStructuredText 作为其标记语言，其许多优点来自 reStructuredText 及其解析和翻译套件 Docutils 的强大功能和直接性。

1.2 安装

Sphinx 是用 Python 编写的，支持 Python 3.5+。

1.2.1 Linux

Debian/Ubuntu

使用 apt-get 安装 python3-sphinx:

```
$ apt-get install python3-sphinx
```

RHEL, CentOS

使用 yum 安装 python-sphinx :

```
$ yum install python-sphinx
```

1.2.2 PyPI

```
pip install -U sphinx
```

1.2.3 源码安装

```
git clone https://github.com/sphinx-doc/sphinx
$ cd sphinx
$ pip install .
```

1.3 与其他标记语言互相转换

- Gerard Flanagan 编写了一个脚本，将纯 HTML 转换为 reST; 它可以在 Python 包索引 找到。
- 为了将旧的 Python 文档转换为 Sphinx，编写了一个转换器，可以在 Python SVN 存储库 中找到。它包含将 Python-doc 样式的 LaTeX 标记转换为 Sphinx reST 的通用代码。
- Marcin Wojdyr 编写了一个脚本，将 Docbook 转换为使用 Sphinx 标记的 reST; 它位于 GitHub 。
- Christophe de Vienne 编写了一个工具，用于将 Open/LibreOffice 文档转换为 Sphinx: odt2sphinx 。
- 要转换不同的标记语言文本，Pandoc 是一个非常有用的工具。

1.4 快速开始

为了简化入门过程，Sphinx 提供了一个工具 sphinx-quickstart，它将生成一个文档源目录并用一些默认值填充它。

```
# 如果网速堪忧，增加 -i https://pypi.douban.com/simple
pip install sphinx sphinx-autobuild sphinx_rtd_theme
```

```
# 从命令行进入你的初始化目录，如果要托管到 github，则为本地仓目录
sphinx-quickstart
```

```
# 初始化过程中的配置项
Separate source and  directories (y/n) [n]:y  # 是否分离 build 和 source 目录
Project name: sphinx_handbook  # 项目名
Author name(s): firewang  # 作者
Project version []:  # 默认没有版本，按 Enter 跳过
Project release []:  # 默认没有发布版本，按 Enter 跳过
Project language [en]: zh_CN  # 默认英文，指定为中文
```

```
# Sphinx 中的插件配置
> autodoc: automatically insert docstrings from modules (y/n) [n]:
> doctest: automatically test code snippets in doctest blocks (y/n) [n]:
> intersphinx: link between Sphinx documentation of different projects (y/n) [n]:
> todo: write "todo" entries that can be shown or hidden on build (y/n) [n]:
> coverage: checks for documentation coverage (y/n) [n]:
> imgmath: include math, rendered as PNG or SVG images (y/n) [n]:
> mathjax: include math, rendered in the browser by MathJax (y/n) [n]:
> ifconfig: conditional inclusion of content based on config values (y/n) [n]:
> viewcode: include links to the source code of documented Python objects (y/n) [n]:
> githubpages: create .nojekyll file to publish the document on GitHub pages (y/n) [n]:
#autodoc: 从模块中自动插入代码
#doctest: 在文档中自动测试代码段
#intersphinx: 在不同的项目文档中的链接
#todo: 写入“todo”条目，可以在构建文档中显示或隐藏
#coverage: 检查文档覆盖率
#imgmath: 提供 PNG 或 SVG 图像 (包含数学)
#mathjax: 提供浏览器插件 MathJax(包含数学)
#ifconfig: 根据配置值的内容纳入条件
#viewcode: 包含指向 Python 对象源代码的链接
#githubpages: 创建.nojekyll 文件以在 GitHub 页面上发布文档
```

```
# 初始化完成以后，在目录下就会生成以下内容
   build
   make.bat
   Makefile
   source
       conf.py
       index.rst
       _static
       _templates

# build 为编译后生成的文档
# source 为文档目录，其中 index.rst 为索引目录，conf.py 是配置文件
```

之后新增、修改文件后更新编译文档，有两种方式

- **使用 sphinx-build 程序启动构建**

```
$ sphinx-build -b html sourcedir builddir
```

其中 sourcedir 是 source directory ， builddir 是您要在其中放置构建文档的目录。-b 选项选择一个构建器。

- **通过 make**

sphinx-quickstart 脚本创建了一个 Makefile 和一个 make.bat，它让你的生活更加轻松。

```
make html
# make pdf
# make epub
```

CHAPTER 2

sphinx 配置

配置文件 /source/conf.py 在构建时作为 Python 代码执行 (使用 execfile() , 并且当前目录设置为其包含目录)。

2.1 项目信息

project	记录的项目名称。
author	该文件的作者姓名。默认值为 unknown
copyright	2020, Author Name 风格的版权声明
version	主要项目版本, 用作 */version/* 的替代品。例如, 对于 Python 文档, 这可能类似于 2.6 。
release	完整的项目版本, 用作 */release/* 的替代品, 例如在 HTML 模板中。例如, 对于 Python 文档, 这可能类似于 2.6.0rc1 。 如果你不需要在 version 和 release 之间提供分隔, 只需将它们设置为相同的值即可。

2.2 一般配置项

2.2.1 extensions

可以是 Sphinx(名为 sphinx.ext.*) 或自定义的扩展。

请注意, 如果扩展名位于另一个目录中, 则可以在 conf 文件中扩展 sys.path(**使用绝对路径**)。

如果扩展路径是相对于 configuration directory , 使用 os.path.abspath()

```
import sys, os
sys.path.append(os.path.abspath('sphinxext'))
extensions = ['extname']
```

这样, 你可以从子目录 sphinxext 加载一个名为 extname 的扩展名。

2.2.2 source_suffix

源文件的文件扩展名。Sphinx 将具有此后缀的文件视为源。

```
source_suffix = {
    '.rst': 'restructuredtext',
    '.txt': 'restructuredtext',
    '.md': 'markdown',
}
```

可以使用源解析器扩展添加新文件类型。

2.2.3 source_encoding

所有 reST 源文件的编码。推荐的编码和默认值是‘utf-8-sig’。

2.2.4 source_parsers

如果给出, 则不同源的解析器类字典就足够了。键是后缀, 值可以是类或字符串, 给出解析器类的完全限定名称。解析器类可以是 docutils.parsers.Parser 或 sphinx.parsers.Parser 。不在字典中的后缀的文件将使用默认的 reStructuredText 解析器进行解析。

```
source_parsers = {'.md': 'recommonmark.parser.CommonMarkParser'}
```

2.2.5 master_doc

主目录文件名，默认为 index

2.2.6 exclude_patterns

查找源文件时应排除的 glob 样式模式列表。它们与源目录相对于源目录进行匹配, 在所有平台上使用斜杠作为目录分隔符。

示例模式:

```
'library/xml.rst' - 忽略 library/xml.rst 文件 (替换条目 unused_docs)
'library/xml' - 忽略 library/xml 目录
'library/xml*' - 忽略以 library/xml 开头的所有文件和目录
'**/.svn' - 忽略所有 .svn 目录
```

2.2.7 templates_path

包含额外模板的路径列表 (覆盖内置/主题特定模板的模板)。相对路径被视为相对于配置目录。

由于这些文件不是要构建的, 因此它们会自动添加到 exclude_patterns 中.

2.2.8 rst_epilog

一串 reStructuredText, 它将包含在每个读取的源文件的末尾。

```
rst_epilog = """
.. |psf| replace:: Python Software Foundation
"""
```

2.2.9 rst_prolog

一串 reStructuredText, 它将包含在每个读取的源文件的开头。

```
rst_prolog = """
.. |psf| replace:: Python Software Foundation
"""
```

2.2.10 primary_domain

2.2.11 default_role

2.2.12 keep_warnings

2.2.13 suppress_warnings

用于禁止任意警告消息的警告类型列表。

Sphinx 支持以下警告类型:

- app.add_node
- app.add_directive
- app.add_role
- app.add_generic_role
- app.add_source_parser
- download.not_readable
- image.not_readable
- ref.term
- ref.ref
- ref.numref
- ref.keyword
- ref.option
- ref.citation
- ref.footnote
- ref.doc
- ref.python
- misc.highlighting_failure
- toc.secnum
- epub.unknown_project_files

2.2.14 needs_sphinx

指定用来构建的 sphinx 版本，如果设置为 major.minor 版本字符串, 如‘1.1’, Sphinx 会将其与版本进行比较, 如果它太旧则拒绝构建。默认是没有要求的。

2.2.15 needs_extensions

指定扩展的版本, 例如: needs_extensions = { 'sphinxcontrib.something' : '1.5' } 。版本字符串应采用 major.minor 形式。不必为所有扩展指定要求, 仅适用于您要检查的扩展。

2.2.16 manpages_url

交叉引用的 URL manpage 指令。如果将其定义为 https://manpages.debian.org/{path} , 则 *man(1)* 角色将链接到 <https://manpages.debian.org/man(1)> 。

可用的模式是:

- page - 手册页 (man)
- section - 手册部分 (1)
- path - 原始的手册页和指定的部分 (man(1))

2.2.17 numfig

如果为 true, 则数字, 表格和代码块如果有标题则会自动编号。numref 角色已启用。

2.2.18 numfig_format

一个字典将 'figure' , 'table' , 'code-block' 和 'section' 映射到用于图号格式的字符串。作为一个特殊字符, %s 将被替换为图号。默认是使用 'Fig. %s' 为 'figure' , 'Table %s' 为 'table' , 'Listing %s' 为 'code-block' 和 'Section' 为 'section'

2.2.19 numfig_secnum_depth

- 如果设置为 0 , 则数字, 表格和代码块从 1 开始连续编号。
- 如果 1 (默认) 数字将是 x.1 , x.2 , ⋯与 x 的节号 (顶级切片; 没有 x 如果没有部分)。只有当通过 toctree 指令的:numbered: 选项激活了段号时, 这才自然适用。
- 2 表示数字将是 xy1 , xy2 , ⋯如果位于子区域 (但仍然是 x.1 , x.2 , ⋯如果直接位于一个部分和 1 , 2 , ⋯如果不在任何顶级部分。)

2.2.20 smartquotes

2.2.21 smartquotes_action

2.2.22 smartquotes_excludes

2.2.23 tls_verify

如果为 true, Sphinx 将验证服务器认证。默认为 True 。

2.2.24 tls_cacerts

CA 的证书文件的路径或包含证书的目录的路径。这也允许字典映射主机名到证书文件的路径。证书用于验证服务器认证。

2.2.25 highlight_language

用于突出显示源代码的默认语言。默认语言为 'python3'。代码高亮

2.2.26 highlight_options

Pygments documentation

2.2.27 pygments_style

用于 Pygments 突出显示源代码的样式名称。如果未设置, 则为 HTML 输出选择主题的默认样式或 'sphinx'。

2.2.28 add_function_parentheses

一个布尔值, 决定是否将括号附加到函数和方法角色文本 (例如 `input()` 的内容) 以表示该名称是可调用的。默认为 True 。

2.2.29 add_module_names

A boolean that decides whether module names are prepended to all object names (for object types where a "module" of some kind is defined), e.g. for py:function directives. Default is True.

2.2.30 show_authors

一个布尔值, 决定 codeauthor 和 sectionauthor 指令在构建的文件中产生任何输出。

2.2.31 modindex_common_prefix

为了对 Python 模块索引进行排序而忽略的前缀列表 (例如, 如果将其设置为 ['foo.'] , 那么 foo.bar 将显示在 B 下, 而不是 F)。如果您记录包含单个包的项目, 这可能很方便。仅适用于当前的 HTML 构建器。默认是 [] 。

2.2.32 trim_footnote_reference_space

在脚注引用之前修剪空格, 这是 reST 解析器识别脚注所必需的, 但在输出中看起来不太好。

2.2.33 trim_doctest_flags

如果为 true, 则删除 doctest 标志 (在行的末尾看起来像 doctest:FLAG, …的注释) 和 <BLANKLINE> 标记, 以显示交互式 Python 会话的所有代码块 (即 doctests) 。默认为 True 。有关包含 doctests 的更多可能性, 请参阅扩展名 doctest 。

2.3 国际化选项

影响 Sphinx 的母语支持

language 文档编写语言的代码。Sphinx 自动生成的任何文本都将使用该语言。

目前 Sphinx 支持的语言是:

- bn – 孟加拉语
- ca – 加泰罗尼亚语
- cs – 捷克语
- da – 丹麦语
- de – 德语
- en – 英语
- es – 西班牙语
- et – 爱沙尼亚语
- eu – 巴斯克语
- fa – 伊朗语
- fi – 芬兰语
- fr – 法语
- he – 希伯来语

- hr – 克罗地亚语
- hu – 匈牙利语
- id – 印度尼西亚语
- it – 意大利语
- ja – 日语
- ko – 朝鲜语
- lt – 立陶宛语
- lv – 拉脱维亚语
- mk – 马其顿语
- nb_NO – 挪威博克马尔语
- ne – 尼泊尔语
- nl – 荷兰语
- pl – 波兰语
- pt_BR – 巴西葡萄牙语
- pt_PT – 欧洲葡萄牙语
- ru – 俄语
- si – 僧伽罗语
- sk – 斯洛伐克语
- sl – 斯洛文尼亚语
- sv – 瑞典语
- tr – 土耳其语
- uk_UA – 乌克兰语
- vi – 越南语
- zh_CN – 简体中文
- zh_TW – 繁体中文

2.3.1 locale_dirs

相对于源目录, 在其中搜索其他消息目录 (请参阅 language) 的目录。此路径上的目录由标准 gettext 模块搜索。

内部消息是从 sphinx 的文本域中获取的; 因此, 如果将目录。/locale 添加到此设置, 则消息目录 (使用 msgfmt 编译为.po 格式) 必须位于./locale/language/LC_MESSAGES/sphinx.mo 。单个文档的文本域取决于 gettext_compact 。

默认是 [‘locales’].

2.3.2 gettext_compact

如果为 true, 则文档的文本域是其 docname, 如果它是顶级项目文件, 则为其基本目录。

默认情况下, 文档 markup/code.rst 最终出现在 markup 文本域中。将此选项设置为 False , 它是标记/代码。

2.3.3 gettext_uuid

如果为 true, 则 Sphinx 会在消息目录中生成用于版本跟踪的 uuid 信息。它用于:

- 在.pot 文件中为每个 msgids 添加 uid 行。
- 计算新 msgids 和以前保存的旧 msgids 之间的相似性。(计算时间长)

如果想加速计算, 可以使用 pip install python-levenshtein 来使用 C 编写的 python-levenshtein 第三方包。

默认是 False.

2.3.4 gettext_location

如果为 true, 则 Sphinx 为消息目录中的消息生成位置信息。默认 True.

2.3.5 gettext_auto_build

如果为 true, 则 Sphinx 为每个翻译目录文件构建 mo 文件。

默认是 True.

2.3.6 gettext_additional_targets

指定名称以启用 gettext 提取和转换。可以指定以下名称:

索引:	索引条款
Literal-block:	文字块和 code-block
Doctest-block:	doctest 块
Raw:	原始内容
图片:	image/figure uri 和 alt

例如: gettext_additional_targets = [‘literal-block’ , ‘image’] 默认是 []

2.3.7 figure_language_filename

语言特定数字的文件名格式。默认值为 *{root}.{language}{ext}* 它将从 *.. image:: dirname/filename.png* 扩展为 *dirname/filename.en.png* 可用的格式标记是:

- {root} - 文件名, 包括任何路径组件, 没有文件扩展名, 例如 dirname/filename
- {path} - 文件名的目录路径组件, 如果非空, 则带有斜杠, 例如 dirname/
- {basename} - 没有目录路径或文件扩展名组件的文件名, 例如 filename
- {ext} - 文件扩展名, 例如.png
- {language} - 翻译语言, 例如 en

例如, 将其设置为 *{path}{language}/{basename}{ext}* 将扩展为 *dirname/en/filename.png*

2.4 数学选项

math_number_all	如果要对所有显示的数学项进行编号, 请将此选项设置为 True 。默认为 False 。
math_eqref_format	用于格式化方程式引用标签的字符串。{number} 占位符代表等式编号。例: ‘Eq.{number}’ 被渲染为, 例如, Eq.10.
math_numfig	如果为 True , 则在页面中下显示的数学公式编号。默认为 True 。

2.5 HTML 输出选项

这些选项会影响 HTML 以及 HTML 帮助输出, 以及使用 Sphinx 的 HTMLWriter 类的其他构建器。

2.5.1 html 主题 html_theme

页面主题模板，默认 alabaster 。内置主题信息

2.5.2 html_theme_options

所选主题外观的选项字典。

2.5.3 html_theme_path

包含自定义主题的路径列表, 可以是子目录, 也可以是 zip 文件。相对路径被视为相对于配置目录。

2.5.4 html_style

用于 HTML 页面的样式表。该名称的文件必须存在于 Sphinx 的 static/ 路径中，或者存在于 html_static_path 中给出的自定义路径之一。默认值是所选主题给出的样式表。如果您只想添加或覆盖与主题样式表相比的一些内容, 使用 CSS @import 导入主题的样式表。

2.5.5 html_title

使用 Sphinx 内置模板生成的 html 页面的 title。默认 *<project> v<revision> documentation*

html_short_title 在 HTML docs 和 HTML Help docs 使用的 html title。默认使用 html_title 的设置值

2.5.6 html_baseurl

指向 HTML 文档根目录的 URL。它用于表示文档的位置, 如 canonical_url 。

html_context 要传递到所有页面的模板引擎上下文的值字典。单个值也可以使用 sphinx-build 的-A 命令行选项来设置。

2.5.7 html_logo

文档的徽标, 位于侧边栏的顶部，宽度不应超过 200 像素。默认值:None。

如果给定, 则必须是图像文件的名称 (相对于 configuration directory 的路径)。

若图片文件不存在于 _static 目录，将被复制到输出 HTML 的 _static 目录中。

2.5.8 html_favicon

文档的 favicon，现代浏览器使用它作为标签, 窗口和书签的图标。它应该是一个 Windows 风格的图标文件 (.ico), 大小为 16x16 或 32x32 像素。默认值: None 。

如果给定, 则必须是图像文件的名称 (相对于 configuration directory 的路径)。

若图片文件不存在于 _static 目录，将被复制到输出 HTML 的 _static 目录中。

2.5.9 html_css_files

CSS 文件列表。该条目必须是 filename 字符串或包含 filename 字符串和 attributes 字典的元组。filename 必须相对于 html_static_path , 或者是一个完整的 URI, 如 http://example.org/style.css 。attributes 用于 <link> 标签的属性。默认为空列表。

```
html_css_files = ['custom.css'
          'https://example.com/css/custom.css',
          ('print.css', {'media': 'print'})]
```

2.5.10 html_js_files

JavaScript filename 列表。该条目必须是 filename 字符串或包含 filename 字符串和 attributes 字典的元组。filename 必须相对于 html_static_path , 或者是一个完整的 URI, 如 http://example.org/script.js 。attributes 用于 <script> 标签的属性。默认为空列表。

```
html_js_files = ['script.js',
             'https://example.com/scripts/custom.js',
             ('custom.js', {'async': 'async'})]
```

2.5.11 html_static_path

包含自定义静态文件 (例如样式表 css 或脚本文件 js) 的路径列表。

相对路径被视为相对于配置目录。它们被复制到主题的静态文件之后的输出的 _static 目录中, 因此名为 default.css 的文件将覆盖主题的 default.css 。

由于这些文件不是要构建的, 因此它们会自动从源文件中排除。

2.5.12 html_extra_path

包含与文档无直接关系的额外文件的路径列表, 例如 robots.txt 或.htaccess 。

相对路径被视为相对于配置目录。它们被复制到输出目录。它们将覆盖任何同名的现有文件。

由于这些文件不是要构建的, 因此它们会自动从源文件中排除。

2.5.13 html_last_updated_fmt

如果这不是 None, 则使用给定的 strftime() 格式在每个页面底部插入 ‘Last updated on:’ 时间戳。空字符串相当于 ‘%b%d, %Y’ (或依赖于语言环境的等价物)。

2.5.14 html_use_smartypants

如果为 true, 则引号和短划线将转换为印刷正确的实体。默认值: True 。

2.5.15 html_add_permalinks

Sphinx will add “permalinks” for each heading and description environment as paragraph signs that become visible when the mouse hovers over them.

This value determines the text for the permalink; it defaults to “¶” . Set it to None or the empty string to disable permalinks.

2.5.16 html_sidebars

自定义侧边栏模板必须是将文档名称映射到模板名称的字典。

键可以包含 glob 样式的模式, 所有匹配的文档都将获得指定的侧边栏。(当多个 glob 样式模式与任何文档匹配时会发出警告)

值是列表, 它指定要包括的侧边栏模板的完整列表。如果要包含所有或部分默认侧边栏, 则必须将它们放入此列表中。默认侧边栏 (适用于与任何模式不匹配的文档) 由主题本身定义。内置主题默认使用这些模板: [‘localtoc.html’, ‘relations.html’, ‘sourcelink.html’, ‘searchbox.html’]

可呈现的内置侧边栏模板是:

- localtoc.html - 当前文档的细粒度目录
- globaltoc.html – 折叠整个文档集的粗粒度目录
- relations.html – 两个指向上一个和下一个文档的链接
- sourcelink.html – 指向当前文档源的链接 (如果在 html_show_sourcelink 中启用)
- searchbox.html – the “quick search” box

```
 html_sidebars = {
'**': ['globaltoc.html', 'sourcelink.html', 'searchbox.html'],
'using/windows': ['windowssidebar.html', 'searchbox.html'],}
```

这将呈现自定义模板 windowssidebar.html 和给定文档侧边栏内的快速搜索框, 并呈现所有其他页面的默认侧边栏 (除了本地 TOC 被全局 TOC 替换)。

如果所选主题不具有侧边栏, 则此值仅无效, 例如内置 scrolls 和 haiku 。

2.5.17 html_additional_pages

为 HTML 页面指定其他模板，为文档名称映射到模板名称的字典。

```
html_additional_pages = {
  'download': 'customdownload.html',}
```

这将把模板 customdownload.html 渲染为页面 download.html 。

2.5.18 html_domain_indices

如果为 true, 则除了常规索引外, 还会生成特定于域的索引。对于例如 Python 域, 这是全局模块索引。默认为 True 。

此值可以是 bool 或应生成的索引名称列表。要查找特定索引的索引名称, 请查看 HTML 文件名。例如, Python 模块索引的名称为‘py-modindex’。

2.5.19 html_use_index

默认为 True, 为 HTML 文档添加索引。

2.5.20 html_split_index

默认 False，如果为 true, 则索引生成两次: 一次作为包含所有条目的单个页面, 一次作为每个起始字母的一个页面。

2.5.21 html_copy_source

默认为 true, reST 源包含在 HTML 构建中 _sources/name

2.5.22 html_show_sourcelink

默认为 true(并且 html_copy_source 也为 true), 则指向 reST 源的链接将添加到侧栏。

2.5.23 html_sourcelink_suffix

附加到源链接的后缀 (参见 html_show_sourcelink), 除非它们已经有这个后缀。默认是‘.txt’。

2.5.24 html_use_opensearch

If nonempty, an OpenSearch description file will be output, and all pages will contain a <link> tag referring to it. Since OpenSearch doesn’t support relative URLs for its search page location, the value of this option must be the base URL from which these documents are served (without trailing slash), e.g. “https://docs.python.org”. The default is ‘’.

2.5.25 html_file_suffix

生成的 HTML 文件后缀，默认为“.html”

2.5.26 html_link_suffix

生成 HTML 文件链接的后缀。默认值为 html_file_suffix 设置值; 它可以设置不同 (例如, 支持不同的 Web 服务器设置)。

2.5.27 html_show_copyright

在 HTML Footer 显示 “(C) Copyright …” , 默认 True

2.5.28 html_show_sphinx

在 HTML Footer 显示 “Created using Sphinx” ,” Built with Sphinx”, 默认 True

2.5.29 html_output_encoding

HTML 输出文件的编码。默认为 ‘utf-8’

2.5.30 html_compact_lists

默认为 True, 如列表中包含单个段落和/或子列表的所有项目等等…(递归定义), 将不会对其任何项目使用 <p> 元素。这是标准的 docutils 行为。

2.5.31 html_secnumber_suffix

章节编号的后缀 (最后一个, 与章节标题相连的部分), 默认是” . “, 比如 “2.1.1. 章节标题”, 可设置为” “(空格)。

2.5.32 html_search_language

全文检索使用的语言, 默认为 en

支持这些语言:

- da – 丹麦语
- nl – 荷兰语
- en – 英语
- fi – 芬兰语
- fr – 法语
- de – 德语

- hu – 匈牙利语
- it – 意大利语
- ja – 日语
- no – 挪威语
- pt – 葡萄牙语
- ro – 罗马尼亚语
- ru – 俄语
- es – 西班牙语
- sv – 瑞典语
- tr – 土耳其语
- zh – 中文

每种语言 (日语除外) 都提供自己的词干算法。Sphinx 默认使用 Python 实现。您可以使用 C 实现来加速构建索引文件。PorterStemmer (en)，PyStemmer (所有语言)

2.5.33 html_search_options

带有搜索语言支持选项的字典, 默认为空。这些选项的含义取决于所选语言。

英语支持没有选择。

日本的支持有这些选择:

Type: type 是点模块路径字符串, 用于指定应该从哪实现 sphinx.search.ja.BaseSplitter。如果未指定或指定 None, 将使用‘sphinx.search.ja.DefaultSplitter’。

可以从以下模块中进行选择:

- ‘sphinx.search.ja.DefaultSplitter’：TinySegmenter algorithm. 这是默认分割器。
- ‘sphinx.search.ja.MecabSplitter’：MeCab 绑定。要使用这个拆分器, 需要‘mecab’python 绑定或动态链接库 (‘libmecab.so’用于 linux, ‘libmecab.dll’用于 windows)。
- ‘sphinx.search.ja.JanomeSplitter’：Janome 绑定。要使用这个分离器, 需要 Janome 。

1.6 版后已移除: ‘mecab’, ‘janome’ and ‘default’ 已弃用. 为了保持兼容性, ‘mecab’, ‘janome’ and ‘default’ 也可以接受。

其他选项值取决于您选择的拆分器值。

‘mecab’ 的选项: + dic_enc: MeCab 算法的编码。+ dict: 用于 MeCab 算法的字典。+ lib: 用于在未安装 Python 绑定的情况下通过 ctypes 查找 MeCab 库的库名。

```
html_search_options = {
'type': 'mecab',
'dic_enc': 'utf-8',
'dict': '/path/to/mecab.dic',
'lib': '/path/to/libmecab.so',}
```

'janome' 的选项:

- user_dic：Janome 的用户词典文件路径。
- user_dic_enc：user_dic 选项指定的用户词典文件的编码。默认为 *utf8*

中文的支持有这些选择: dict – 如果想使用自定义词典, jieba 字典路径。

2.5.34 html_search_scorer

实现搜索结果记分器的 JavaScript 文件的名称 (相对于配置目录)。如果为空, 则使用默认值。

2.5.35 html_scaled_image_link

默认为 True，图像本身会链接到原始图像 (如果它没有 target 选项或缩放相关选项: *scale* , *width* , *height*

2.5.36 html_math_renderer

HTML 输出的 math_renderer 扩展名。默认为 `mathjax` 。

2.5.37 singlehtml_sidebars

单个 HTML 页面输出选项，自定义侧边栏模板必须是将文档名称映射到模板名称的字典。它只允许一个名为 "index" 的键。所有其他键都被忽略。默认情况下，与 html_sidebars 相同。

2.5.38 htmlhelp_basename

HTML 帮助构建器的输出文件基名。默认是 `pydoc`

2.5.39 htmlhelp_file_suffix

HTML 帮助文档文件名后缀，默认 `.html`

2.5.40 htmlhelp_link_suffix

HTML 帮助文档链接后缀，默认 *.html*

2.6 EPUB 输出配置项

EPUB 输出配置项

2.7 LaTeX 输出配置项

LaTeX 输出配置项

2.8 文本输出选项

2.8.1 text_newlines

确定在文本输出中使用哪个行尾字符。

- 'unix' : 使用 Unix 风格的行结尾 (\n)
- 'windows' : 使用 Windows 风格的行结尾 (\r\n)
- 'native' : 使用构建文档的平台的行结束样式

默认值: 'unix'。

2.8.2 text_sectionchars

一个 7 个字符的字符串, 应该用于下划线部分。第一个字符用于第一级标题, 第二个字符用于第二级标题, 依此类推。

默认为 *=-~"+`

2.8.3 text_add_secnumbers

一个布尔值, 用于决定文本输出中是否包含节号。默认为 True 。

2.8.4 text_secnumber_suffix

章节编号的后缀 (最后一个, 与章节标题相连的部分), 默认是 ". ", 比如 2.1.1. 章节标题, 可设置为" "(空格)。

2.9 HTML 主题

Sphinx 支持通过 themes 更改其 HTML 输出的外观。

主题是 HTML 模板, 样式表和其他静态文件的集合。此外, 它还有一个配置文件, 用于指定要继承的主题, 要使用的突出显示样式以及用于自定义主题外观的选项。

也可以自己 制作主题

2.9.1 使用（配置）主题

使用内置主题

设置 conf.py 中 html_theme 的值即可

```
html_theme = 'classic'
```

修改主题的一些配置选项，修改 html_theme_options 配置项，对于使用的主题适用于哪些修改，视具体主题而定。

```
html_theme_options = {
 "rightsidebar": "true",
 "relbarbgcolor": "black"}
```

使用自定义主题

自定义主题可以是静态文件形式或 Python 包。对于静态表单, 支持目录 (包含 theme.conf 和其他所需文件) 或具有相同内容的 zip 文件。有配置值 html_theme_path，路径为相对 conf.py 所在目录的 ** 相对路径 **
,

例如, 如果文件中有一个主题 blue.zip, 则可以将其放在包含 conf.py 的目录中并使用此配置

```
html_theme = "blue"
html_theme_path = ["."]
```

python 包主题

使用 pip 安装主题包之后，和上一小节一样的流程配置即可。

```
pip install sphinxjp.themes.dotted
```

2.9.2 内置主题

图 1: alabaster

图 2: agogo

图 3: bizstyle

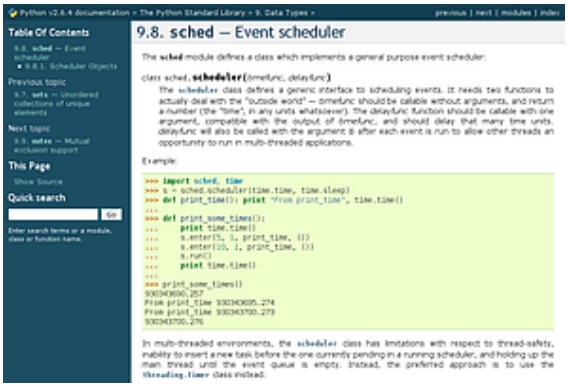

图 4: classic

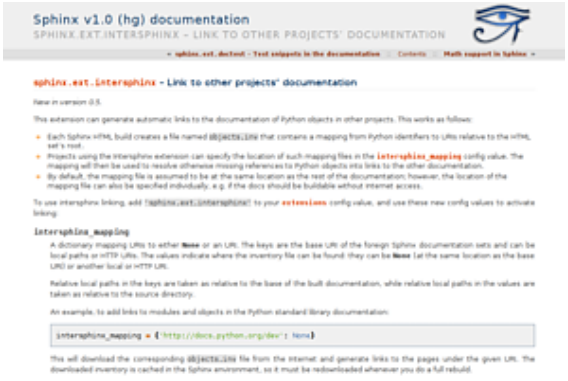

图 5: haiku

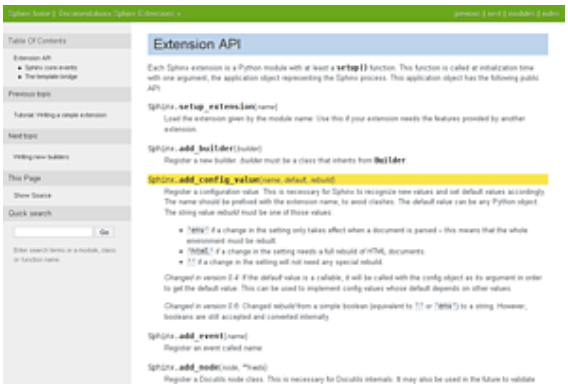

图 6: nature

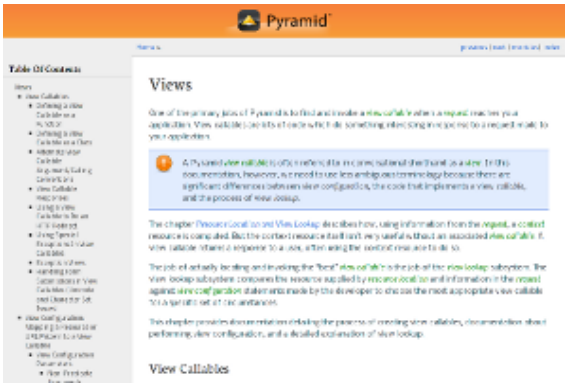

图 7: pyramid

图 8: sphinxdoc

图 9: scrolls

basic

基本上没有样式的布局, 用作其他主题的基础, 也可用作自定义主题的基础

- nosidebar (true or false): 不包括侧边栏. 默认为 False .
- sidebarwidth (int 或 str): 侧边栏的宽度 (以像素为单位). 这可以是 int, 它被解释为像素或有效的 CSS 维度字符串,

图 10: traditional

例如‘70em’或‘50%’. 默认为 230 像素.

- body_min_width (int 或 str): 文档正文的最小宽度. 这可以是 int, 它被解释为像素或有效的 CSS 维度字符串, 例如’70em’或’50%’. 如果您不想要宽度限制, 请使用 0. 默认值可能取决于主题 (通常为 450px).
- body_max_width (int 或 str): 文档正文的最大宽度. 这可以是 int, 它被解释为像素或有效的 CSS 维度字符串, 例如’70em’或’50%’. 如果您不想要宽度限制, 请使用 none . 默认值可能取决于主题 (通常为 800px).

alabaster

来自 @kennethreitz 的修改后的 Kr Sphinx 主题 (Requests 项目中使用), 它本身最初基于 @mitsuhiko 用于 Flask 及相关项目的主题。配置信息

classic

经典主题

- rightsidebar (true or false): 将侧边栏放在右侧。默认为 False
- stickysidebar (true or false): 使侧边栏固定。默认为 False
- collapsiblesidebar (true or false): 添加一个实验性 JavaScript 代码段, 通过侧面的按钮使侧边栏可折叠。默认为 False
- externalrefs (true 或 false): 显示外部链接与内部链接不同。默认为 False

还有各种颜色和字体选项可以更改颜色方案, 而无需编写自定义样式表:

- footerbgcolor (CSS 颜色): 页脚行的背景颜色.
- footertextcolor (CSS 颜色): 页脚行的文本颜色.
- sidebarbgcolor (CSS 颜色): 侧边栏的背景颜色.
- sidebarbtncolor （CSS 颜色: 侧边栏折叠按钮的背景颜色 (当 collapsiblesidebar 为 True 时使用)。
- sidebartextcolor (CSS 颜色): 侧边栏的文本颜色.
- sidebarlinkcolor (CSS 颜色): 侧边栏的链接颜色.

- relbarbgcolor (CSS 颜色): 关系栏的背景颜色.
- relbartextcolor (CSS 颜色): 关系栏的文本颜色.
- relbarlinkcolor (CSS 颜色): 关系栏的链接颜色.
- bgcolor (CSS 颜色): 身体背景颜色.
- textcolor (CSS 颜色): 正文文本颜色.
- linkcolor (CSS 颜色): 正文链接颜色.
- visitedlinkcolor (CSS 颜色): 访问过的链接的正文颜色.
- headbgcolor (CSS 颜色): 标题的背景颜色.
- headtextcolor (CSS 颜色): 标题的文本颜色.
- headlinkcolor (CSS 颜色): 标题的链接颜色.
- codebgcolor (CSS 颜色): 代码块的背景颜色.
- codetextcolor (CSS 颜色): 代码块的默认文本颜色，如果没有通过突出显示样式设置不同.
- bodyfont (CSS 字体系列): 普通文本的字体.
- headfont (CSS 字体系列): 标题的字体.

sphinxdoc

可配置 nosidebar, sidebarwidth

scrolls

一个更轻量级的主题, 基于 Jinja 文档。有以下颜色选项:

- headerbordercolor
- subheadlinecolor
- linkcolor
- visitedlinkcolor
- admonitioncolor

agogo

- bodyfont (CSS 字体系列): 普通文本的字体.
- headerfont (CSS 字体系列): 标题字体.
- pagewidth (CSS 长度): 页面内容的宽度, 默认为 70em.

- documentwidth (CSS 长度): 文档的宽度 (没有侧边栏), 默认为 50em.
- sidebarwidth (CSS 长度): 侧边栏的宽度, 默认为 20em.
- bgcolor (CSS color): 背景颜色.
- headerbg ("background" 的 CSS 值): 标题区域的背景, 默认为浅灰色渐变.
- footerbg ("background" 的 CSS 值): 页脚区域的背景, 默认为浅灰色渐变.
- linkcolor (CSS 颜色): 正文链接颜色.
- headercolor1, headercolor2 (CSS 颜色):<h1> 和 <h2> 标题的颜色.
- headerlinkcolor (CSS 颜色): 标题中后向引用链接的颜色.
- textalign (CSS text-align 值): 正文的文本对齐方式, 默认为 justify.

nature

一个绿色的主题。可配置 nosidebar, sidebarwidth

pyramid

由 Blaise Laflamme 设计的金字塔网络框架项目的主题。

可配置 nosidebar, sidebarwidth

haiku

没有侧栏的主题

- full_logo (true 或 false, 默认为 False): 如果 True, 标题只会显示 html_logo. 用于大型徽标。如果为 False, 则徽标 (如果存在) 将浮动右侧显示, 文档标题将放在标题中.
- textcolor, headingcolor, linkcolor, visitedlinkcolor, hoverlinkcolor (CSS 颜色): 各种身体元素的颜色.

traditional

一个类似于旧 Python 文档的主题。可配置 nosidebar, sidebarwidth

epub

epub 构建器的主题。这个主题试图保留视觉空间, 这是电子书阅读器上的稀疏资源。

- relbar1 (true 或 false, 默认为 True): 如果为 true, 则将 relbar1 块插入 epub 输出中
- footer (true 或 false, 默认为 True): 如果为 true, 则在脚本输出中插入 footer 块

bizstyle

一个简单的蓝色主题。可配置 nosidebar, sidebarwidth, rightsidebar

2.9.3 第三方主题

有许多第三方主题可用。其中一些是一般用途, 而另一些则是针对单个项目的。

可在 PyPI , GitHub 和 sphinx-themes.org 上可以找到更多第三方主题。

sphinx_rtd_theme

Read the Docs Sphinx Theme.

这是一个针对 readthedocs.org 制作的适合移动设备的 sphinx 主题.

2.10 Markdown 支持

为了支持基于 Markdown 的文档，Sphinx 使用 recommonmark 。

recommonmark 是一个 Docutils 桥接器，是用于解析 CommonMark Markdown 风格的 Python 包。

1. 安装 Markdown 解析器 recommonmark

```
pip install --upgrade recommonmark
```

2. 将 recommonmark 添加到扩展名列表

```
extensions = ['recommonmark']
```

3. 调整 source_suffix 变量。

   下面的示例配置 Sphinx 将所有扩展名为.md 和.txt 的文件解析为 Markdown

```
    source_suffix = {
'.rst': 'restructuredtext',
'.txt': 'markdown',
'.md': 'markdown',}
```

4. 进一步配置 recommonmark 以允许标准 CommonMark 不支持的自定义语法

   详阅 recommonmark documentation

2.11 HTML 模板

2.12 sphinx 扩展 ext

2.12.1 内置扩展

2.12.2 外部扩展

2.13 配置文件示例

```
# -*- coding: utf-8 -*-
# test documentation␣
↪build configuration file, created by
# sphinx-
↪quickstart on Sun Jun 26 00:00:43 2016.
#
# This file is execfile()d␣
↪with the current directory set to its
# containing dir.
#
# Note that not all possible␣
↪configuration values are present in this
# autogenerated file.
```

(下页继续)

(续上页)

```
#
# All configuration values␣
↪have a default; values that are commented out
# serve to show the default.

# If extensions (or modules to document␣
↪with autodoc) are in another directory,
# add these directories to sys.path␣
↪here. If the directory is relative to the
# documentation root, use os.path.
↪abspath to make it absolute, like shown here.
#
# import os
# import sys
# sys.path.insert(0, os.path.abspath('.'))

# -- General configuration ---
↪------------------------------------------

# If your documentation needs␣
↪a minimal Sphinx version, state it here.
#
# needs_sphinx = '1.0'

# Add any Sphinx extension␣
↪module names here, as strings. They can be
# extensions coming with␣
↪Sphinx (named 'sphinx.ext.*') or your custom
# ones.
extensions = []

# Add any paths that contain␣
↪templates here, relative to this directory.
templates_path = ['_templates']

# The suffix(es) of source filenames.
# You can␣
↪specify multiple suffix as a list of string:
#
# source_suffix = ['.rst', '.md']
```

(下页继续)

(续上页)

```
source_suffix = '.rst'

# The encoding of source files.
#
# source_encoding = 'utf-8-sig'

# The master toctree document.
master_doc = 'index'

# General information about the project.
project = u'test'
copyright = u'2016, test'
author = u'test'

# The version info for the project␣
↪you're documenting, acts as replacement for
# |version| and |release|, also␣
↪used in various other places throughout the
# built documents.
#
# The short X.Y version.
version = u'test'
# The␣
↪full version, including alpha/beta/rc tags.
release = u'test'

# The language for content autogenerated␣
↪by Sphinx. Refer to documentation
# for a list of supported languages.
#
# This is also used if you␣
↪do content translation via gettext catalogs.
# Usually you set "language
↪" from the command line for these cases.
language = None

# There are two options for replacing␣
↪|today|: either, you set today to some
# non-false value, then it is used:
#
```

(下页继续)

(续上页)

```
# today = ''
#
# Else, today_fmt␣
↪is used as the format for a strftime call.
#
# today_fmt = '%B %d, %Y'

# List of patterns, relative␣
↪to source directory, that match files and
# directories␣
↪to ignore when looking for source files.
# These patterns also␣
↪affect html_static_path and html_extra_path
exclude_patterns␣
↪= ['_build', 'Thumbs.db', '.DS_Store']

# The reST default role␣
↪(used for this markup: `text`) to use for all
# documents.
#
# default_role = None

# If true, '()' will be␣
↪appended to :func: etc. cross-reference text.
#
# add_function_parentheses = True

# If true, the current module␣
↪name will be prepended to all description
# unit titles (such as .. function::).
#
# add_module_names = True

# If true, sectionauthor and␣
↪moduleauthor directives will be shown in the
# output. They are ignored by default.
#
# show_authors = False

# The name of the␣
↪Pygments (syntax highlighting) style to use.
```

(下页继续)

(续上页)

```
pygments_style = 'sphinx'

# A list␣
↪of ignored prefixes for module index sorting.
# modindex_common_prefix = []

# If true, keep warnings as "system␣
↪message" paragraphs in the built documents.
# keep_warnings = False

# If true, `todo` and `todoList`␣
↪produce output, else they produce nothing.
todo_include_todos = False


# -- Options for HTML output -
↪-------------------------------------------

# The theme to use for HTML and␣
↪HTML Help pages.  See the documentation for
# a list of builtin themes.
#
html_theme = 'alabaster'

# Theme options are theme-specific␣
↪and customize the look and feel of a theme
# further.  For a list␣
↪of options available for each theme, see the
# documentation.
#
# html_theme_options = {}

# Add any paths that contain custom␣
↪themes here, relative to this directory.
# html_theme_path = []

# The name for this set of Sphinx documents.
# "<project>
↪ v<release> documentation" by default.
#
```

(下页继续)

(续上页)

```
# html_title = u'test vtest'

# A shorter title for the navigation␣
↪bar.  Default is the same as html_title.
#
# html_short_title = None

# The name of an image file (relative␣
↪to this directory) to place at the top
# of the sidebar.
#
# html_logo = None

# The name of an image file (relative␣
↪to this directory) to use as a favicon of
# the docs.  This file should be a␣
↪Windows icon file (.ico) being 16x16 or 32x32
# pixels large.
#
# html_favicon = None

# Add any paths that contain custom␣
↪static files (such as style sheets) here,
# relative to this directory. They␣
↪are copied after the builtin static files,
# so a file named "default.css
↪" will overwrite the builtin "default.css".
html_static_path = ['_static']

# Add any extra paths that␣
↪contain custom files (such as robots.txt or
# .htaccess) here, relative␣
↪to this directory. These files are copied
# directly to the root of the documentation.
#
# html_extra_path = []

# If not None, a 'Last updated␣
↪on:' timestamp is inserted at every page
# bottom, using the given strftime format.
```

(下页继续)

(续上页)

```
# The␣
↪empty string is equivalent to '%b %d, %Y'.
#
# html_last_updated_fmt = None

# Custom sidebar templates,
↪ maps document names to template names.
#
# html_sidebars = {}

# Additional templates that should␣
↪be rendered to pages, maps page names to
# template names.
#
# html_additional_pages = {}

# If false, no module index is generated.
#
# html_domain_indices = True

# If false, no index is generated.
#
# html_use_index = True

# If true, the index is␣
↪split into individual pages for each letter.
#
# html_split_index = False

# If true, links␣
↪to the reST sources are added to the pages.
#
# html_show_sourcelink = True

# If true, "Created using Sphinx"␣
↪is shown in the HTML footer. Default is True.
#
# html_show_sphinx = True

# If true, "(C) Copyright ..."␣
↪is shown in the HTML footer. Default is True.
```

(下页继续)

(续上页)

```
#
# html_show_copyright = True

# If true, an OpenSearch description␣
↪file will be output, and all pages will
# contain a <link> tag referring␣
↪to it.  The value of this option must be the
# base␣
↪URL from which the finished HTML is served.
#
# html_use_opensearch = ''

# This is the file␣
↪name suffix for HTML files (e.g. ".xhtml").
# html_file_suffix = None

# Language to be used for␣
↪generating the HTML full-text search index.
# Sphinx supports the following languages:
#   'da', 'de
↪', 'en', 'es', 'fi', 'fr', 'hu', 'it', 'ja'
#   'nl
↪', 'no', 'pt', 'ro', 'ru', 'sv', 'tr', 'zh'
#
# html_search_language = 'en'

# A dictionary with options for the␣
↪search language support, empty by default.
# 'ja' uses this config value.
# 'zh' user␣
↪can custom change `jieba` dictionary path.
#
# html_search_options = {'type': 'default'}

# The name of a javascript file (relative␣
↪to the configuration directory) that
# implements a search results␣
↪scorer. If empty, the default will be used.
#
# html_search_scorer = 'scorer.js'
```

(下页继续)

(续上页)

```
# Output file base name for HTML help builder.
htmlhelp_basename = 'testdoc'

# -- Options for LaTeX output␣
↪--------------------------------------------

latex_elements = {
    #␣
↪The paper size ('letterpaper' or 'a4paper').
    #
    # 'papersize': 'letterpaper',

    # The font size ('10pt', '11pt' or '12pt').
    #
    # 'pointsize': '10pt',

    # Additional stuff for the LaTeX preamble.
    #
    # 'preamble': '',

    # Latex figure (float) alignment
    #
    # 'figure_align': 'htbp',
}

# Grouping the document␣
↪tree into LaTeX files. List of tuples
# (source start file, target name, title,
#  author,␣
↪documentclass [howto, manual, or own class]).
latex_documents = [
    (master_
↪doc, 'test.tex', u'test Documentation',
     u'test', 'manual'),
]

# The name of an image file (relative␣
↪to this directory) to place at the top of
# the title page.
```

(下页继续)

(续上页)

```
#
# latex_logo = None

# If true,
↪ show page references after internal links.
#
# latex_show_pagerefs = False

# If true,
↪ show URL addresses after external links.
#
# latex_show_urls = False

# Documents␣
↪to append as an appendix to all manuals.
#
# latex_appendices = []

# If false, no module index is generated.
#
# latex_domain_indices = True


# -- Options for manual page output␣
↪--------------------------------------

# One entry per manual page. List of tuples
# (source start file,
↪ name, description, authors, manual section).
man_pages = [
    (master_doc, 'test', u'test Documentation',
     [author], 1)
]

# If true,
↪ show URL addresses after external links.
#
# man_show_urls = False
```

(下页继续)

(续上页)

```
# -- Options for Texinfo output␣
↪-------------------------------------------

# Grouping the document␣
↪tree into Texinfo files. List of tuples
# (source␣
↪start file, target name, title, author,
#  dir menu entry, description, category)
texinfo_documents = [
    (master_doc, 'test', u'test Documentation',
     author,
↪ 'test', 'One line description of project.',
     'Miscellaneous'),
]

# Documents␣
↪to append as an appendix to all manuals.
#
# texinfo_appendices = []

# If false, no module index is generated.
#
# texinfo_domain_indices = True

# How to display␣
↪URL addresses: 'footnote', 'no', or 'inline'.
#
# texinfo_show_urls = 'footnote'

# If true, do not generate␣
↪a @detailmenu in the "Top" node's menu.
#
# texinfo_no_detailmenu = False

# -- A random example --------
↪----------------------------------------

import sys, os
sys.path.insert(0, os.path.abspath('.'))
exclude_patterns = ['zzz']
```

(下页继续)

(续上页)

```
numfig = True
#language = 'ja'

extensions.append('sphinx.ext.todo')
extensions.append('sphinx.ext.autodoc')
#extensions.append('sphinx.ext.autosummary')
extensions.append('sphinx.ext.intersphinx')
extensions.append('sphinx.ext.mathjax')
extensions.append('sphinx.ext.viewcode')
extensions.append('sphinx.ext.graphviz')



autosummary_generate = True
html_theme = 'default'
#source_suffix = ['.rst', '.txt']
```

CHAPTER 3

reStructureText 语法

3.1 内联标记 Inline Markup

语法	效果	说明
`*emphasis*`	*emphasis*	强调，斜体
`**strong**`	**strong**	加粗
‘interpreted-text‘	*interpreted-text*	使用两个反逗号包裹内容，表征对其解释
‘‘inline literal‘‘	`inline literal`	当包括内容包含空格时使用两个反逗号来包裹
`ref_ .. _ref:` 链接	ref	链接：纯文本，外部链接
‘phrase ref‘__	带空格外链	链接：文字间有空格标点，外部链接
`anonymous__`	anonymous	链接: 匿名链接
`_`inline_link`	inline_link	交叉引用链接
`|substitution ref|`	|substitution ref|	指示链接 (图片，链接等)
`footnote [1]_`	footnote[1]	脚注（包括参考文献）
`citation [CIT2002]_`	`citation [CIT2002]`	引用
http://docutils.sf.net/	http://docutils.sf.net/	独立链接

3.2 反斜杠转义 Escaping with Backslashes

使用反斜杠来转义任意 rst 语法符号为符号本身，包括反斜杠自己

[1] 脚注 1

- 转义内联标记：* 去除斜体 *
- 转义反斜杠（用两个反斜杠）：\\

在 python 中，最简单的方式是在字符串外表示 raw strings(加 r)

py 字符串	显示结果
r”“” * 去除斜体 * “”“” “”	r”“” * 去除斜体 * “”“” “”
“” “* 去除斜体 * “\”“” “”	“” “* 去除斜体 * “”“” “”
“” “* 去除斜体 * “”“” “”	“” “* 去除斜体 * “”“” “”

3.3 章节标记 Section Structure

任意可打印的 7 个 bit 的 ASCII 码字符都可以作为章节标识符，它们是

```
! " # $ % & ' ( ) * + , - . / : ; < = > ? @ [ \ ] ^ _ ` { | } ~
```

不过有些可能会看起来比较奇怪，因此推荐使用其中的

```
= - ` : . ' " ~ ^ _ * + #
```

- 在 reStructureText 中未明确各个章节标识符层级的顺序，**它按照标识符在书写文本中的顺序来指定标识符指示的标题层级**。
- 在标题上下，使用两行标识符；和只在标题下使用一行标识符。效果是一样的。
- 标题标识符的数量至少要和标题文本等长
- 建议定义如下标题标识符层级（从高到低）为 = - , . *

可以使用如下标准定义各级标题

```
一级标题
==========
二级标题
----------
三级标题
,,,,,,,,,
四级标题
.............
五级标题
**************
```

3.4 段落 Paragraphs

段落一般隶属于某个章节中，是一块左对齐并且没有其他元素体标记的块。在.rst 文件中，段落和其他内容的分割是靠 **空行**来完成

> 如果段落相较于其他的段落有 **缩进 **（这段缩进了 4 个空格），reStructuredText 会解析为 ** 引用段落**，样式上有些不同。

3.5 无序列表 Bullet Lists

reStructuredText 中 无序列表 的语法和 Markdown 中的是一样的。一般使用 "+ ","- ","* " 来作为无序列表的指示符，利用缩进来指示列表之间的嵌套关系。

- 列表的起始项和终止项前后是需要空行的
- 同层级之间的列表项之间空行可加可不加，不同层级之间的列表项必须加空行
- 层级中内容有多段，则第二段（后续段落内容）无需指示列表标识符，只需保持同样缩进（即左对齐）。
- 指示标识符可以混用，但是不推荐，推荐同样层级使用同样符号，一般层级顺序就按照 "+ ","- ","* "

示例如下:

```
+ 一级列表第一项
+ 一级列表第二项

  - 二级列表，换层级加空行

    二级列表的第二段内容，加空行，缩进对齐

  + 依然是二级列表，指示标识符可以混用，但不推荐

    * 第三级
```

效果如下:

- 一级列表第一项
- 一级列表第二项
  - 二级列表，换层级加空行

    二级列表的第二段内容，加空行，缩进对齐
  - 依然是二级列表，指示标识符可以混用，但不推荐
    * 第三级

3.6 有序列表 Enumerated Lists

枚举列表 即顺序列表 (ordered-list)，可以使用不同的枚举序号来表示列表。

枚举指示符有：

- 阿拉伯数字: 1, 2, 3, ⋯(无上限)。
- 大写字母: A-Z。
- 小写字母: a-z。
- 大写罗马数字: I, II, III, IV, ⋯, MMMMCMXCIX (4999)。
- 小写罗马数字: i, ii, iii, iv, ⋯, mmmmcmxcix (4999)。

并且可以使用“#”来自动自增。

支持添加的前缀和后缀：

- . 后缀: “1.”，“A.”，“a.”，“I.”，“i.”。
- () 包起来: “(1)”，“(A)”，“(a)”，“(I)”，“(i)”。
-) 后缀: “1)”，“A)”，“a)”，“I)”，“i)”。

当正常的文本中包含可被识别为列表的内容时（`A. 1. (b) I` 等），为了避免被识别，可以采取如下措施：

1. 将一行内容，折断为多行书写，这样会被识别为 **段落**内容，而不会解析为列表；
2. 使用反斜杠“”在段首进行转义。

示例如下：

```
A. Einstein was a really
smart dude. (跨行避免)

A. Einstein was a really smart dude. (未避免)

\A. Einstein was a really smart dude. (使用\转义)
```

效果如下：

A. Einstein was a really smart dude.

A. Einstein was a really smart dude.

A. Einstein was a really smart dude.

有序列表也支持嵌套，规则和无序列表一致

```
1. Item 1 initial text.
```

(下页继续)

(续上页)

```
   a) Item 1a.
   b) Item 1b.

#. a) Item 2a. 使用# 号自增
   b) Item 2b.
```

效果如下:

1. Item 1 initial text.

   a) Item 1a.

   b) Item 1b.

2. a) Item 2a. 使用 # 号自增

   b) Item 2b.

3.7 定义列表 Definition Lists

定义列表 可以理解为解释列表，即名词解释 (definition_list, classifier, definition)。

条目占一行，解释文本要有缩进；多层可根据缩进实现。

各个条目由三部分组成，条目名称 (term)，条目属性 (classifier)，条目定义 (definition)，条目名称和条目属性在同一行，使用空格、冒号、空格（" : "）连接，其中条目属性可以为空，也可以有多个条目定义需要换行缩进。

示例如下:

```
term 1
  Definition 1.

term 2
    Definition 2, paragraph 1.

    Definition 2, paragraph 2.

term 3 : classifier
    Definition 3.

term 4 : classifier one : classifier two
    Definition 4.
```

效果如下:

term 1 Definition 1.

term 2 Definition 2, paragraph 1.

Definition 2, paragraph 2.

term 3 [classifier] Definition 3.

term 4 [classifier one][classifier two] Definition 4.

3.8 字段列表 Field Lists

字段列表 用于指令解释，或者数据库字段（记录）解释的场景。

它在形式上有点像两列的表格，因此在 field body 中的功能是和在表格中一样的（即支持嵌套，跨行等等）。

示例如下：

```
:Date: 2020-02-02
:Version: 1
:Authors: - fire
          - firewang
          - firewang
:Indentation: Since the field marker may be quite long, the second
   and subsequent lines of the field body do not have to line up with first line.
   解释可能很长，第二行不用和第一行对齐，但是后续行必须和第二行对齐。
:Parameter i: field name 可以是 phrase，即可以带空格，但是不能带":"
```

效果如下：

Date 2020-02-02

Version 1

Authors

- fire
- firewang
- firewang

Indentation Since the field marker may be quite long, the second and subsequent lines of the field body do not have to line up with first line. 解释可能很长，第二行不用和第一行对齐，但是后续行必须和第二行对齐。

Parameter i field name 可以是 phrase，即可以带空格，但是不能带”:”

3.9 参数 (选项) 列表 Option Lists

选项列表是一个左列为参数，右列为参数说明的两列列表（无表头），用于 command-line 参数解释。

支持三种参数书写形式:

- 由一个短横（Short dash）连接的 POSIX 式。
- 由两个短横（Short dash）连接的长 POSIX 式。
- DOS/VMS 参数形式，即由 / 起始的参数形式。

示例如下：

```
-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
```

效果如下：

-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

3.10 文字块 Literal Blocks

文字块 就是一段文字信息，指示符为连续两个冒号 :: ，支持文字块的嵌套。

文字块支持三种形式的语法 (完全等价)

1. 起始新行，后接空行，块内容需缩进

示例如下：

```
::

  缩进后填写块内容
```

效果如下：

```
缩进后填写块内容
```

2. 部分简化，前文带一个冒号，加一个空格后，双冒号接在前文后面，不另起行，同时会显示单个冒号，块内容同样缩进

示例如下：

```
这里是前面内容，下面引用：::

  缩进后填写块内容
```

效果如下：

这里是前面内容，下面引用:

```
缩进后填写块内容
```

3. 完全简化，双冒号接在前文后面，不另起行，同时会显示单个冒号，块内容同样缩进

示例如下：

```
这里是前面内容，下面引用::

> 在（部分/完全）简化形势下支持单行引用形式的嵌套
> 再来一个单行引用
```

效果如下：

这里是前面内容，下面引用:

```
> 在简化形势下支持单行引用形式的嵌套
> 再来一个单行引用
```

3.11 行块 Line Blocks

```
| 行块使用 | 指示符，
| 一般用于描述地址，歌
  词，诗歌，简单列表等。
```

效果如下：

行块使用“|”指示符，
一般用于描述地址，歌词，诗歌，简单列表等。

3.12 块引用 Block Quotes

块引用是 **通过缩进来实现**的，引用块要在前面的段落基础上缩进。

通常引用结尾会加上出处 (attribution)，出处的文字块开头是两个或者三个连续短横（” –” ,” —”）后面加上出处信息。

块引用可以使用空的注释.. 分隔上下的块引用。

注意在新的块和出处都要添加一个空行。

示例如下：

```
实际效果：

    “真的猛士，敢于直面惨淡的人生，敢于正视淋漓的鲜血。”

    --- 鲁迅

..

    “人生的意志和劳动将创造奇迹般的奇迹。”

    -- 涅克拉索
```

实际效果：

> “真的猛士，敢于直面惨淡的人生，敢于正视淋漓的鲜血。”
>
> —鲁迅

> “人生的意志和劳动将创造奇迹般的奇迹。”
>
> —涅克拉索

3.13 文档测试块 Doctest Blocks

文档测试块是交互式的 Python 会话，以 >>> 开始，一个空行结束，是一种特殊的文字块，**内容不需要缩进**。

可直接复制到 python 的 docstrings 中，用于为 doctest 模块提供测试环境。

当文字块语法和文档测试块语法同时出现时，文字块语法优先级更高。

```
>>> print('this is a Doctest block')
this is a Doctest block
```

3.14 表格 Tables

reStructureText 提供两种表格：网格表格（Grid Tables），简单表格（Simple Tables）。

表格前后都需要空行

3.14.1 网格表格

- “-” 分隔行 (短破折号，减号)
- “=” 分隔表头和表体行
- “|” 分隔列
- “+” 表示行和列相交的节点

网格表格注意点：

- 网格表格编辑复杂，可以使用 Emacs 来编辑生成
- 行和列都支持并格
- 如果文本内包含”|”，并且恰好与表格内分隔对齐了，那么会产生错误。解决方案：方式一是加空格避免对齐，方式二是为该行增加一行
- 可以不包含表头。
- 列需要和”=”左对齐，不然可能会导致出错
- 如果碰到第一列为空，需要使用“” 来转义, 不然会被视为是上一行的延续。

示例：

```
+------------------------+------------+----------+----------+
| Header row, column 1   | Header 2   | Header 3 | Header 4 |
| (header rows optional) |            |          |          |
+========================+============+==========+==========+
| body row 1, column 1   | column 2   | column 3 | column 4 |
+------------------------+------------+----------+----------+
| body row 2             | Cells may span columns.          |
+------------------------+------------+---------------------+
| body row 3             | Cells may  | - Table cells       |
+------------------------+ span rows. | - contain           |
| body row 4             |            | - body elements.    |
+------------------------+------------+---------------------+
```

结果：

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.	• Table cells • contain • body elements.	
body row 4			

3.14.2 简单表格

简单表格使用“=”和“_”来进行绘制，其中”=”放置于表格的最外两行（首行和末行），如果有表头，则表头也用该符号进行分隔，”_”用于跨列合并（column span）。

简单表格需要各列首字母与该列指示的”=”对齐 (表头可不对齐，为了保持统一，尽量保持左对齐)，每列的”=”需要覆盖该列字符的长度

包含表头的简单表格

语法如下：

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

效果如下：

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

无表头的简单表格

语法如下：

```
=====  =====  =======
False  False  False
True   False  False
False  True   False
True   True   True
=====  =====  =======
```

效果如下：

False	False	False
True	False	False
False	True	False
True	True	True

跨列合并

“_”用于跨列合并，**仅支持在表头使用**，”_”长度需要从起始列的第一个指示符”=”到终止列的最后一个指示符”=”

语法如下：

```
=====  =====  ======
合并两列       单独列
-----------  ------
  A      B    A or B
=====  =====  ======
False  False  False
True   False  True
False  True   True
True   True   True
=====  =====  ======
```

效果如下：

合并两列		单独列
A	B	A or B
False	False	False
True	False	True
False	True	True
True	True	True

单个表格中可以多行

- 简单表格的单个格子中可以包含多行的内容（比如列表），但是不支持行合并；
- 增加空行可以进行换行，否则会自动将文本连接在一起。
- 首列不能为空，为空时使用 \ 进行占位。

语法如下：

```
=====  ===================================
col 1  col 2
=====  ===================================
1      Second column of row 1.
2      Second column of row 2.
       Second line of paragraph.
3      - Second column of row 3.

       - Second item in bullet
         list (row 3, column 2).
\      Row 4; column 1 will be empty.
=====  ===================================
```

效果如下：

col 1	col 2
1	Second column of row 1.
2	Second column of row 2. Second line of paragraph.
3	• Second column of row 3. • Second item in bullet list (row 3, column 2).
	Row 4; column 1 will be empty.

3.15 Transitions

转换分隔用于段与段之间的分隔，相当于 html 中的 <hr>，就是跨屏的一个横线。

使用 4 个及以上的标点符号 (推荐使用短横 "- ") 就可以生成，同样需要前后空行，另外，**不能连续出现**，不能在文档结尾使用。

示例如下：

```
前后需要空行

,,,,,,,,

使用标点符号

............

不能连续出现

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

不能在结尾使用
```

效果如下：

前后需要空行

使用标点符号

不能连续出现

不能在结尾使用

3.16 脚注 Footnotes

脚注 有三种形式，手工序号 (标记序号 123 之类)、自动序号 (填入 # 号会自动填充序号)、自动符号 (填入 * 会自动生成符号)

手工序号可以和 # 结合使用，会自动延续手工的序号。

表示的方法可以在后面加上一个名称，这个名称就会生成一个链接。

1. 手工标序 (标记序号 123 之类)

示例如下：

```
Footnote references, like [5]_.
Note that footnotes may get [3]_
rearranged, e.g., to the bottom of
the "page".
```

(下页继续)

(续上页)

```
.. [5] A numerical footnote. Note
   there's no colon after the ].
.. [3] 脚注 3
```

效果如下：

Footnote references, like[5]. Note that footnotes may get[3] rearranged, e.g., to the bottom of the "page".

2. 自动序号 (填入 # 号会自动填充序号)

示例如下：

```
自动排序脚注, like using [#]_ and [#]_.
.. [#] This is the first one.
.. [#] This is the second one.
```

效果如下：

自动排序脚注, like using[2] and[4].

可以添加别名，即可同时实现自动排序，又带有自定义名称，**这个功能相当于实现了文献引用功能**；

示例如下：

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

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

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

效果如下：

They may be assigned 'autonumber labels' - for instance,[7] and[6].

3. 自动符号 (填入 * 会自动生成符号)。

自动填符号功能上和自动填序号是一样的，只是换了一种辨识符号。

示例如下：

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

[3] 脚注 3

[2] This is the first one.

[4] This is the second one.

[7] a.k.a. *fourth*

[6] a.k.a. *third*

```
自动脚注符号, like this: [*]_ ,[*]_ , [*]_ and [*]_.

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

效果如下：

自动脚注符号, like this:*[0] ,[*]_ ,†[0] and‡[0].

3.17 引用 Citations

引用和脚注是一样的，只不过引用只能用文本而不能用数字。

示例如下：

```
引用参考的内容通常放在页面结尾处，比如 [One]_, Two_

.. [One] 参考引用一
.. [Two] 参考引用二
```

效果如下：

引用参考的内容通常放在页面结尾处，比如 [One]，[Two]

3.18 超链接 Hyperlink Targets

超链接 Hyperlink 有三种：

- 带别名的超链接，语法为 `.. _hyperlink-name: link-address` ；由 `..`，空格，短下划线”`_`”，别名，冒号，空格和链接地址构成。在原文引用处书写语法为 `hyperlink-name_` （特别注意原文中”`_`”在别名后，而在指示链接出，”`_`”在别名前）。
- 匿名 anonymous 的超链接，即不带别名的超链接，语法为 `.. __: link-address` ；由 `..`，空格，两个短下划线”`___`”，冒号，空格和链接地址构成。
- 匿名的超链接，另一种语法形式，语法为 `__ link-address` 。

[0] This is the first one.
[0] This is the second one.
[0] This is the third one.

3.18.1 外部链接 External Hyperlink Targets

外部链接有两种方式，需要引用的话，使用上述带别名的超链接的语法形式，即

示例如下：

```
这是我的 reStructureText_ 实践笔记。

.. _reStructureText: https://sphinx-practise.readthedocs.io/zh_CN/latest/index.html
```

效果如下：

这是我的 reStructureText 实践笔记。

另一种是直接在名称后附加地址，语法为 ‘别名 < 链接 >‘_

示例如下：

```
这是我的 `reStructureText <https://sphinx-practise.readthedocs.io/zh_CN/latest/index.
↪html>`_ 实践笔记。
```

效果如下：

这是我的 reStructureText 实践笔记。

3.18.2 锚点链接 Internal Hyperlink Targets

内部超链接，即锚点。

锚点的语法即外部超链接中*带别名的超链接* 去除外部链接，其他语法一致。

3.18.3 间接链接 Indirect Hyperlink Targets

间接超链接是基于匿名链接的基础上的，就是将匿名链接地址换成了外部引用名。

示例如下：

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

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

__ Python_
```

效果如下：

Python is my favourite programming language.

其中 python_ 就是一个正常的外部链接，而后面那句话是一个匿名链接，对这个匿名链接使用间接链接方式链接到 Python 这个外部链接的链接地址上去。

3.18.4 Implicit Hyperlink Targets

隐式超链接

小节标题、脚注和引用参考会自动生成超链接地址，使用小节标题、脚注或引用参考名称作为超链接名称就可以生成隐式链接。

本质上它们的写法都是和外部链接 *External Hyperlink Targets* 相一致的，只是做了一些微小改动，以做出区别。

例如链接到超链接 *Hyperlink Targets* 这个章节目录去

```
`超链接 Hyperlink Targets`_
```

3.19 扩展指令 Directives

指令（Directives）是 reStructureText 的扩展机制。可以在不增加新语法的情况下，增加新的结构性支持 (a way of adding support for new constructs)。

指令由三部分组成

```
.. directive-type:: directive-block
```

其中指令类型（directive-type）指明指令的类型，指令内容体又由三部分组成

- 指令作用对象 Directive arguments：指明该指令针对哪个对象作用
- 指令选项参数 Directive options：该指令的可选参数（可选），是一个参数列表
- 指令内容说明 Directive content：说明文档（可选）

比如插入一个图片

```
.. figure:: 图片名.png  # 这里是指令作用对象
   :scale: 50      # 这里是指令参数
   :width: 100

   这是一个图片    # 这里是说明
```

已在 reference reStructuredText parser 中实现的指令。

3.19.1 警告信息 Admonitions

特定的警告信息

格式为

```
.. admonition:: admonition-title(可空)
   :class: class-name(可选)
       :name: name(可选)
   admonition-content 说明信息
```

admonition-title 和 admonition-content 显示效果是一体的，但是 admonition-title(可空) 会在 html 中单独存在一个 title 标签中。

支持如下特定警告信息

- attention
- caution
- danger
- error
- hint
- important
- note
- tip
- warning

示例如下：

```
.. attention:: This is a attention admonition.
   second attention paragraph.

.. caution:: This is a caution admonition.
   second caution paragraph.

.. danger:: This is a danger admonition.
   second danger paragraph.

.. error:: This is a error admonition.
   second error paragraph.
```

(下页继续)

(续上页)

```
.. hint:: This is a hint admonition.
   second hint paragraph.

.. important:: This is a important admonition.
   second important paragraph.

.. note:: This is a note admonition.
   This is the second line of the first paragraph.

   - The note contains all indented body elements
     following.
   - It includes this bullet list.

.. tip:: This is a tip admonition.
   second tip paragraph.

.. warning:: This is a warning admonition.
   second warning paragraph.
```

效果如下：

注意： This is a attention admonition. second attention paragraph.

警告： This is a caution admonition. second caution paragraph.

危险： This is a danger admonition. second danger paragraph.

错误： This is a error admonition. second error paragraph.

提示： This is a hint admonition. second hint paragraph.

重要: This is a important admonition. second important paragraph.

注解: This is a note admonition. This is the second line of the first paragraph.

- The note contains all indented body elements following.
- It includes this bullet list.

小技巧: This is a tip admonition. second tip paragraph.

警告: This is a warning admonition. second warning paragraph.

通用警告信息 Generic Admonition

通用警告信息即不指定为特定的警告类别，使用 admonition 指代警告。与特定警告不同的是，特定警告的 admonition-title 在通用警告中为 admonition-name，这是我们自定义的警告名，用于和特定警告（danger,hint,important 等）提供同等标识。

示例如下：

```
.. admonition:: And, by the way...

   You can make up your own admonition too.
```

结果如下：

And, by the way…

You can make up your own admonition too.

3.19.2 图片 Images

使用 image

```
.. image:: picture.jpeg
   :class: class-name
```

(下页继续)

(续上页)

```
      :name: name
   :height: 100 px(长度)
   :width: 200 px (长度或者百分比)
   :scale: 50 % (百分比, 百分号可省略)
   :alt: alternate text
   :align: right
      :target: https://www.baidu.com
```

align 可选 top,middle,bottom,left,center,right

target 使得图片可点击跳转。

scale 表示等比例伸缩（放大或者缩小）

重要: scale 需要和 width 或者 height（或者 2 者）一起使用。

使用 figure

```
.. figure:: picture.png
   :figwidth: 200 px (长度或者百分比)
   :scale: 50 %
      :align: center
      :figclass: figure-class
   :alt: map to buried treasure

    +---------------------------+
    |        figure             |
    |                           |
    |<------ figwidth --------->|
    |                           |
    |  +---------------------+  |
    |  |     image           |  |
    |  |                     |  |
    |  |<--- width --------->|  |
    |  +---------------------+  |
    |                           |
    |The figure's caption should|
    |wrap at this width.        |
    +---------------------------+
```

figure 相当于一个画布（类似于 html 中的一个 div 或者一个 canvas），它对处于其内的内容进行样式统一管

理。相比 image 可以包含除图片外的更多内容。

figure 支持 image 的所有指令选项参数，除了 align 在 figure 中指示整个画布的对齐方式。且它只能选择为 left,center,right。

重要： 和 image 一致，要使得 scale（这里是对整个画布作用）起作用需要和 figwidth 一起使用

3.19.3 页面元素 Body Elements

3.19.4 表格 Tables

3.19.5 文档 Documents

3.19.6 References

3.19.7 HTML-Specific

3.19.8 Substitution Definitions

3.19.9 其他

3.19.10 通用指令选项参数

:class:	得到
:name:	为指令设置名称 (可用于简化别名链接)

```
.. image:: build.png
   :name: my pic
与下列方式等价
.. _my pic

.. image:: build.png
```

3.20 Substitution References and Definitions

3.21 Comments

非上述语法，则都作为 Comments 处理。

CHAPTER 4

Reference

4.1 编辑工具

https://github.com/steenhulthin/reStructuredText_NPP notepad++ 插件

http://rst.ninjs.org/ 在线

4.2 参考内容

4.2.1 sphinx

https://www.sphinx.org.cn/

https://www.sphinx-doc.org/en/master/

https://zh-sphinx-doc.readthedocs.io/en/latest/

4.2.2 reStructureText

https://docutils.sourceforge.io/docs/user/rst/quickref.html

https://blog.csdn.net/liuskyter/article/details/86570790

CHAPTER 5

Pygments

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Bibliography

[One] 参考引用一

[Two] 参考引用二