

采用 tcolorbox 宏包设计的用于排版终端窗口和代码的宏包 boxie.sty

耿楠

西北农林科技大学信息工程学院，陕西·杨凌，712100

2018 年 8 月 28 日

摘要

在使用<https://github.com/latexstudio/ChenLaTeXBookTemplate>提供的 L^AT_EX 书籍模板排版时，发现该模板中设计代码盒子环境和命令非常实用，另外，该模板还提供了排版终端命令窗口的基本思路。基于此，通过研读该模板的代码，重新设计了用于个 Ubuntu、Windows 和 Mac 终端命令窗口的 12 个排版环境和 12 个从文件读取窗口内容的排版命令，以期更为方便的排版终端命令窗口。

该宏包可以为经常有编写终端命令窗口和代码排版的人员提供帮助，但由于作者水平有限，一定存在不足之处，欢迎大家多提宝贵意见和建议。

一、终端窗口盒子环境与命令

1、终端窗口分类

- 带底部说明的黑底白字 (用于屏幕阅读)
- 带底部说明的白底黑字 (用于打印输出)
- 无底部说明的黑底白字 (用于屏幕阅读)
- 无底部说明的白底黑字 (用于打印输出)

2、环境与命令命名规范

- 操作系统前缀
 - ubt: Ubuntu
 - win: Windows

- mac: Mac OS
- 配色中缀
 - dark: 黑底白字
 - light: 白底黑字
- 底部说明中缀
 - c: 有底部说明
 - 空: 无底部说明
- 文件后缀
 - file: 窗口内容来自文件
 - 空: 窗口内容在环境中

3、环境与命令列表

共计 12 个环境，分别是：

- Ubuntu
 - “ubtdarke”: 带底部说明的黑底白字
 - “ubtlighc”: 带底部说明的白底黑字
 - “ubtdark”: 无底部说明的黑底白字
 - “ubtlight”: 无底部说明的白底黑字
- Windows
 - “windarke”: 带底部说明的黑底白字
 - “winlighc”: 带底部说明的白底黑字
 - “windark”: 无底部说明的黑底白字
 - “winlight”: 无底部说明的白底黑字
- Mac
 - “macdarke”: 带底部说明的黑底白字
 - “maclighc”: 带底部说明的白底黑字

- “macdark”: 无底部说明的黑底白字
- “maclight”: 无底部说明的白底黑字

共计 12 个命令，分别是：

- Ubuntu

- “\ubtdarkfile”: 带底部说明的黑底白字，内容来自文件
- “\ubtlightfile”: 带底部说明的白底黑字，内容来自文件
- “\ubtdarkfile”: 无底部说明的黑底白字，内容来自文件
- “\ubtlightfile”: 无底部说明的白底黑字，内容来自文件

- Windows

- “\windarkfile”: 带底部说明的黑底白字，内容来自文件
- “\winlightfile”: 带底部说明的白底黑字，内容来自文件
- “\windarkfile”: 无底部说明的黑底白字，内容来自文件
- “\winlightfile”: 无底部说明的白底黑字，内容来自文件

- Mac

- “\macdarkfile”: 带底部说明的黑底白字，内容来自文件
- “\maclightfile”: 带底部说明的白底黑字，内容来自文件
- “\macdarkfile”: 无底部说明的黑底白字，内容来自文件
- “\maclightfile”: 无底部说明的白底黑字，内容来自文件

4、基本使用语法

以黑底白字 Ubuntu 终端窗口排版为例，其排版语法为：

1.4.1 环境

带底部说明的黑底白字环境

```
1 \begin{ubtdarkc}{底部说明}{标题}  
2 窗口内容  
3 \end{ubtdarkc}
```

无底部说明的黑底白字环境

```
1 \begin{ubtdark}{标题}  
2   窗口内容  
3 \end{ubtdark}
```

1.4.2 命令

带底部说明的黑底白字环境

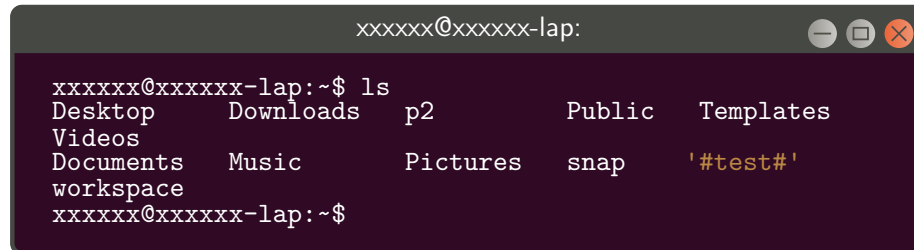
```
1 \ubtdarkcfile{底部说明}{标题}{窗口内容文件名}
```

无底部说明的黑底白字环境

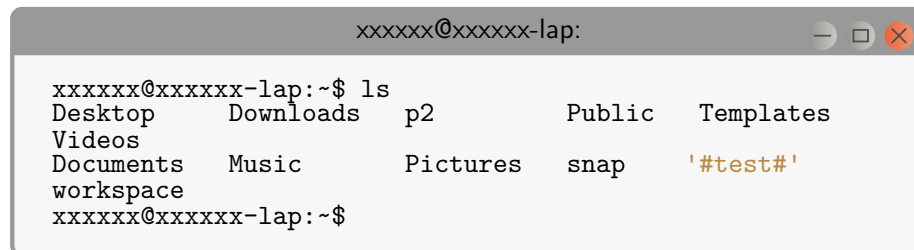
```
1 \ubtdarkfile{标题}{窗口内容文件名}
```

5、排版示例

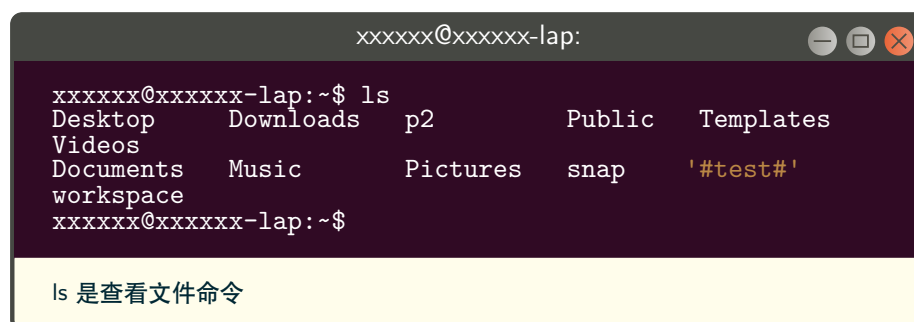
6、Ubuntu 终端窗口



```
xxxxxx@xxxxxx-lap:  
xxxxxx@xxxxxx-lap:~$ ls  
Desktop    Downloads  p2         Public    Templates  
Videos  
Documents  Music      Pictures   snap      '#test#'  
workspace  
xxxxxx@xxxxxx-lap:~$
```

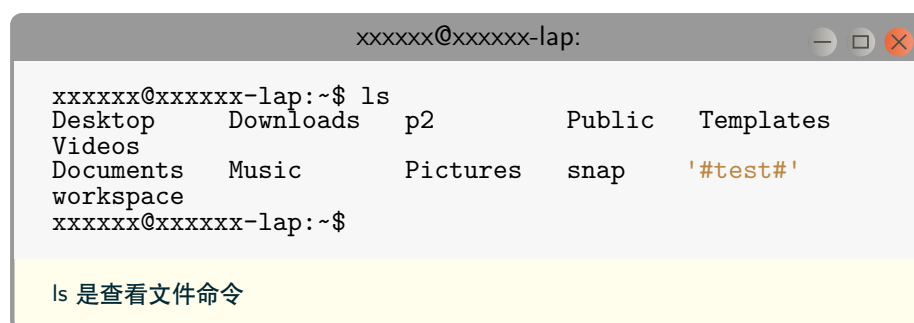


```
xxxxxx@xxxxxx-lap:  
xxxxxx@xxxxxx-lap:~$ ls  
Desktop    Downloads  p2         Public    Templates  
Videos  
Documents  Music      Pictures   snap      '#test#'  
workspace  
xxxxxx@xxxxxx-lap:~$
```



```
xxxxxx@xxxxxx-lap:~$ ls
Desktop    Downloads  p2         Public    Templates
Videos
Documents  Music      Pictures   snap      '#test#'
workspace
xxxxxx@xxxxxx-lap:~$
```

ls 是查看文件命令



```
xxxxxx@xxxxxx-lap:~$ ls
Desktop    Downloads  p2         Public    Templates
Videos
Documents  Music      Pictures   snap      '#test#'
workspace
xxxxxx@xxxxxx-lap:~$
```

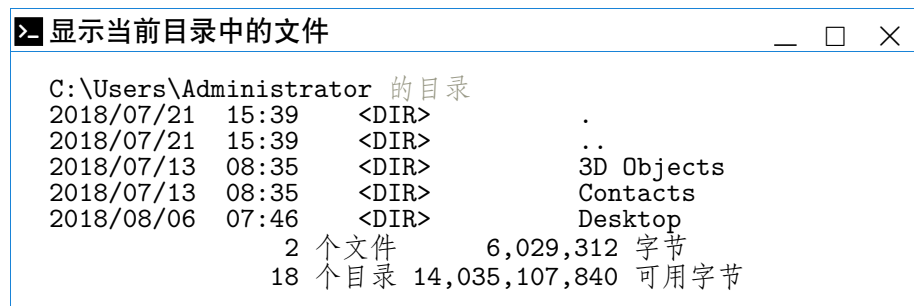
ls 是查看文件命令

7、Windows 命令行窗口



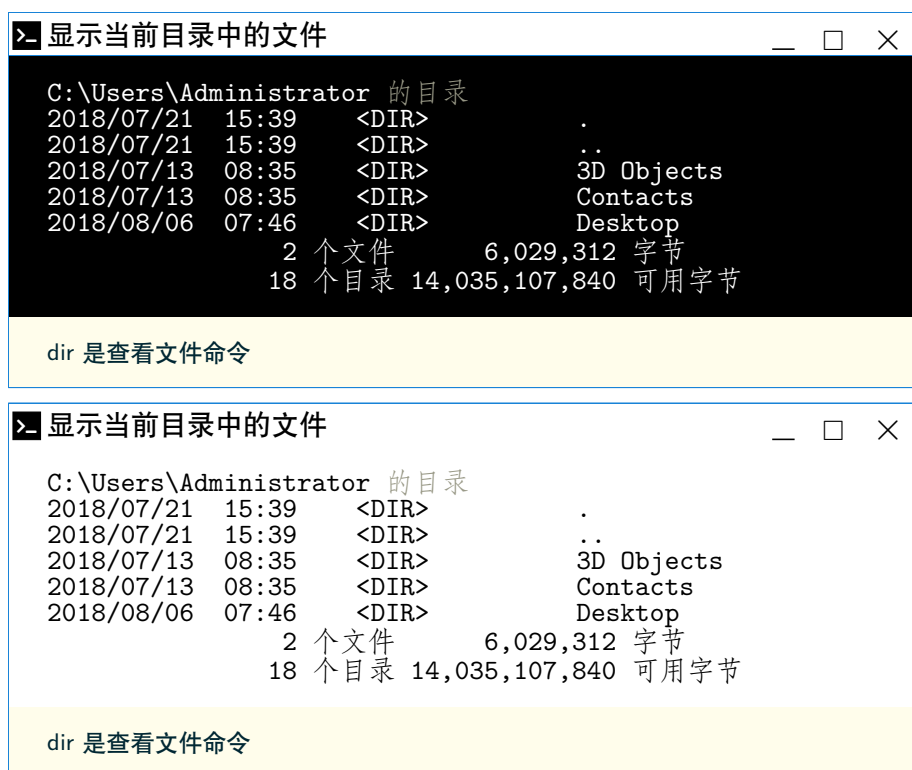
```
> 显示当前目录中的文件

C:\Users\Administrator 的目录
2018/07/21  15:39    <DIR>          .
2018/07/21  15:39    <DIR>          ..
2018/07/13   08:35    <DIR>          3D Objects
2018/07/13   08:35    <DIR>          Contacts
2018/08/06   07:46    <DIR>          Desktop
                2 个文件          6,029,312 字节
                18 个目录 14,035,107,840 可用字节
```

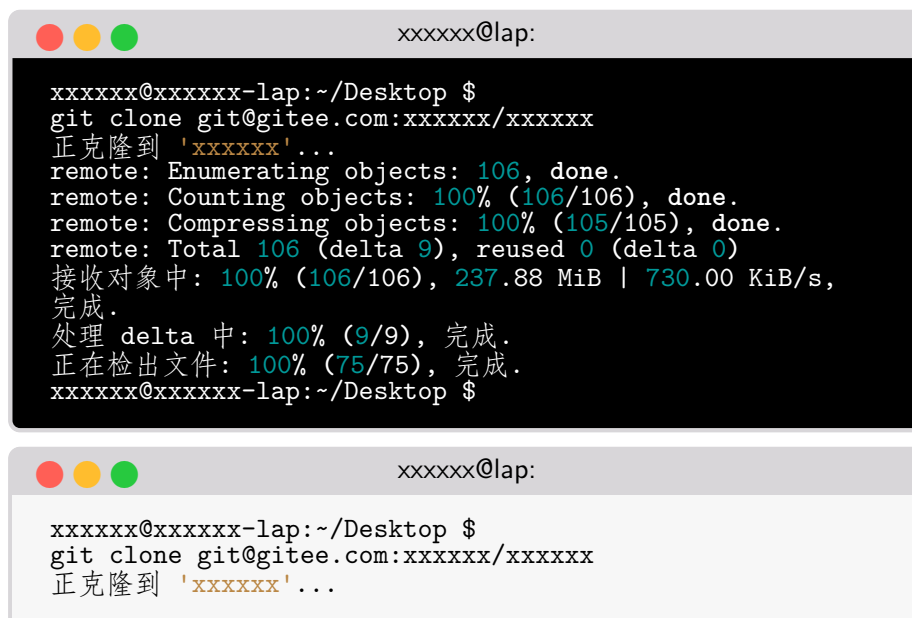


```
> 显示当前目录中的文件

C:\Users\Administrator 的目录
2018/07/21  15:39    <DIR>          .
2018/07/21  15:39    <DIR>          ..
2018/07/13   08:35    <DIR>          3D Objects
2018/07/13   08:35    <DIR>          Contacts
2018/08/06   07:46    <DIR>          Desktop
                2 个文件          6,029,312 字节
                18 个目录 14,035,107,840 可用字节
```



8、Mac 终端窗口

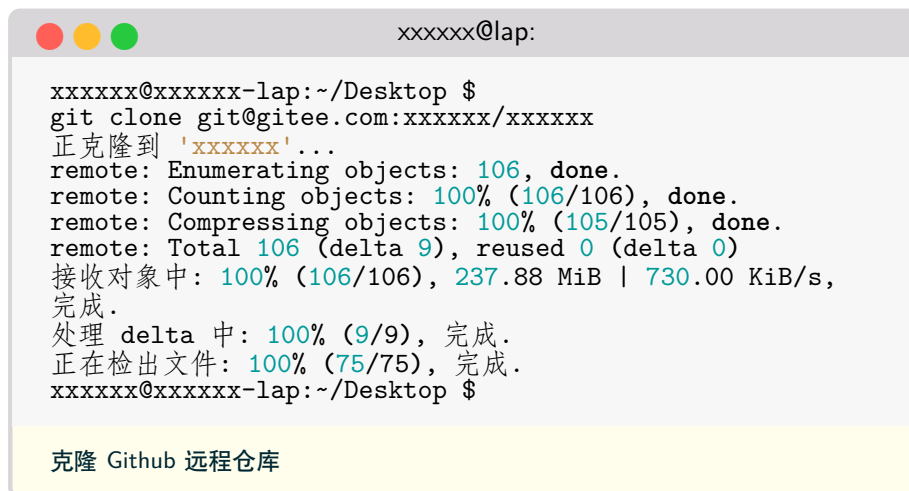


```
remote: Enumerating objects: 106, done.
remote: Counting objects: 100% (106/106), done.
remote: Compressing objects: 100% (105/105), done.
remote: Total 106 (delta 9), reused 0 (delta 0)
接收对象中: 100% (106/106), 237.88 MiB | 730.00 KiB/s,
完成.
处理 delta 中: 100% (9/9), 完成.
正在检出文件: 100% (75/75), 完成.
xxxxxx@xxxxxx-lap: ~/Desktop $
```



A terminal window titled 'xxxxxx@lap:' with a dark background. It shows the execution of 'git clone git@gitee.com:xxxxxx/xxxxxx'. The output includes progress bars for object enumeration, counting, and compression, followed by a summary of the total objects, delta, and reused objects. It also shows the progress of receiving objects and processing the delta. The prompt is 'xxxxxx@xxxxxx-lap: ~/Desktop \$'.

克隆 Github 远程仓库



A terminal window titled 'xxxxxx@lap:' with a light background. It shows the execution of 'git clone git@gitee.com:xxxxxx/xxxxxx'. The output includes progress bars for object enumeration, counting, and compression, followed by a summary of the total objects, delta, and reused objects. It also shows the progress of receiving objects and processing the delta. The prompt is 'xxxxxx@xxxxxx-lap: ~/Desktop \$'.

克隆 Github 远程仓库

9、通用环境与命令

定义了 2 个通用 Ubuntu 黑底白字样式的终端窗口环境，可以通过可选参数指定窗口内容的语言，其基本语法是

有底部说明

```
1 \begin{GitExample}[代码语言]{底部说明}{标题}  
2 ...  
3 \end{GitExample}
```

和

无底部说明

```
1 \begin{GitExempla}[代码语言]{标题}  
2 ...  
3 \end{GitExempla}
```

及

无底部说明

```
1 \gitfile[代码语言]{标题}{文件名}
```

二、代码排版环境与命令

1、代码排版样式分类

- 带底部说明无交叉引用
- 无底部说明无交叉引用
- 无底部说明有交叉引用

2、环境与命令列表

2.2.1 环境

- “langPyTwo”：带底部说明无交叉引用代码排版环境
- “langPyOne”：无底部说明无交叉引用代码排版环境
- “langCVOne”：无底部说明有交叉引用代码排版环境

2.2.2 命令

- “\langPyfile”: 无底部说明无交叉引用代码排版命令，代码带自文件
- “\langCVfile”: 无底部说明有交叉引用代码排版命令，代码带自文件

3、基本语法

有底部说明无交叉引用代码排版环境

```
1 \begin{langPyTwo}[语言]{底部说明}{标题}  
2 ...  
3 \end{langPyTwo}
```

无底部说明无交叉引用代码排版环境

```
1 \begin{langPyOne}[语言]{标题}  
2 ...  
3 \end{langPyOne}
```

无底部说明有交叉引用代码排版环境

```
1 \begin{langCVOne}[语言][交叉引用标签][显示语言名]{标题}  
2 ...  
3 \end{langCVOne}
```

排版命令使用

```
1 \langCVfile[语言][交叉引用标签][语言名显示]{标题}{文件名}  
2 \langPyfile[语言]{标题}{文件名}
```

其中，方括号的参数是可选的。“[语言名显示]”是显示在标题中要排版代码的语言名称。

4、代码排版实例

C 语言代码排版 (langPyTwo 环境)

```
1 #include<stdio.h>
2 #include<stdlib.h>
3
4 int main()
5 {
6     printf("Hello World!\n");
7     return 0;
8 }
```

这是 C 语言代码排版一个实例

Java 语言代码排版 (langPyOne 环境)

```
1 public class HelloWorld {
2     public static void main(String[] args){
3         System.out.println("Hello World!");
4     }
5 }
```

C 语言代码排版 (langPyfile 命令)

```
1 #include<stdio.h>
2 #include<stdlib.h>
3
4 int main()
5 {
6     printf("Hello World!\n");
7     return 0;
8 }
```

用“langCVOne”环境可以实现交叉引用，如代码1所示。

```
程序清单 1: Python(langCVOne 环境) Python
1 # -*- coding: UTF-8 -*-
2
3 # Filename : helloworld.py
4 # 该实例输出 Hello World!
5
6 print('Hello World!')
```

用“\langCVfile”命令可以实现交叉引用，如代码2所示。

```
程序清单 2: Matlab(langCVfile 命令) Matlab
1 function HelloWorld()
2
3 % 输出 Hello, World!
4
5 % Detailed explanation goes here
6
7 disp('Hello,World!');
8
9 end
```

三、 注意事项

1、 字体

本宏包需要使用 fontawesome5 图标字体支持,请在 <https://fontawesome.com/> 下载安装。

2、 代码排版引擎

本宏包建议使用“minted”宏包实现代码的排版，用“xelatex -shell-escape main.tex”编译 tex 文件，但如果没有安装 minted 需要的 python 及其 pygments 模块，请提前安装该模块。

若在编译是不使用 “-shell-escape” 参数，则会自动切换到用 listings 排版代码，注意有部分代码名称与 pygments 定义不一致，请自行查阅相关手册。