用 Sphinx 写书

发布 1.0

HYRY Studio

2011 年 07 月 28 日

Contents

前言

写技术书是一件十分费时费力的事情，作者不但需要编写有趣的内容，还需要用标准且美观大方的格式呈现内容。在编写《Python 科学计算》一书的过程中，我尝试使用 Sphinx、Leo、MiKTeX 等软件，拼凑出了一套适合编写技术书籍和文档的编写环境。这本书是关于这个编写环境的一些介绍。

第 1 章

安装编辑环境

1.1 Python 和 Sphinx

首先到 Python 的官方网站下载并安装 2.6 或 2.7 系列的 Python 运行环境。

http://python.org/getit/
Python 的下载地址

Sphinx 是一套使用 reStructuredText 作为标记语言的文档生成工具。它是 Python 的标准库，因此通常情况下不需要再安装它。如果需要单独升级 SPhinx 库，可以在控制台中输入如下命令：

```
easy_install -U sphinx
```

reStructuredText

restructuredText 是一种简单易用的所见即所得的纯文本标记语法。可以通过转换工具将其转换为 HTML、latex、PDF 等多种格式。通常 restructuredText 的扩展名为''rst''。

1.2 Leo

Leo 是一个用 Python 编写的提纲式程序编辑器，我们用它组织和编辑构成书籍内容的 reStructuredText 文档，并管理 Sphinx 的插件程序、HTML 模板以及配置文档等。

http://sourceforge.net/projects/leo/files/Leo/
Leo 的下载地址，可以下载源程序版或打包版，由于系统中已经安装了 Python 环境，因此推荐安装源程序版

1.3 MiKTeX

http://miktex.org/
MiKTex 是一个 Windows 下的 Tex 编译环境，我们用它将 Sphinx 自动生成的 LaTex 源文件编译成 PDF 文件

安装完成之后，执行：

```
xelatex sample.tex
```

就可以将“sample.tex”编译成“sample.pdf”。

1.4 书籍目录

http://hyry.dip.jp/files/books.zip
下载本书的编辑环境

编写书籍项目的目录结构如下：

```
[books]
    master.leo      -- 管理所有内容的leo文件
    [exts]          -- 插件和模板
    [sphinxbook]    -- 本书的文件夹
    [xxxbook]       -- 其它书籍的文件夹
```

其中 exts 文件夹中包含了所有 Sphinx 插件程序以及 LaTex 和 HTML 的模板。而其它文件夹均为 Sphinx 书籍的文件夹。每本书籍的目录结构如下：

```
[sphinxbook]
    make.bat        -- 编译书籍的批处理脚本
    [source]        -- 书籍的源文件
        conf.py     -- 书籍配置
        *.rst       -- 各个章节的reStructuredText文件
        [images]    -- 保存所有插图的文件夹
        [codes]     -- 保存所有代码的文件夹
    [build]
        [latex]     -- PDF的编译输出文件夹
        [html]      -- HTML的编译输出文件夹
```

在书籍文件夹下运行“make.bat html”命令将书籍编译成 HTML 格式，而运行“make.bat latex”则编译成 LaTex 格式。这些命令可以通过 Leo 的按钮工具栏 (第 3.1 节) 运行。

为了保证程序能正常运行，请保证所有路径中没有空格或中文。

第 2 章

扩展程序

为了让最终的作品格式更美观和规范，我们提供了一些 Sphinx 插件程序和模板，本章对这些插件和模板进行介绍。为了让 Sphinx 能找到插件和模板，需要编辑书籍项目的配置文件“conf.py”中的路径设置。在此文件开头添加：

```
_exts = "../../exts"
sys.path.append(os.path.abspath(_exts))
```

并修改 HTML 模板相关的配置：

```
html_theme = 'book'
html_theme_path = [_exts +"./theme"]
```

2.1 LaTeX 的编号

number_ref.py
为 LaTeX 文件添加带编号的章节和插图参照，适合制作印刷版的 PDF 文档

需要进行编号的插图使用以“fig”开头的标签，例如：

```
\ :ref:`fig-leo`\ 是Leo 4.9的界面截图。

.. _fig-leo:

.. figure:: images/leo.png
    :width: 12.0cm
```

```
Leo 的界面截图
```

在 Leo 编辑器中，可以输入“fig>leo”并按 CTRL+1，快速生成上面的代码。

需要进行引用的章节可以用以“sec”开头的标签，例如：

```
章节名
======

.. _sec-test:

这是一个章节。

这是一个引用：\ :ref:`sec-test`\ 。
```

例如：关于书籍目录的相关说明请阅读第 1.4 节。

为了让章节标签包含在章节内部，本插件对以“sec”开头的标签进行特殊处理，因此可以在章节名之下定义标签。

2.2 代码说明标签

number_labe.py
为代码添加如“❶ ❷ ”的说明标签

为了对代码中的重要语句进行说明，本插件对代码中的“#❶ ”等进行处理。例如：

```
import os
print os.getcwd() ❶
print os.environ  ❷
```

❶ 输出当前路径，❷ 输出环境变量。

在 Leo 编辑器中，可以通过输入数字并按 CTRL+1，快速输入“❶ ❷ ”等符号。如果通过“literalinclude”命令从外部文件载入代码段，则可以在代码中使用“#{1}”、“#{2}”等标签，它们会自动被转换为对应的数字符号。

目前此功能只支持 Python 语言。

2.3 带图标的块

block.py
可以在文章中间插入带图的块

本扩展程序提供了 5 种图片块，例如：

```
.. ttip::

    这个一个小提示。
```

生成：

这个一个小提示。

```
.. tcode::

    example.py

    这是一个例子程序
```

生成：

example.py
这是一个例子程序

```
.. twarning::

    警告，如果你看到这个警告，那么请无视它。
```

生成：

警告，如果你看到这个警告，那么请无视它。

```
.. tlink::

    http://hyry.dip.jp

    欢迎访问我们的主页
```

生成：

http://hyry.dip.jp
欢迎访问我们的主页

```
.. tanim::

    demo.avi
```

```
    这是一个动画演示文件
```

生成：


demo.avi
这是一个动画演示文件

为了添加新的图标块命令“tnews”，需要准备两个图标文件：“news.png”和“news.pdf”，将它们分别放到下面两个目录中：

```
exts\latexstyle\news.pdf
exts\theme\book\static\news.png
```

并编辑“block.py”文件，在其中的 setup() 中添加：

```
app.add_directive('tnews', MakeFileDirective("tnews"))
```

2.4 HTML 的中文分词


chinese_search.py
增加 HTML 的中文搜索功能

本扩展程序使用SmallSeg对中文进行分词。

2.5 插入代码片段


literal_include.py
修改 literalinclude 命令，为其添加 section 选项，可从源程序中载入文件中的部分源代码

例如程序“example.py”的内容如下：

```
class Directive(object):
    """
    Fake Directive class to allow Sphinx directives to be written in
    class style.
    """
    required_arguments = 0
    optional_arguments = 0
    final_argument_whitespace = False
    option_spec = None
    has_content = False
    ###1###
    def __init__(self, name, arguments, options, content, lineno,
                 content_offset, block_text, state, state_machine):
        self.name = name
```

```
        self.arguments = arguments  ❶
        self.options = options      ❷
        self.content = content
        self.lineno = lineno        ❸
        self.content_offset = content_offset
        self.block_text = block_text
        self.state = state
        self.state_machine = state_machine
    ###1###
    ###2###
    def run(self):
        raise NotImplementedError('Must override run() is subclass.')
    ###2###
```

使用下面的命令可以载入其中以“###1###”包围的部分。

```
.. literalinclude:: codes/example.py
    :section: 1
```

结果为：

```
def __init__(self, name, arguments, options, content, lineno,
             content_offset, block_text, state, state_machine):
    self.name = name
    self.arguments = arguments  ❶
    self.options = options      ❷
    self.content = content
    self.lineno = lineno        ❸
    self.content_offset = content_offset
    self.block_text = block_text
    self.state = state
    self.state_machine = state_machine
```

第 3 章

Leo 编辑器

我们使用 Leo 编辑器管理和编辑所有的扩展程序和 rst 文件，图 3.1 是用 Leo 4.9 打开“master.leo”时的画面。

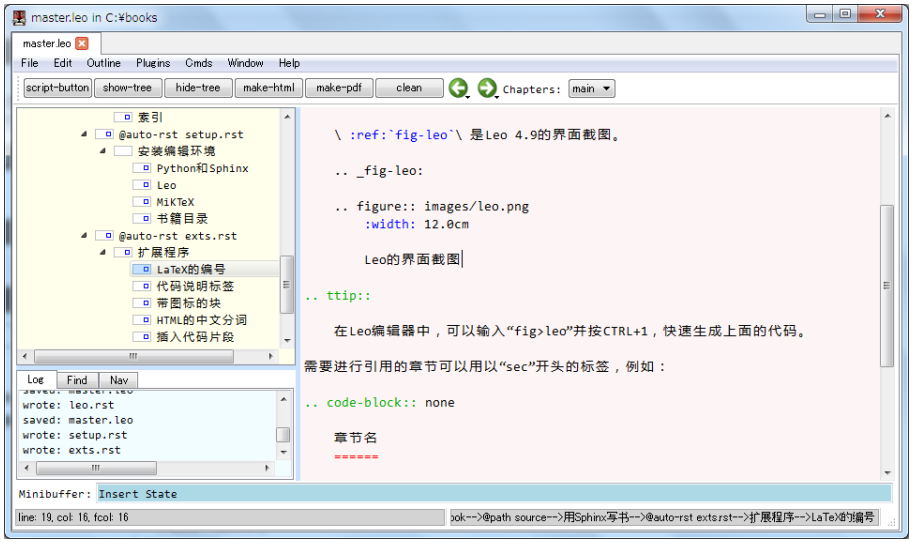

图 3.1 - Leo 的界面截图

3.1 按钮工具栏

打开“master.leo”之后，可以在窗口的上方看到如下图所示的按钮工具栏。

Table 3.1 - Leo 的按钮工具栏说明

按钮名	功能
script-button	Leo 自带的按钮，用它可以创建新的按钮
show-tree	显示并调整提纲窗口的宽度
hide-tree	隐藏提纲窗口
make-html	将当前的书籍项目编译为 HTML
make-pdf	将当前的书籍项目编译为 PDF
clean	清除当前的书籍项目的编译结果

其中，make-html、make-pdf 以及 clean 等三个按钮，需要提纲栏中的当前节点为某个书籍项目的子节点。

3.2 快速输入宏

在“master.leo”中定义了可快速输入各种命令的宏，其节点路径为：

```
@chapters-->Scripts-->@command rst-macro
```

输入宏之后按 CTRL+1 即可执行，将其扩展为对应的文本。下表列出了一些常用的宏：

Table 3.2 - 快速输入文本的宏

输入	输出
table	table 命令
inc>	literalinclude 命令
math	math 命令
fig>	figure 命令
_s	章节标签
_f	图表标签
sec	章节参照
fig	图表参照
m	行内 math 命令
tl	tlink 命令
tt	ttip 命令
tw	twaring 命令
tc	tcode 命令
ta	tanim 命令
cb	code-block 命令
t	topic 命令
l	超链接
数字	对应的符号，如 ❶
->	→

其中带“>”的宏可以输入参数，例如“fig>leo.png”、“inc>example.py>1”等。

第 4 章

索引

- genindex
- search