



# 北京理工大学本科生 L<sup>A</sup>T<sub>E</sub>X 模板使用手册

[spencerwooo/BIThesis](#)

主编：北京理工大学 2016 级计算机学院 武上博 王赞

二〇二〇年三月十日 BIT<sub>HESIS</sub> 版本 v0.0.2

## 简介

BIT<sub>HESIS</sub> 北京理工大学本科生 L<sup>A</sup>T<sub>E</sub>X 模板是北京理工大学本科生毕业设计开题报告、总论文，以及其他课程报告、实验报告等重要论文、报告的 L<sup>A</sup>T<sub>E</sub>X 模板集合。如果你厌烦了 Word 格式的不专业、参考文献的难以管理、公式输入的差劲体验……那么欢迎来尝试用专业的学术稿件排版利器——L<sup>A</sup>T<sub>E</sub>X，来排版你的论文。专业高端、学界认可、开源免费，L<sup>A</sup>T<sub>E</sub>X 是你论文排版的最佳搭档。

BIT<sub>HESIS</sub> 北京理工大学本科生 L<sup>A</sup>T<sub>E</sub>X 模板目前支持使用 X<sub>g</sub>L<sup>A</sup>T<sub>E</sub>X 进行编译，使用以 biber 为后端的 BibLaTeX 进行参考文献的生成，符合《信息与文献参考文献著录规则》（GB/T 7714—2015）的标准。目前主要设计完成了计算机学院本科生毕业论文开题报告、毕业设计毕业论文与通用实验报告的 L<sup>A</sup>T<sub>E</sub>X 模板。

## 目录

第 1 章 如何开始	4
1.1 在线说明文档：Wiki	4
1.2 准备工作	4
1.3 下载合适的 L <sup>A</sup> T <sub>E</sub> X 发行版	5
1.3.1 Windows 和 Linux 系统	5
1.3.2 macOS 系统	6
1.3.3 确认安装	6
1.4 挑选合适的 L <sup>A</sup> T <sub>E</sub> X 编辑器	7
1.4.1 使用 VS Code 配合 L <sup>A</sup> T <sub>E</sub> X Workshop 插件编辑 L <sup>A</sup> T <sub>E</sub> X 文档	7

目录	2
1.4.2 使用 T <sub>E</sub> Xstudio 编辑 L <sup>A</sup> T <sub>E</sub> X 文档	8
<b>第 2 章 下载与使用模板</b>	<b>10</b>
2.1 熟悉简单 L <sup>A</sup> T <sub>E</sub> X 语法	10
2.2 在项目的 Release 页面下载你希望使用的模板	10
2.3 编译模板	11
2.3.1 徒手编译	12
2.3.2 使用 VS Code 撰写与编译 L <sup>A</sup> T <sub>E</sub> X 模板	13
2.3.3 使用 T <sub>E</sub> Xstudio 撰写与编译 L <sup>A</sup> T <sub>E</sub> X 模板	15
<b>第 3 章 计算机学院本科生开题报告使用指南</b>	<b>17</b>
3.1 熟悉项目	17
3.2 你的内容从哪里开始?	18
3.3 其他注意事项	19
3.3.1 插入图片	19
3.3.2 插入表格	19
<b>第 4 章 北京理工大学本科生毕业设计论文模板使用指南</b>	<b>21</b>
4.1 熟悉项目	21
4.2 你的内容从哪里开始?	23
4.2.1 开始	23
4.2.2 中英摘要	24
4.2.3 正文	24
4.2.4 后续模块	24
4.3 参考文献管理	25
4.4 图片素材	26
4.5 表格插入	27
4.6 公式插入	27
4.7 其他	28
4.7.1 代码高亮	29
4.7.2 算法模块	30
<b>第 5 章 通用北京理工大学本科生实验报告模板使用指南</b>	<b>31</b>
5.1 熟悉项目	31
5.2 编译方式与使用	31
<b>第 6 章 如何将 L<sup>A</sup>T<sub>E</sub>X 文档转换为 Word</b>	<b>33</b>
6.1 安装 Pandoc 命令行工具	33
6.2 完善 Word 格式的模板文件	34

6.3 进行格式转换 . . . . .	34
6.3.1 朴素格式转换 . . . . .	35
6.3.2 含有目标模板 Word 文档的格式转换 . . . . .	35
6.3.3 含有参考文献文档的格式转换 . . . . .	35
<b>第 7 章 疑难杂症</b>	<b>37</b>
7.1 出现字体缺失的编译失败提示 . . . . .	37
7.2 出现参考文献样式找不到的编译失败提示 . . . . .	38
7.3 无法使用代码高亮 minted 宏包 . . . . .	39
7.3.1 排查是否正确安装 Python 与 pygments 包 . . . . .	39
7.3.2 添加额外的编译参数 . . . . .	40
7.3.3 删除 minted 宏包的缓存文件夹 . . . . .	41
7.4 编译过慢，一次更改需要编译半分钟 . . . . .	42
<b>第 8 章 致谢</b>	<b>43</b>

## 第 1 章 如何开始

BIT<sub>HESIS</sub> 为各位在北京理工大学就读的本科同学提供了基于北京理工大学计算机学院教务部给出的“北京理工大学计算机学院本科生毕业论文：开题报告”与北京理工大学教务部提供的“北京理工大学本科生毕业设计：论文模板（目前是 2019 届版本）”的 L<sup>A</sup>T<sub>E</sub>X 样版。借助于 BIT<sub>HESIS</sub> 的 L<sup>A</sup>T<sub>E</sub>X 模板，你可以在保证论文格式整齐、完美、符合要求的前提下，专注于学术研究、项目实施，从而顺利完成你的学术项目。

本“使用手册”希望为大家全面地介绍 L<sup>A</sup>T<sub>E</sub>X 环境的搭建方法、BIT<sub>HESIS</sub> 的使用方法，从而快速掌握使用 L<sup>A</sup>T<sub>E</sub>X 排版引擎进行基本的论文撰写的方法，完成符合学校要求的学位论文。BIT<sub>HESIS</sub> 目前使用 GitHub 进行维护，官方项目地址位于：

<https://github.com/spencerwooo/BIThesis>

### 1.1 在线说明文档：Wiki

和本手册的目标类似，BIT<sub>HESIS</sub> 项目同样维护了一个在线版本的说明文档，位于：BIThesis - wiki，二者的目的、内容、功能类似，且会随着模板的开发与维护同步更新。

BIT<sub>HESIS</sub> 在线说明文档目前拥有与本手册一致的如下模块：

1. 主页：Home
2. 如何开始：First things first
3. 使用其中一个模板：Using one of the templates
4. 本科生开题报告：Proposal report
5. 本科生毕业论文：Graduation thesis
6. 本科生实验报告：Lab report
7. 将 LaTeX 文档转换为 Word：Converting to Word

接下来，我们正式开始介绍 L<sup>A</sup>T<sub>E</sub>X 与 BIT<sub>HESIS</sub> 的使用方法。

### 1.2 准备工作

首先，在使用模板之前，你需要在本机安装 L<sup>A</sup>T<sub>E</sub>X 环境。一个完整的 L<sup>A</sup>T<sub>E</sub>X 环境包括：

- 开源免费的 L<sup>A</sup>T<sub>E</sub>X 发行版（包含有必备的 L<sup>A</sup>T<sub>E</sub>X 编译器与有用的宏包）
- 以及一个得心应手的 L<sup>A</sup>T<sub>E</sub>X 编辑器

我们在 Windows、macOS 与 Linux 环境中均可以使用 L<sup>A</sup>T<sub>E</sub>X 进行文档撰写。按照操作系统的不同，我们分别进行介绍。

### 1.3 下载合适的 L<sup>A</sup>T<sub>E</sub>X 发行版



**注意：**BIT<sub>H</sub>ES<sub>IS</sub> 中参考文献为了和校方规定的模板格式《信息与文献参考文献著录规则》（GB/T 7714—2015）保持一致，使用了仅支持 T<sub>E</sub>XLive 2019 版本的宏包，如果你曾经安装过 T<sub>E</sub>XLive 且目前正在使用的 T<sub>E</sub>XLive 版本不是 2019 版本，那么请及时更新为最新的 T<sub>E</sub>XLive 2019 版本。

#### 1.3.1 Windows 和 Linux 系统

对于 Windows 和 Linux 系统，我们可以直接下载使用 T<sub>E</sub>XLive 发行版。

**在线安装** 官方的安装指南位于：Installing T<sub>E</sub>XLive over the Internet。使用这一方法会下载 install-tl-windows.exe（Windows）或 install-tl-unx.tar.gz（Linux），之后运行相应的可执行程序，安装程序即可将整个 T<sub>E</sub>XLive 发行版下载安装到我们本机。（通常会安装 3GB 左右的程序。）

#### Index of /CTAN/systems/texlive/Images

Name	Last modified	Size	Description
<a href="#">Parent Directory</a>		-	
<a href="#">README.md</a>	30-Apr-2019 04:28	1.1K	
<a href="#">texlive.iso</a>	10-Apr-2019 23:59	3.3G	
<a href="#">texlive2019-20190410.iso</a>	10-Apr-2019 23:59	3.3G	
<a href="#">texlive2019-20190410.iso.md5</a>	10-Apr-2019 23:59	59	
<a href="#">texlive2019-20190410.iso.sha512</a>	10-Apr-2019 23:59	155	
<a href="#">texlive2019-20190410.iso.sha512.asc</a>	10-Apr-2019 23:59	455	
<a href="#">texlive2019.iso</a>	10-Apr-2019 23:59	3.3G	
<a href="#">texlive2019.iso.md5</a>	10-Apr-2019 23:59	50	
<a href="#">texlive2019.iso.sha512</a>	10-Apr-2019 23:59	146	
<a href="#">texlive2019.iso.sha512.asc</a>	10-Apr-2019 23:59	455	

最新版本的 TeX Live 镜像

图 1-1 北京理工大学开源镜像站 T<sub>E</sub>XLive 下载

**离线安装** 使用北京理工大学校园网的同学也可以直接使用我校官方 T<sub>E</sub>XLive 镜像进行安装。我校 T<sub>E</sub>XLive 镜像资源位于 /CTAN/systems/texlive/Images，其

中我们选择下载 `texlive2019.iso` 即可，如图 1-1 所示。Windows 10 可直接挂载 ISO 镜像（双击即可），其余系统用合适的软件也可。之后在打开的文件夹中点击执行 `install-tl-windows`（Windows）或 `install-tl`（Linux）即可离线安装全部  $\text{\TeX}$ Live 组件。

**使用包管理工具进行安装** 使用 Linux 系统的同学也可以选择使用合适的包管理工具进行  $\text{\TeX}$ Live 的安装。以 Ubuntu 为例子，只需要运行下面命令，即可下载安装整个  $\text{\TeX}$ Live 发行版。

```
sudo apt install texlive
```

### 1.3.2 macOS 系统

对于 macOS 系统，我们可以直接下载使用  $\text{MacTeX}$  发行版。 $\text{MacTeX}$  发行版是以 `pkg` 文件进行发布安装的，我们进入  $\text{MacTeX}$  的下载页面，点击下载 `MacTeX.pkg` 即可下载完整的  $\text{MacTeX}$  安装包（大约 3.9GB）。之后双击运行即可安装。

另外，使用 Homebrew 包管理的同学，也可以通过 Homebrew Cask 直接安装  $\text{MacTeX}$ ：

```
# 加载 Homebrew Cask
brew tap caskroom/cask

# 利用 Cask 安装 MacTeX
brew cask install mactex
```

### 1.3.3 确认安装

为了保证我们  $\text{\LaTeX}$  发行版的安装没有问题，我们需要验证一下  $\text{\LaTeX}$  编译工具的安裝情况。我们打开终端（Windows 打开 PowerShell、macOS 打开 Terminal、Linux 打开你所使用的终端模拟器），在其中输入下面的命令：

- 验证 `xelatex`  $\text{\LaTeX}$  编译器的安装情况：

```
xelatex --version
```

```
Welcome to PowerShell, the powerful shell for Windows

~ > xelatex --version
XeTeX 3.14159265-2.6-0.999991 (TeX Live 2019/W32TeX)
kpathsea version 6.3.1
Copyright 2019 SIL International, Jonathan Kew and Khaled Hosny.
There is NO warranty. Redistribution of this software is
covered by the terms of both the XeTeX copyright and
the Lesser GNU General Public License.
For more information about these matters, see the file
named COPYING and the XeTeX source.
Primary author of XeTeX: Jonathan Kew.
Compiled with ICU version 63.1; using 63.1
Compiled with zlib version 1.2.11; using 1.2.11
Compiled with FreeType2 version 2.9.1; using 2.9.1
Compiled with Graphite2 version 1.3.13; using 1.3.13
Compiled with HarfBuzz version 2.3.1; using 2.3.1
Compiled with libpng version 1.6.36; using 1.6.36
Compiled with poppler version 0.68.0
Compiled with fontconfig version 2.13.1; using 2.13.1
```

图 1-2 Xe<sub>La</sub>TeX 安装成功输出

- 验证 biber 参考文献编译器的安装情况:

```
biber --version
```

```
~ > biber --version
biber version: 2.12

~ >
```

图 1-3 biber 安装成功输出

出现如图 1-2 与 1-3 类似的输出，说明我们编译器安装应该是没有问题的。

## 1.4 挑选合适的 L<sup>A</sup>T<sub>E</sub>X 编辑器

理论上来说，任何一个“文本编辑器”均可以用来撰写 L<sup>A</sup>T<sub>E</sub>X 文档，但是一个得心应手的 L<sup>A</sup>T<sub>E</sub>X 编辑器一定会让我们撰写论文的效率大增。

### 1.4.1 使用 VS Code 配合 L<sup>A</sup>T<sub>E</sub>X Workshop 插件编辑 L<sup>A</sup>T<sub>E</sub>X 文档

VS Code 是微软开发的基于 Electron 跨平台技术的新晋代码编辑器，开源免费、拓展性强、功能强大，是当代开发者的首选。用 VS Code 配合 L<sup>A</sup>T<sub>E</sub>X Workshop 插件我们可以打造一个强大的 L<sup>A</sup>T<sub>E</sub>X 编辑器。

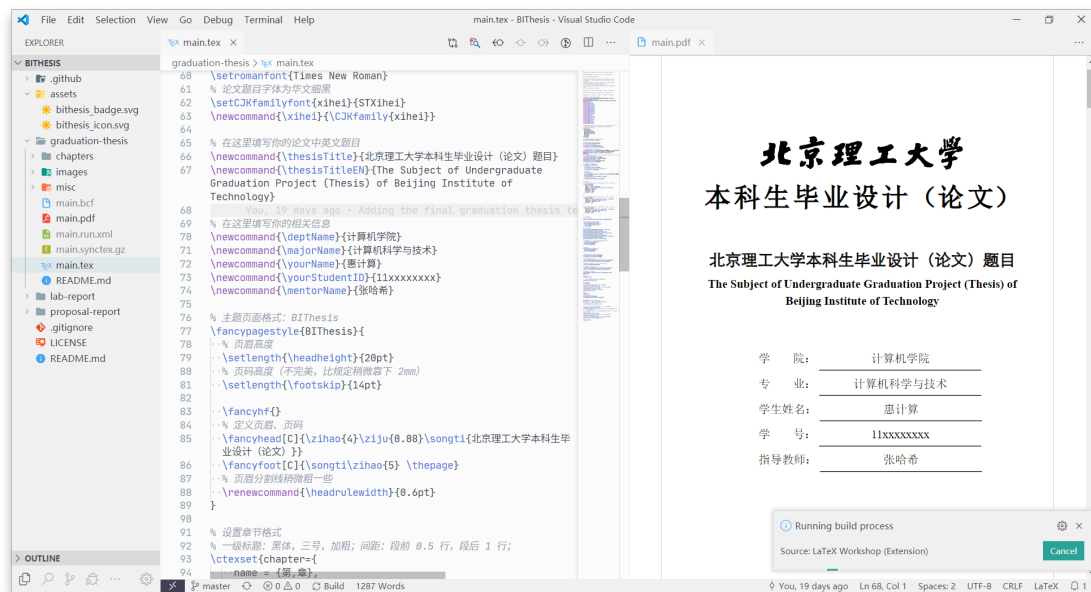


图 1-4 VS Code 代码编辑器

- 安装 VS Code 编辑器：Visual Studio Code - Code editing. Redefined.
- 安装插件：
  - 安装  $\text{\LaTeX}$  Workshop 插件：Visual Studio Code  $\text{\LaTeX}$  Workshop Extension
    - \* 提供基本的浏览、编辑、自动补全、自动格式化  $\text{\LaTeX}$  文档的功能
    - \* 提供在 VS Code 内直接预览  $\text{\LaTeX}$  文档编译得到的 PDF 的功能
    - \* 提供编译工具链、自定义编译方法等功能提供 SyncTeX 双向定位功能（ $\text{\LaTeX}$  源码  $\longleftrightarrow$  PDF）
  - （可选）安装  $\text{\LaTeX}$  Utilities 插件：Visual Studio Code  $\text{\LaTeX}$  Utilities
    - \* 提供实时  $\text{\LaTeX}$  文档字数统计的功能
    - \* 提供与参考文献管理工具 Zotero 连接的功能

使用 VS Code 作为  $\text{\LaTeX}$  编辑器时，我们需要特别配置编译工具 `tools` 与编译工具链 `recipes`。对于包含有目录、参考文献、图片与表格引用的  $\text{\LaTeX}$  文档，我们往往需要使用多个编译工具串联编译。具体的 VS Code 编译方法，请继续阅读下一章《第 2 章：下载与使用模板》。

### 1.4.2 使用 $\text{\TeX}$ studio 编辑 $\text{\LaTeX}$ 文档



注意：使用 macOS 的同学，请谨慎使用  $\text{\TeX}$ studio，据反映  $\text{\TeX}$ studio 有比较复杂难以解决的问题，因此请使用 Mac 的同学尽量使用 VS Code（或下文提到的付费 Texpad）来使用本模板。



TeXstudio 是老牌 L<sup>A</sup>T<sub>E</sub>X 编辑器，使用跨平台技术 Qt 编写而成。虽然界面相对老旧，但是依旧可靠。我们可以去 TeXstudio 的官网下载安装各个系统版本的 TeXstudio。

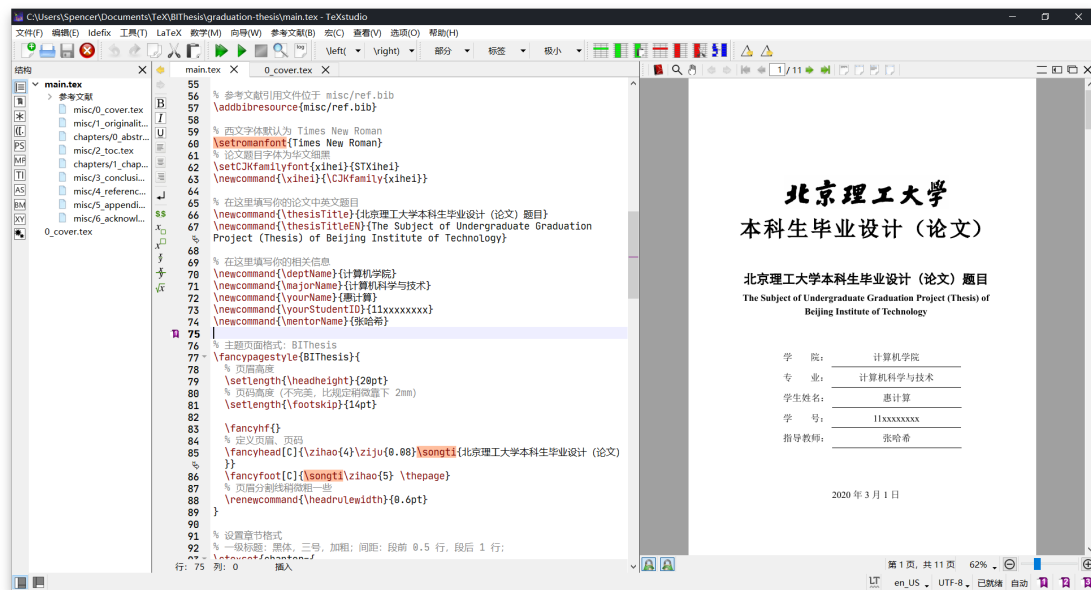


图 1-5 TeXstudio 老牌 L<sup>A</sup>T<sub>E</sub>X 编辑器

默认情况下 TeXstudio 的编译工具链均已经配置完毕，基本开箱即用。对于如何用 TeXstudio 编译本模板，请继续阅读下一章《第 2 章：下载与使用模板》。



另外，如果你是特别不差钱的 Mac 用户，希望用最好用最牛逼的 L<sup>A</sup>T<sub>E</sub>X 编辑器，你也可以去购买目前售价 \$29.99 美元（约合人民币 209.69 元）的 Texpad。使用 macOS、iOS 原生技术栈开发，Texpad 可能是目前使用体验最顺滑的 L<sup>A</sup>T<sub>E</sub>X 编辑器，另外由于 Texpad 使用私有 L<sup>A</sup>T<sub>E</sub>X 发行版，使得 Texpad 支持实时预览成果 PDF 与双向同步滚动支持。有这方面需要（与金钱）的同学可以考虑入手。

准备就绪后，我们就可以开始使用 BIT<sub>H</sub>ES<sub>IS</sub> 提供的模板进行 L<sup>A</sup>T<sub>E</sub>X 文档的撰写啦！

## 第 2 章 下载与使用模板

BITHESIS 整个项目中包含多个模板，每个模板各自位于独立的文件夹中。

### 2.1 熟悉简单 L<sup>A</sup>T<sub>E</sub>X 语法

如果你之前没有接触过 L<sup>A</sup>T<sub>E</sub>X，请前往 Overleaf 的“30 分钟学习 L<sup>A</sup>T<sub>E</sub>X”文档进行阅读，从而对 L<sup>A</sup>T<sub>E</sub>X 有大致的印象。

一些常用的 L<sup>A</sup>T<sub>E</sub>X 格式与使用技巧：

- L<sup>A</sup>T<sub>E</sub>X 章节设定：Sections and chapters
- L<sup>A</sup>T<sub>E</sub>X 段落格式：Paragraphs and new lines
- L<sup>A</sup>T<sub>E</sub>X 粗体、斜体与下划线：Bold, italics and underlining
- L<sup>A</sup>T<sub>E</sub>X 有序列表、无序列表：Lists
- L<sup>A</sup>T<sub>E</sub>X 插入图片：Inserting Images
- L<sup>A</sup>T<sub>E</sub>X 构建表格：Tables
- L<sup>A</sup>T<sub>E</sub>X 插入数学公式：Mathematical expressions
- L<sup>A</sup>T<sub>E</sub>X 插入代码与代码高亮：Code Highlighting with minted
- L<sup>A</sup>T<sub>E</sub>X 插入算法伪代码描述：Algorithms
- 使用 BibL<sup>A</sup>T<sub>E</sub>X 管理参考文献：Bibliography management in LaTeX

有关 L<sup>A</sup>T<sub>E</sub>X 使用的更多技巧，请直接前往 Overleaf 官方文档进行查看。准备就绪之后，你就可以前往下载 BITHESIS 模板啦。

### 2.2 在项目的 Release 页面下载你希望使用的模板

为了方便各位同学使用，项目按照 Release 发布的流程，将每个模板进行打包，并在每次发版后用 GitHub Release 进行模板分发。也就是，你可以直接前本项目的 GitHub Release 页面，直接下载你所希望使用的模板压缩包，并解压到本地进行使用。

你可以点击这个链接前往最新的 Release 版本进行模板下载：

<https://github.com/spencerwooo/BIThesis/releases/latest>

在 Release 页面，你会看到：

```
/
├── proposal-report.zip ... 本科生毕业设计开题报告模板压缩包
├── graduation-thesis.zip ... 本科生毕业设计毕业论文模板压缩包
└── lab-report.zip ... 本科生实验报告模板压缩包
```

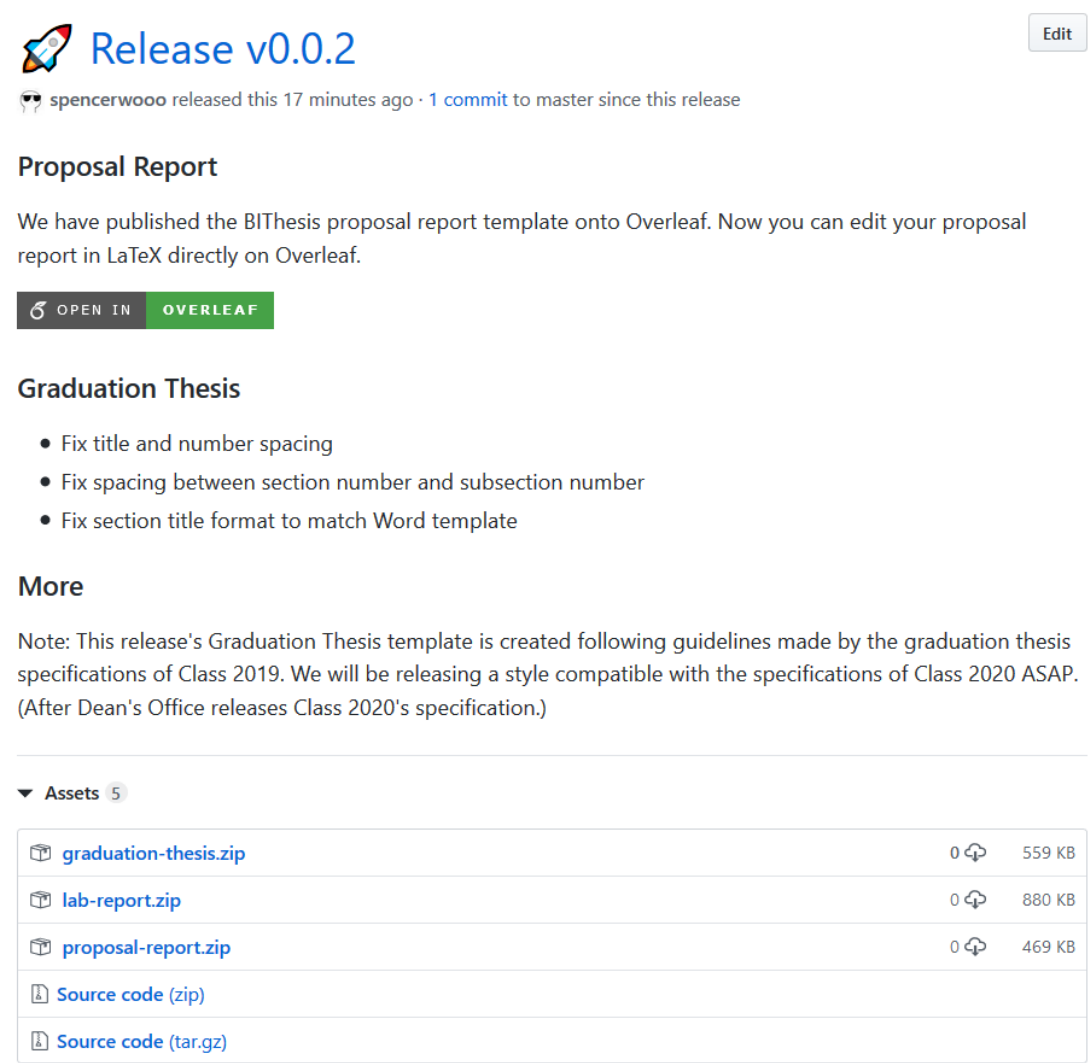


图 2-6 BIThesis 的 Release 页面

根据你的选择，下载其中你所要使用的模板即可。（当然，你也可以直接用 Git 将本项目完整克隆至本地，使用最新版本的模板。）

2.3 编译模板

与 Word 不同的是，L<sup>A</sup>T<sub>E</sub>X 模板需要我们用合适的工具进行编译，才能生成最终 PDF 文件。我们接下来介绍 BIThesis 中的模板在各个编辑器中的编译方法。

BIThesis 中的模板编译方式大同小异，我们都会使用 Xe<sup>L</sup>A<sub>T</sub>E<sub>X</sub>、biber 以及 latexmk 等工具来编译它们。编译 BIThesis 有两种方法：

- 1. 使用 xelatex 配合 biber 进行编译：需要使用“四步走”xelatex -> biber

-> `xelatex` -> `xelatex` 的编译顺序编译模板，全量编译，编译一次可能花费较长时间

2. 使用 `latexmk` 进行编译：只需要使用一次 `latexmk` 即可编译整个模板，自动识别参考文献编译器，增量编译

这两种编译方式均可以用于编译我们的模板，大家可以综合自己的使用习惯来挑选工具。事实上，后面我们将要介绍的  $\text{\LaTeX}$  编辑器，它们背后所使用的编译方法就是运行这里提到的两种编译工具。只是我们需要单独配置编辑器的编译方法，才能让编辑器正确的调用编译方式，编译我们的  $\text{\LaTeX}$  文档。

在这里，我挑选了三种常见的  $\text{\LaTeX}$  编写环境：

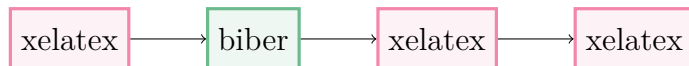
- 直接使用“命令行”徒手编写编译
- 使用 VS Code 配合  $\text{\LaTeX}$  Workshop 编写与编译
- 使用 `TeXstudio` 编写与编译

我会依次介绍在这三种环境下  $\text{\LaTeX}$  编译器配置方法。

### 2.3.1 徒手编译

当然，你完全可以不借助任何编辑器，直接使用“命令行”编译  $\text{\LaTeX}$  文档。

**使用  $\text{\XeLaTeX}$  编译** 如果你使用  $\text{\XeLaTeX}$  编译项目，那么你需要按照下面的顺序依次调用 `xelatex` 与 `biber` 命令行工具：



比如，编译主文档为 `main.tex` 的  $\text{\LaTeX}$  项目，我们具体的命令为：

```

1  # 第一步 xelatex
2  xelatex -no-pdf --interaction=nonstopmode main
3  # 第二步 biber
4  biber main
5  # 第三步 xelatex
6  xelatex -no-pdf --interaction=nonstopmode main
7  # 第四步 xelatex
8  xelatex --interaction=nonstopmode main
  
```

**使用 `latexmk` 编译** 如果你使用 `latexmk` 编译模板，那么你只需要使用如下的命令即可编译主文件为 `main.tex` 的  $\text{\LaTeX}$  项目：

```
1 # 只需要调用一次 latexmk 工具即可
2 latexmk -synctex=1 -interaction=nonstopmode -file-line-error
   ↪ -xelatex main.tex
```

### 2.3.2 使用 VS Code 撰写与编译 L<sup>A</sup>T<sub>E</sub>X 模板

VS Code 的设置项目可以通过快捷键 `ctrl or ⌘ + ,` 打开 UI 设置界面, 之后点击右上角 Open Settings (JSON) 按钮即可打开相应的 JSON 格式配置文件, 我们在这里即可定义 L<sup>A</sup>T<sub>E</sub>X 编译工具。其中:

- “编译工具”是在 "latex-workshop.latex.tools": [ ... ] 处进行定义, 即我们在这里定义每次调用工具 xelatex 或 latexmk 时所执行的命令
- “编译工具链”是在 "latex-workshop.latex.recipes": [ ... ] 处进行定义, 即我们在这里定义编译整个文档的工具链。对我们的模板第一种编译方式来说, 就是定义 xelatex -> biber -> xelatex -> xelatex “四步走”的串联过程

**使用 X<sub>q</sub>L<sup>A</sup>T<sub>E</sub>X 编译** 这种方法需要调用的工具有: xelatex 和 biber。我们在 VS Code 的设置中加入如下内容定义这两个工具:

```
1 "latex-workshop.latex.tools": [
2   {
3     "name": "xelatex",
4     "command": "xelatex",
5     "args": [
6       "-synctex=1",
7       "-interaction=nonstopmode",
8       "-file-line-error",
9       "-pdf",
10      "-outdir=%OUTDIR%",
11      "-cd",
12      "%DOC%"
13    ],
14    "env": {}
15  },
16  {
17    "name": "biber",
```

```
18     "command": "biber",
19     "args": [
20         "%DOCFILE%"
21     ],
22     "env": {}
23 }
24 ]
```

用这一方法编译整个文档的工具链串联方法是 `xelatex -> biber -> xelatex -> xelatex` “四步走”。我们在 VS Code 的设置中加入如下内容定义这个工具链：

```
1  "latex-workshop.latex.recipes": [
2      {
3          "name": "xelatex -> biber -> xelatex * 2",
4          "tools": [
5              "xelatex",
6              "biber",
7              "xelatex",
8              "xelatex"
9          ]
10     }
11 ]
```

**使用 latexmk 编译** 这种方法我们只需要使用 `latexmk` 这一个命令行工具。我们在 VS Code 的设置中添加如下的内容定义这一工具：

```
1  "latex-workshop.latex.tools": [
2      {
3          "name": "latexmk",
4          "command": "latexmk",
5          "args": [
6              "-synctex=1",
7              "-interaction=nonstopmode",
8              "-file-line-error",
9              "-xelatex",
10             "-outdir=%OUTDIR%",

```

```

11     "-cd",
12     "%DOC%"
13 ],
14     "env": {}
15 },
16 ]

```

之后我们再填入下面的内容定义整个工具链（只有一个 `latexmk`）：

```

1  "latex-workshop.latex.recipes": [
2    {
3      "name": "latexmk",
4      "tools": [
5        "latexmk"
6      ]
7    },
8  ]

```

之后，我们使用快捷键 `ctrl or ⌘ + ⌥ + P` 打开命令执行栏，并搜索“LaTeX Workshop: Build with recipe”，并选择你所用的 recipe（即上面配置的工具链），即可编译整个  $\text{\LaTeX}$  项目。

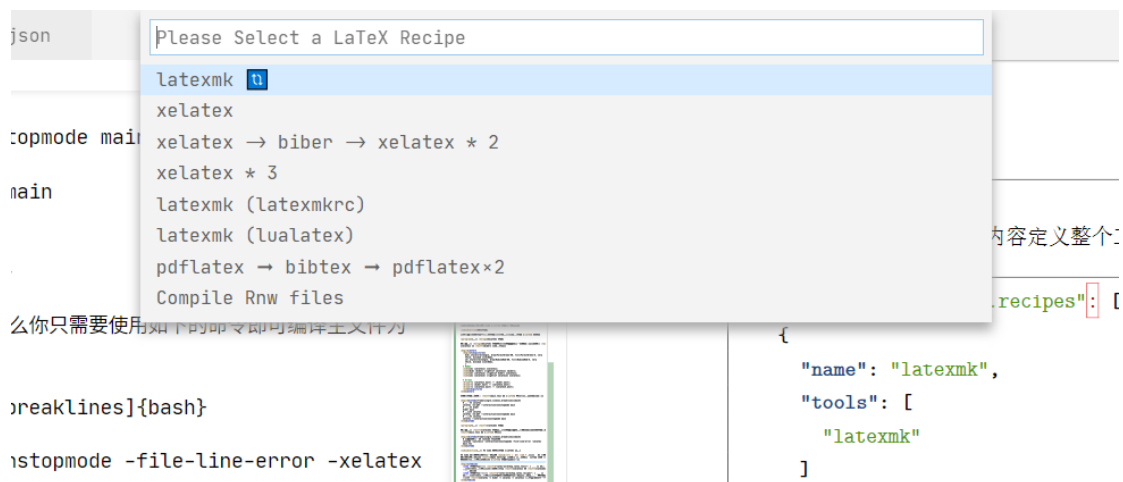


图 2-7 VS Code 挑选 recipe 工具链来编译  $\text{\LaTeX}$  项目

### 2.3.3 使用 $\text{\TeX}$ studio 撰写与编译 $\text{\LaTeX}$ 模板

$\text{\TeX}$ studio 的编译工具大部分已经为我们配置完毕，我们只需要在  $\text{\TeX}$ studio 的设置中定义编译所用的编译器即可。在  $\text{\TeX}$ studio 中点击“选项 » 设置  $\text{\TeX}$ s-

tudio”，在打开的窗口中选择“构建”，并在元命令里面将“默认编译器”设置为 xelatex 或 latexmk，将默认文献工具设置为 biber 即可。

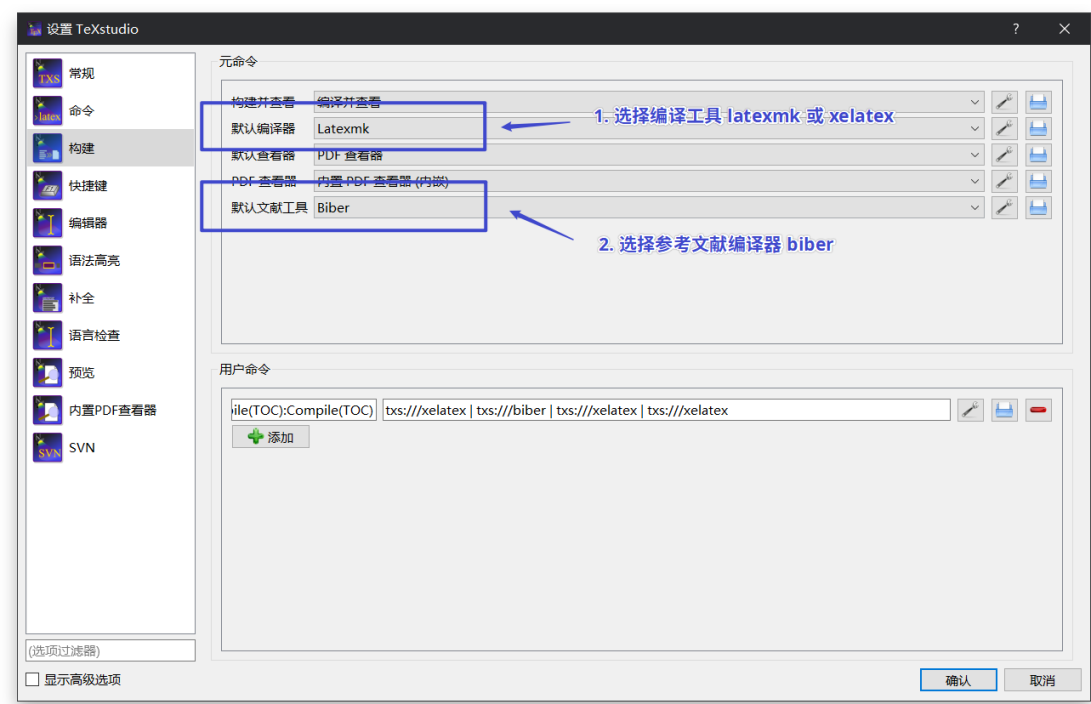


图 2-8 TeXstudio 挑选默认编译器和参考文献工具来编译 L<sup>A</sup>T<sub>E</sub>X 项目

你可以使用快捷键 **F5** 一键编译与预览 L<sup>A</sup>T<sub>E</sub>X 项目。

接下来，请继续在《第 3 章》、《第 4 章》和《第 5 章》中阅读各个模板的详细模块介绍与用模板撰写论文的具体实施方法。



### 第 3 章 计算机学院本科生开题报告使用指南

本模板已经发布在 Overleaf 上，你可以打开直接使用（点击下图 3-9 所示中的 Open as Template 即可）：

<https://www.overleaf.com/latex/templates/bei-jing-li-gong-da-xue-ben-ke-sheng-bi-ye-lun-wen-kai-ti-bao-gao-mo-ban/dgqdjptfqtrn>



图 3-9 Overleaf 在线版本的开题报告模板

Overleaf 缺少一些微软版权字体（比如宋体、黑体等），因此如果你希望格式完全准确，请使用本机进行编辑。

#### 3.1 熟悉项目

```
/
├── README.md
├── main.pdf
├── main.tex
├── merge-sort-recursion-tree.png
├── misc
│   ├── cover.tex
│   ├── refs.bib
│   └── reviewTableBlank.pdf
```

本项目由四个主要文件编译而成：`main.tex`、`cover.tex`、`refs.bib` 与 `reviewTableBlank.pdf`（也包括文档中所涉及到的图片等素材文件，比如：`merge-sort-recursion-tree.png`）。请大家重点关注这四个文件的功能与作用：

- `main.tex` 开题报告的开始文件（主文件），你的报告内容应该从此文件开始撰写。  
`main.tex` 中有详细的注释，介绍了每一部分内容都有什么作用，请仔细阅读后进行相应的修改、
- `main.pdf` 开题报告编译得到的 PDF 文件
- `./misc` 开题报告中所需要的杂项所在文件夹，其中包含有：
  - `cover.tex` 开题报告封面，按照教务部提供的封面设计，如无特殊需要请不要修改
  - `reviewTableBlank.pdf` 开题报告 PDF 格式的“评审表”，由于考虑到评审表后期由评委老师填写，因此本部分如无需要也无需改动
  - `refs.bib` 开题报告的参考文献 BibTeX 数据库，你应该向其中加入开题报告中所需要的所有参考文献的 BibTeX 格式引用（详见下文）

3.2 你的内容从哪里开始？

开题报告项目结构相对来说比较简单，因此你只需要重点关注 `main.tex` 这一文件——项目的主文件。你的内容应该从 `main.tex` 第 127 行的 % 内容开始开始。你需要重点关注的部分有：

表 3-1 开题报告内容概要

文章部分	内容主旨	对应 L <sup>A</sup> T <sub>E</sub> X 模板 section
第一部分	选题内容	<code>\section{毕业设计（论文）}</code>
第二部分	研究方案	<code>\section{研究方案}</code>
2-1	主要任务	<code>\subsection{本选题的主要任务}</code>
2-2	技术方案	<code>\subsection{技术方案的分析、选择}</code>
2-3	实施方案所需环境	<code>\subsection{实施技术方案所需的条件}</code>
2-4	存在问题与技术关键	<code>\subsection{存在的主要问题和关键技术关键}</code>
2-5	预期研究目标	<code>\subsection{预期能够达到的研究目标}</code>
第三部分	课题计划进度表	<code>\section{课题计划进度表}</code>

以及最后的“参考文献”。你应该将参考文献的 BibTeX 引用复制进入 `./misc/refs.bib`，并在正文中用 `\cite{}` 的方法进行引用。其中 BibTeX 格式的引用内容可以在谷歌学术中搜索文章直接复制得到，也可以考虑使用 Zotero 等文献管理工具批量生成。

### 3.3 其他注意事项



有关具体的 L<sup>A</sup>T<sub>E</sub>X 语法, 请参考前文中《第二章 2.1》给出的参考链接与学习文档。以下是模板中提供的一些示例性代码。

#### 3.3.1 插入图片

如果你希望加入图片, 可以将图片直接放在根目录 (比如此处的 `merge-sort-recursion-tree.png`), 或者统一将图片安置在一个文件夹下, 在正文里按照相对路径进行引用。模板中有一处插入图片的参考样例, 位于 `main.tex` 的第 138 行, 可以进行参考。比如, 我填入一个放在 `images/BIT_Name.jpg` 处的图片:

```
1 \begin{figure}[!ht]
2   \centering
3   \includegraphics[width=0.6\linewidth]{images/BIT_Name.jpg}
4   \caption{北京理工大学（一张示意图）}
5   \label{fig:BITName}
6 \end{figure}
```

这样就会渲染如图 3-10 的效果:



图 3-10 北京理工大学（一张示意图）

#### 3.3.2 插入表格

如果你希望插入表格, 可以统一使用 LaTeX Tables Generator 进行生成, 再粘贴进入模板之中。模板中有两处表格的参考样例, 分别位于 第 151 行 和 第 176 行, 可以进行参考。比如:


```
1 \begin{table}[!ht]
2   \centering
```

```
3 \caption{硬件、软件环境}
4 \label{tab:soft-hardware}
5 \begin{tabular}{@{}lcl@{}}
6 \toprule
7 & 指标 & & \multicolumn{1}{c}{版本参数} \\
8 & \multicolumn{2}{*}{硬件环境} & CPU & Intel i7-6500U \\
9 & & & RAM & 8 GB \\
10 & \multicolumn{2}{*}{软件环境} & 操作系统 & \\
11 & & & Python & Python 3.7.6 \\
12 \end{tabular}
13 \end{table}
```

渲染效果如表 3-1 所示：

表 3-2 硬件、软件环境		
	指标	版本参数
硬件环境	CPU	Intel i7-6500U
	RAM	8 GB
软件环境	操作系统	Windows 10 Pro x86_64 Ubuntu 18.04.3 LTS
	Python	Python 3.7.6

第 4 章 北京理工大学本科生毕业设计论文模板使用指南



注意：目前版本的毕业设计论文是按照北京理工大学计算机学院 2015 级毕业论文模板进行的设计与排版，如果 2016 级毕业论文模板有任何格式更新，我们会及时在这里更新。

本模板已经发布在 Overleaf 上，你可以打开直接使用（点击下图 4-11 所示中的 Open as Template 即可）：

<https://www.overleaf.com/latex/templates/bei-jing-li-gong-da-xue-ben-ke-sheng-bi-ye-she-ji-lun-wen-mo-ban/mwhjgqsnccxxg>

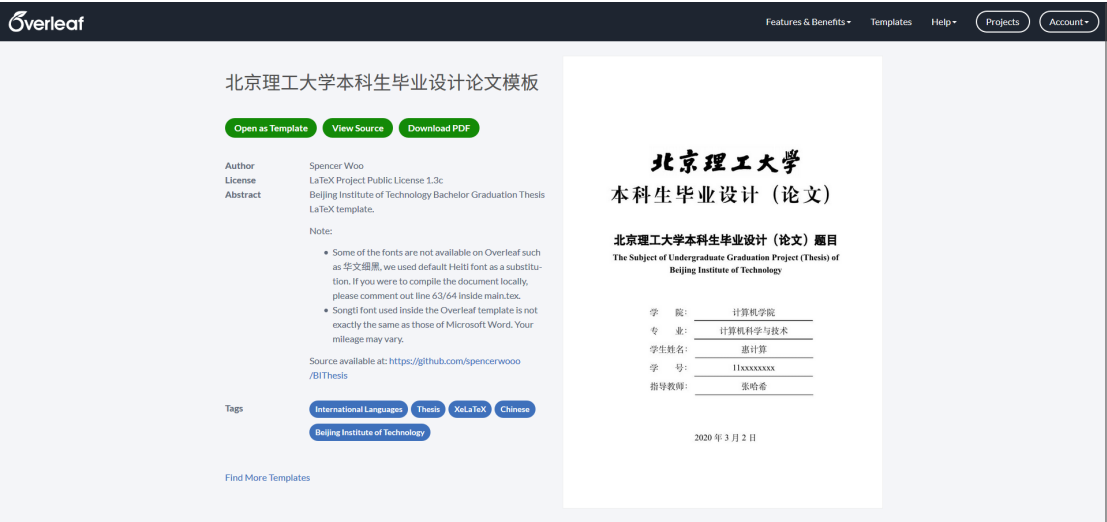
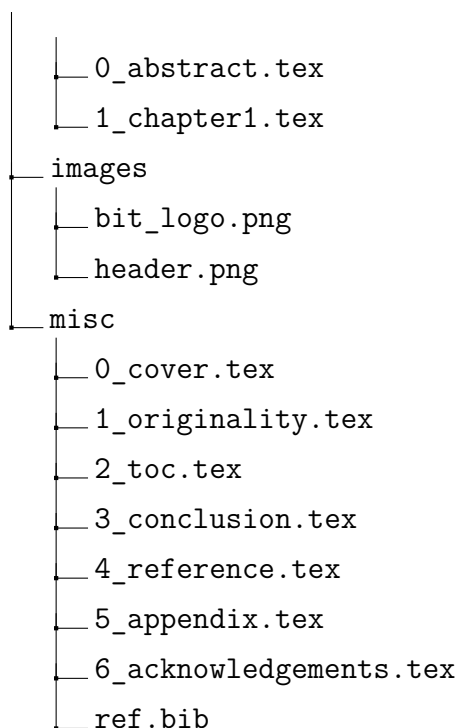


图 4-11 Overleaf 在线版本的毕业论文模板

Overleaf 版本的毕业论文模板中由于没有微软版权字体“华文细黑”，导致封面的毕业论文中文大标题无法用 Word 模板中规定的字体渲染，使得最终呈现样式与要求有些出入，如果希望保证 L<sup>A</sup>T<sub>E</sub>X 模板输出和学校模板一致，那么还是推荐在本地进行撰写和编译。

4.1 熟悉项目

```
/
├── README.md
├── main.tex
├── main.pdf
└── chapters
```



本项目由一个主文件和与之并存的几个辅助文件夹中的文件构成：

**main.tex** 毕业论文模板的主文件

**./chapters** 文件夹：包含有整个毕业论文的“摘要”和正文的全部“章节”

**0\_abstract.tex** 毕业论文的“摘要”（中文摘要与英文摘要）

**1\_chapter1.tex** 毕业论文正文“第一章”（示例章节）

...（你可以继续添加第二章 **2\_chapter2.tex**、第三章 **3\_chapter3.tex** .....，并在主文件 **main.tex** 中引用（详见下文）

**./misc** 文件夹：包含有毕业论文模板中的封面、后置章节与参考文献

**0\_cover.tex** 毕业论文的“封面”，一般情况无需更改

**1\_originality.tex** 毕业论文的“原创性声明”，一般情况无需更改（签字和日期后期手动添加）

**2\_toc.tex** 毕业论文的“目录”，一般情况无需更改（由  $\text{\LaTeX}$  自动生成）

**3\_conclusion.tex** 毕业论文的“结论”，按照一般章节文件对待

**4\_reference.tex** 毕业论文的“参考文献”，一般情况无需更改（由  $\text{\LaTeX}$  根据你文档中的  $\text{\cite{}}$  自动生成）

**5\_appendix.tex** 毕业论文的“附录”，按照一般章节文件对待

**6\_acknow...ments.tex** 毕业论文的“致谢”，按照一般章节文件对待

**ref.bib** 参考文献  $\text{BIB}\text{\TeX}$  数据库

主文件与其余文件之间的引用关系大致如下图 4-12 所示：

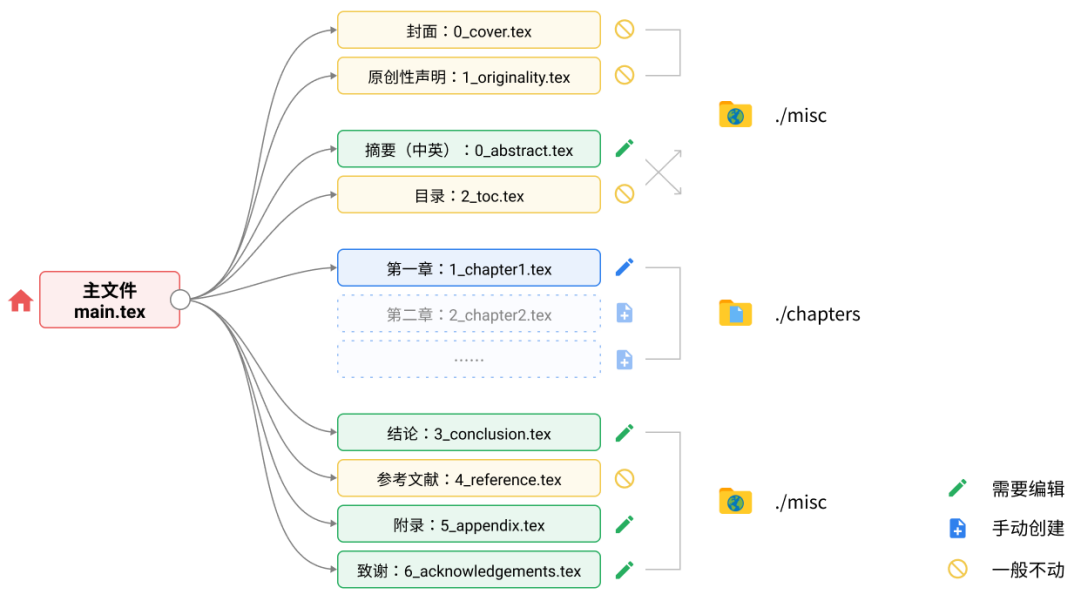


图 4-12 毕业论文模板主模块与各个分支之间的关系

## 4.2 你的内容从哪里开始？

熟悉项目之后，你应该发现，我们毕业设计论文共分为如下的几个模块：封面、原创性声明、中英文摘要、目录、正文（多个章节）、结论、参考文献、附录与致谢。我们的整个毕业设计论文  $\text{\LaTeX}$  项目将每个模块单独提取出来，成为单独的  $\text{\LaTeX}$  文件，使用 `main.tex` 主文件统一引用，方便各位进行分模块的修改。你需要重点关注的地方有如下几个。

### 4.2.1 开始

首先，你需要定义毕业设计论文的“中文标题”和“英文标题”，这两个“变量”将影响模板封面的渲染，以及后续摘要中出现的标题的渲染。

中英文标题的定义位于 `main.tex` 的第 65 至第 67 行：

- 你可以通过控制 `\thesisTitle` 这一变量来控制整个论文的“中文标题”
- 你可以通过控制 `\thesisTitleEN` 这一变量来控制整个项目的“英文标题”

接下来，你需要定义你的个人信息，这些信息将被渲染在毕业设计论文的封面。个人信息包括你所在学院，你的专业、学号、姓名和指导教师。

个人信息的定义位于 `main.tex` 的第 69 行至第 74 行：

- `\deptName`：你所在学院
- `\majorName`：你所就读的专业

- `\yourName`: 你的姓名
- `\yourStudentID`: 你的学号
- `\mentorName`: 你的指导教师

### 4.2.2 中英摘要

接下来, 你需要撰写论文的摘要。模板中英文摘要位于 `chapters/0_abstract.tex`:

- 中文摘要位于 `0_abstract.tex` 的第 41 行至第 48 行。其中第 48 行定义摘要的中文关键词
- 英文摘要位于 `0_abstract.tex` 的第 71 行至第 76 行。其中第 76 行定义摘要的英文关键词

### 4.2.3 正文

正文是一篇论文中最为重要的部分, 是一篇论文的核心。正文部分可以分为多个章节, 模板中仅创建了第一章的示范性文件: `chapters/1_chapter1.tex`, 你可以将它作为正文章节的“模板”, 继续在 `./chapters` 目录下自行创建第二章 `2_chapter2.tex`、第三章 `3_chapter3.tex` 等等, 并需要在 `main.tex` 的第 199 行处添加对应章节文件的相对路径引用:

```
% 第一章
\input{chapters/1_chapter1.tex}
% 在这里添加第二章、第三章……TeX 文件的引用
\input{chapters/2_chapter2.tex}
\input{chapters/3_chapter3.tex}
```

之后, 你可以分别在每个章节独立的 `TeX` 文件中撰写每一章节的内容。

### 4.2.4 后续模块

在正文之后, 我们的论文还剩下: 结论、参考文献、附录与致谢这四个模块。它们依次位于:

- Conclusion 结论: `misc/3_conclusion.tex`
- Reference 参考文献: `misc/4_reference.tex`
- Appendix 附录: `misc/5_appendix.tex`
- Acknowledgements 致谢: `misc/6_acknowledgements.tex`



其中,你不需要手动编辑“参考文献”这一文件,只需要撰写“结论”、“附录”和“致谢”即可。这三个模块的撰写逻辑与前面正文章节的撰写逻辑是一致的。



有关具体的 L<sup>A</sup>T<sub>E</sub>X 语法,请参考前文中《第二章 2.1》给出的参考链接与学习文档。接下来是模板中提供的一些示例性代码的使用方法。

### 4.3 参考文献管理

为了保证你的毕业论文的参考文献格式标准,你需要将参考文献的 Bib<sub>T</sub>E<sub>X</sub> 引用复制进入 ./misc/refs.bib,并在正文中用 \cite{xxx} 的方法进行引用。

Bib<sub>T</sub>E<sub>X</sub> 是一种表示、存储与引用参考文献的语法,谷歌学术中搜索文章直接复制得到,如图 4-13 所示:

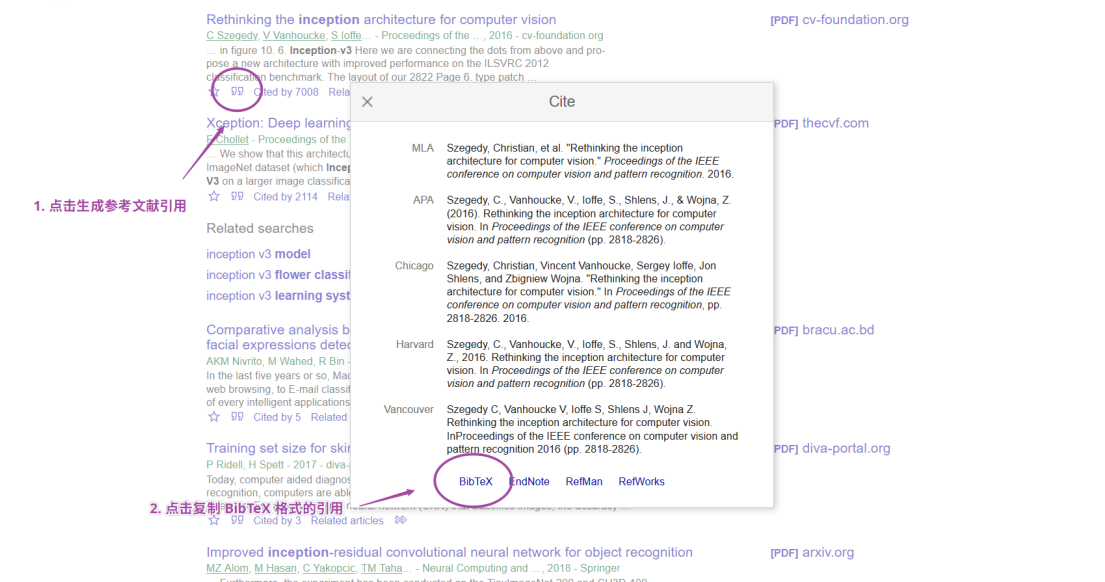


图 4-13 谷歌学术搜索并复制文献的 Bib<sub>T</sub>E<sub>X</sub> 格式

复制得到的参考文献 Bib<sub>T</sub>E<sub>X</sub> 类似:

```

1 @inproceedings{szegedy2016rethinking,
2   title={Rethinking the inception architecture for computer
3     ↪ vision},
4   author={Szegedy, Christian and Vanhoucke, Vincent and Ioffe,
5     ↪ Sergey and Shlens, Jon and Wojna, Zbigniew},
6   booktitle={Proceedings of the IEEE conference on computer
7     ↪ vision and pattern recognition},

```

```
5     pages={2818--2826},
6     year={2016}
7 }
```

将上面的内容复制进入 misc/ref.bib 即可，之后你就可以直接在文章中使用这一参考文献的地方用类似下面的方法引用这一标签为 szegedy2016rethinking 的参考文献：

```
1 正文，正文正文 \cite{szegedy2016rethinking} 正文正文……
```

另外，你也可以考虑使用 Zotero 等专业文献管理工具批量生成。参考：文献管理神器 Zotero 学习路径指南。

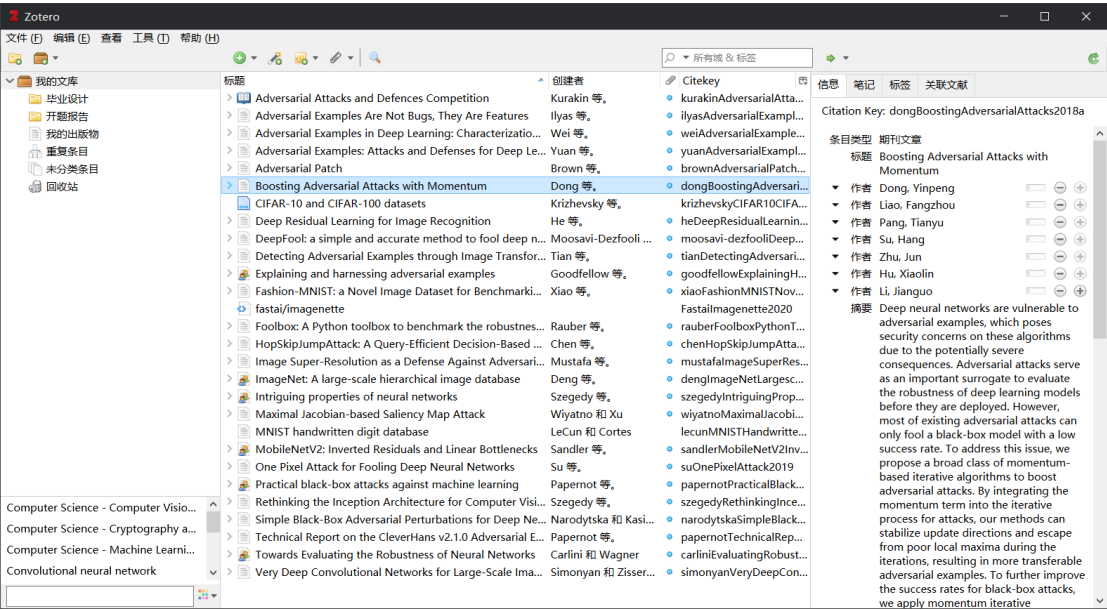


图 4-14 专业文献管理工具 Zotero

4.4 图片素材

整个模板的图片素材都整理在图片文件夹中： ./images。你可以将论文中使用到的图片统一放在这一目录下进行管理，在论文中使用“相对路径”进行引用。你可以用类似下面的语法引用图片：

```
1 \begin{figure}[htbp]
2     \vspace{13pt} % 调整图片与上文的垂直距离
3     \centering
```

```

4 \includegraphics[width=0.8\textwidth]{images/bit_logo.png}
5 \caption{标题序号}
6 \label{标题序号} % label 用来在文中索引
7 \end{figure}

```

可以看到：

- 我们首先将图片放置在了一个 `\begin{figure} ... \end{figure}` 的环境中，其中 `[htbp]` 是用于定位图片。
- 之后，在环境中，我们首先使用 `\centering` 保证图片水平居中
- 之后用 `\includegraphics[图片大小]{图片路径}` 的格式引用了图片本身（图片大小的语法 `width=0.8\textwidth` 表示图片宽度是整个页宽的 0.8 倍）
- 最后我们定义了图片的说明文字 `\caption{图片说明}` 和图片的标签编号 `\label{图片编号}`，前者显示在图片下方起到说明注释的作用，后者让我们可以用 `\ref{图片编号}` 的语法来在正文中引用图片

请注意，为了保证图片引用的格式和 Word 模板完全一致，我们手动设置了 `\vspace{13pt}` 的垂直空白，你引用新图片时，也需要添加这一垂直空白。

在第一章节 `chapters/1_chapter1.tex` 中的第 38 行至第 43 行 是一个示范。

## 4.5 表格插入

表格一直是  $\text{\LaTeX}$  排版系统非常强大又非常不好实现的一个模块，如果你希望方便的插入表格，可以统一使用 LaTeX Tables Generator 进行生成，再粘贴进入模板之中。

在第一章节 `chapters/1_chapter1.tex` 中的第 47 行至第 60 行 是一个示范。

## 4.6 公式插入

$\text{\LaTeX}$  的行内数学符号和公式等，使用 `\( ... \)` 的语法进行定义。比如类似如下的正文：

```

1 The well known Pythagorean theorem  $(x^2 + y^2 = z^2)$  was
2 proved to be invalid for other exponents.
3 Meaning the next equation has no integer solutions:

```

4  
5
$$\backslash[ x^{\wedge}n + y^{\wedge}n = z^{\wedge}n \backslash]$$

即可非常简单的渲染如下的公式效果：

The well known Pythagorean theorem  $x^2 + y^2 = z^2$  was proved to be invalid for other exponents. Meaning the next equation has no integer solutions:

$$x^n + y^n = z^n$$

另外，行内数学环境也可以用  $\$ \dots \$$  的语法进行定义：

1  
2

In physics, the mass-energy equivalence is stated  
by the equation  $\$E=mc^2\$$ , discovered in 1905 by Albert Einstein.

In physics, the mass-energy equivalence is stated by the equation  $E = mc^2$ , discovered in 1905 by Albert Einstein.

复杂的独立模块数学公式可以用如下的语法进行定义：

1  
2  
3

```
\begin{equation}
LRI=1/\wedge \sqrt{1+\{\left(\frac{\mu }{ }_{R}\)}{\{\mu
\rightarrow \}_{s}\}\right)\}^{2}\{\left(\frac{\delta }{ }_{R}\)}{\{\delta
\rightarrow \}_{s}\}\right)\}^{2}}
\end{equation}
```

$$LRI = 1/\sqrt{1 + \left(\frac{\mu_R}{\mu_s}\right)^2 \left(\frac{\delta_R}{\delta_s}\right)^2} \quad (1)$$

为了保证与 Word 模板中的数学公式要求一致，我们的  $\text{\LaTeX}$  模板中的公式默认会进行相应的编号（比如上面的例子）。在第一章节 `chapters/1_chapter1.tex` 中的第 67 行至第 69 行 是一个示范。

## 4.7 其他

以下模块的使用可能需要你手动在 `main.tex` 的开头用 `\usepackage{ ... }` 的方法引入其他  $\text{\LaTeX}$  宏包。

### 4.7.1 代码高亮

你可以使用 `minted` 宏包来进行代码块的渲染。比如：

- 在文档开题引入宏包：

```
\usepackage{minted}
```

- 渲染代码块：

```
\begin{minted}{python}
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int) #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)
\end{minted}
```

你就会得到类似下面的渲染效果：

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int) #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)
```

有关 `minted` 的更多使用方法，请阅读：Code Highlighting with `minted`。

如果你在使用 `minted` 的过程中遇到了任何问题，请阅读：《第 7 章：疑难杂症 Troubleshooting》。

### 4.7.2 算法模块

你可以使用 `algorithm2e` 宏包来渲染一个“伪代码算法”模块。比如：

- 在文档开头引入宏包：

```
\usepackage[ruled,vlined]{algorithm2e}
```

- 渲染伪代码模块：

```
\begin{algorithm}[H]
\SetAlgoLined
\KwResult{Write here the result }
initialization\;
\While{While condition}{
instructions\;
\eIf{condition}{
instructions1\;
instructions2\;
}{
instructions3\;
}
}
\caption{How to write algorithms}
\end{algorithm}
```

这样，你就会得到类似如下的渲染效果：

---

#### Algorithm 1: How to write algorithms

---

**Result:** Write here the result

initialization;

**while** *While condition* **do**

instructions;

**if** *condition* **then**

instructions1;

instructions2;

**else**

instructions3;

**end**

**end**

---

有关更多伪代码算法模块的使用，请阅读：Algorithms。

## 第 5 章 通用北京理工大学本科生实验报告模板使用指南



说明：这个实验报告模板是一个通用的报告模板，不适用所有实验报告要求。实验课程未提供实验报告模板时可以使用该模板。当前本实验报告模板只包含一个封面，欢迎大家往项目仓库 PR 制作更多的封面。

### 5.1 熟悉项目

```
/
├── README.md
├── main.pdf
├── main.tex
├── misc
│   └── cover_v1.tex
├── assets
│   └── .....
└──
```

**main.tex** tex 源文件，本实验报告模板的主体文件，所有需要添加的内容都在该文件里进行修改即可

**main.pdf** 编译项目生成的 pdf 文件

**./misc** 杂项（包括实验报告封面等）：

**cover\_v1.tex** 这是一个示范性的报告封面，该文件无需修改

**./asset** 一些图片资源存放文件夹

### 5.2 编译方式与使用

由于实验报告模板没有涉及到参考文献的使用，因此我们只需要使用  $\text{XeLaTeX}$  即可进行全文编译。

整个项目的编译工具链的顺序为：



其中，按照 VS Code 的 LaTeX Workshop 设置格式： $\text{XeLaTeX}$  的编译命令为：

```
{
  "name": "xelatex",
```

```
"command": "xelatex",  
"args": [  
    "-synctex=1",  
    "-interaction=nonstopmode",  
    "-file-line-error",  
    "-pdf",  
    "-outdir=%OUTDIR%",  
    "-cd",  
    "%DOC%"  
],  
"env": {}  
}
```

整个编译的 recipe 为:

```
{  
  "name": "xelatex * 3",  
  "tools": [  
    "xelatex",  
    "xelatex",  
    "xelatex"  
  ]  
}
```

使用 各种内容的插入请参考源文件。



## 第 6 章 如何将 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 文档转换为 Word

如果你决定使用  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  进行论文撰写，但是你的导师希望你提供 Word 版本的论文进行方便的批注等，你可以使用下面的方法来将你的  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  论文转换为 Word 格式的文档。需要注意的是，这种方式的转换并不能完整的将  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  论文的全部格式进行转换，仅能保证正文部分内容的丢失，其余包括“论文封面”、“表格”等等部分的内容，在转换的过程中都可能丢失。这些部分需要你后期手动进行添加。

### 6.1 安装 Pandoc 命令行工具

Pandoc 是一个支持几乎所有 Markup 格式文档的“通用文档转换器”，是一个命令行工具，支持所有常用操作系统。我们可以借助 pandoc 进行格式转换。

Pandoc 详细的安装说明请见官方文档：Installing pandoc.

简单来说：

- 在 Windows 上：
  - 你可以使用 scoop 包管理工具安装 pandoc（为了方便设置命令行工具的环境变量，推荐使用 scoop 进行安装）：

```
scoop install pandoc
```

关联阅读：“一行代码”搞定软件安装卸载，用 Scoop 管理你的 Windows 软件

- 当然你也可以直接在 Pandoc 的 GitHub Release 页面 下载 Windows MSI 安装文件手动安装
- 在 macOS 上你可以使用 Homebrew 包管理工具安装 pandoc：

```
brew install pandoc
```

- 在 Linux 上你可以使用你所用发行版的包管理工具安装 pandoc，比如：

```
sudo apt install pandoc
```

之后，在终端中输入：

```
pandoc --version
```

如果出现类似下面的输出，说明你的 pandoc 安装成功。

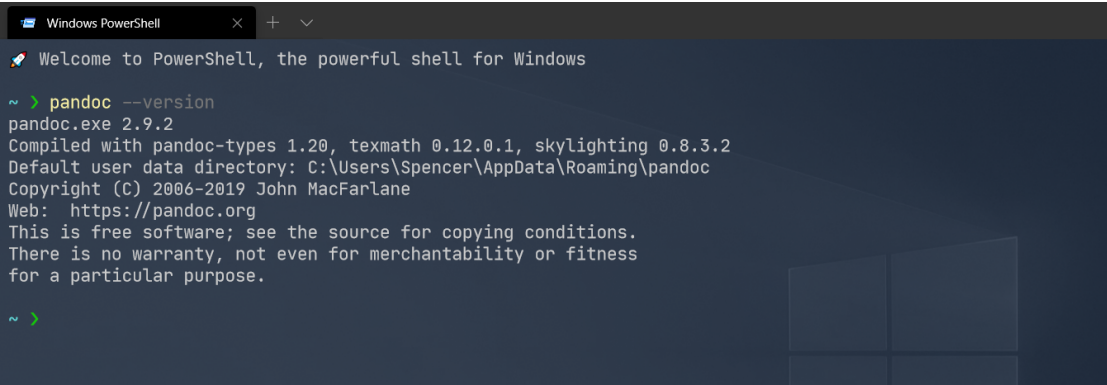


图 6-15 测试 pandoc 安装成功

6.2 完善 Word 格式的模板文件

为了保证导出的 Word 文档格式和学校提供的模板大体一致，我们需要确认 Word 版本的模板已经准确定义了各级标题、正文等部分的格式。

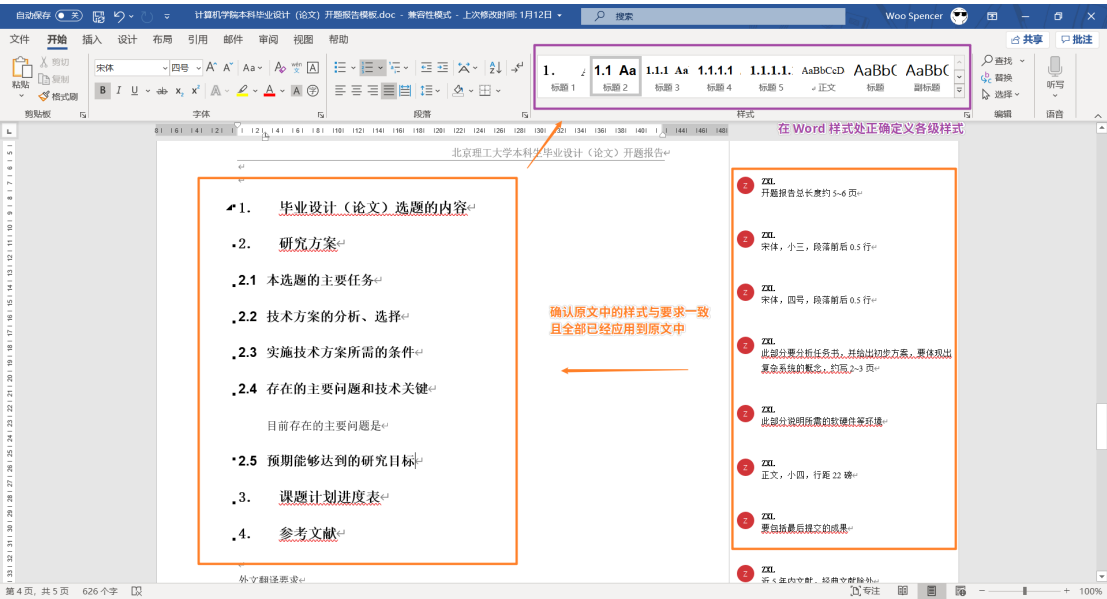


图 6-16 确认 Word 版本的模板准确生成

之后，我们需要将这一文件（doc 或 docx）保存，留作 pandoc 的格式参考。

6.3 进行格式转换

最后，我们进行格式的转换。

### 6.3.1 朴素格式转换

如果你只希望将文本内容导出为 Word，不在意格式或其他内容的正确性，你可以直接使用下面的命令进行最普通的文本转换：

```
pandoc {LaTeX 文档文件} -o {输出 Word 文档}
```

比如：

```
pandoc main.tex -o main.docx
```

没有特别指明模板 Word 文档格式与参考文献文档的情况下，pandoc 仅会处理你  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  文档中的文字内容，按照标题、正文的格式整理进入 Word。

### 6.3.2 含有目标模板 Word 文档的格式转换

如果你希望按照模板 Word 文档的规定格式进行转换，那么你可以直接使用下面的命令进行格式转换：

```
1 pandoc {LaTeX 文档文件} --reference-doc={参考模板 Word 文档} -o  
  ↪ {输出 Word 文档}
```

比如，我们的  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  文档文件名称为 `main.tex`，参考模板 Word 文档名称为 `template.docx`，希望输出名为 `main.docx` 的 Word 文档，我们即可如下组织 pandoc 转换命令：

```
pandoc main.tex --reference-doc=template.docx -o main.docx
```

### 6.3.3 含有参考文献文档的格式转换

如果你的 LaTeX 文档中包含有参考文献的引用，那么你需要特别明确参考文献  $\text{BibT}_{\text{E}}\text{X}$  文件，将文件以 `--bibliography={参考文献文件}` 的参数告知 pandoc，从而让 pandoc 正确处理你的参考文献。比如：

```
1 pandoc main.tex --bibliography=refs.bib  
  ↪ --reference-doc=template.docx -o main.docx
```

特别强调：pandoc 格式转换功能有限，无法处理  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  多级嵌套表格（在本模板的“开题报告”与“毕业论文”中都有使用多级嵌套表格，直接调用 pandoc 默认情况下会报错。需要删掉  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  文档中的表格，并后期手动录入 Word 之中），也无法保证格式与  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  原文档 **完全一致**，都需要我们后期手动进行调整。

## 第 7 章 疑难杂症

### 7.1 出现字体缺失的编译失败提示

一般情况下，如果你在编译“毕业设计论文”模板时出现了类似下面的编译报错：

```
Package fontspec Error: The font "STXihei" cannot be found. ...
```

这是由于你的电脑中尚未安装“华文细黑”这一字体，或你的电脑上安装的“华文细黑”字体文件名称不是 STXIHEI.TTF，导致 L<sup>A</sup>T<sub>E</sub>X 编译器找不到这一字体，也就导致无法正常编译模板。（毕业论文模板的封面中，中文标题要求字体为“华文细黑”。）

你可以通过下面的方法对这一问题进行排查。首先，在终端中运行：

```
fc-list :lang-zh > fclist.txt
```

这一命令会将你系统中安装的字体全部列出并保存在你执行命令所在目录下的 fclist.txt 文件中，你可以用文本编辑器打开这一文件，全局搜索“华文细黑”：

```
1995 C:/Users/Spencer/texlive/2019/texmf-dist/fonts/truetype/bh/gofonts/GoMono-Regular-Italic.ttf: Go Mono:style=Italic
1996 C:/Users/Spencer/texlive/2019/texmf-dist/fonts/truetype/paratype/ptserif/PTF56F.ttf: PT Serif:style=Italic
1997 C:/WINDOWS/fonts/sarasa-mono-sc-extralight.ttf: Sarasa Mono SC,等距更纱黑体 SC,Sarasa Mono SC Extralight,等距更纱黑体 SC
Extralight:style=Extralight,Regular
1998 C:/WINDOWS/fonts/SmallFont/SmallFont:style=Regular
1999 C:/WINDOWS/fonts/STXIHEI.TTF: STXihei,华文细黑:style=Regular
2000 C:/WINDOWS/fonts/serif/serif:style=Regular
2001 C:/WINDOWS/fonts/simkai.ttf: KaiTi,楷体:style=Regular,Normal,obyčejné,Standard,Kavoviká,Normaali,Normal,Normale,Standaard,
Обычный,Normalne,Navadno,Arrunta
2002 C:/Users/Spencer/texlive/2019/texmf-dist/fonts/opentype/gust/poltawski/antpoltltcond-italic.otf: Antykwa Poltawskiego Li
Cond:style=Cond Italic,Italic
2003 C:/Users/Spencer/texlive/2019/texmf-dist/fonts/opentype/public/drm/drmctbx12.otf: drmtcbx12:style=Regular
```

图 7-17 字体搜索

如果你发现自己系统中并没有这一字体，需要手动安装，那么你需要确保安装之后字体文件的名称为 STXIHEI.TTF。你可以在 Windows 的 C:\Windows\Fonts 目录下找到你系统全局安装的字体，找到“华文细黑”并“右键 » 属性”，确认如下图所示：

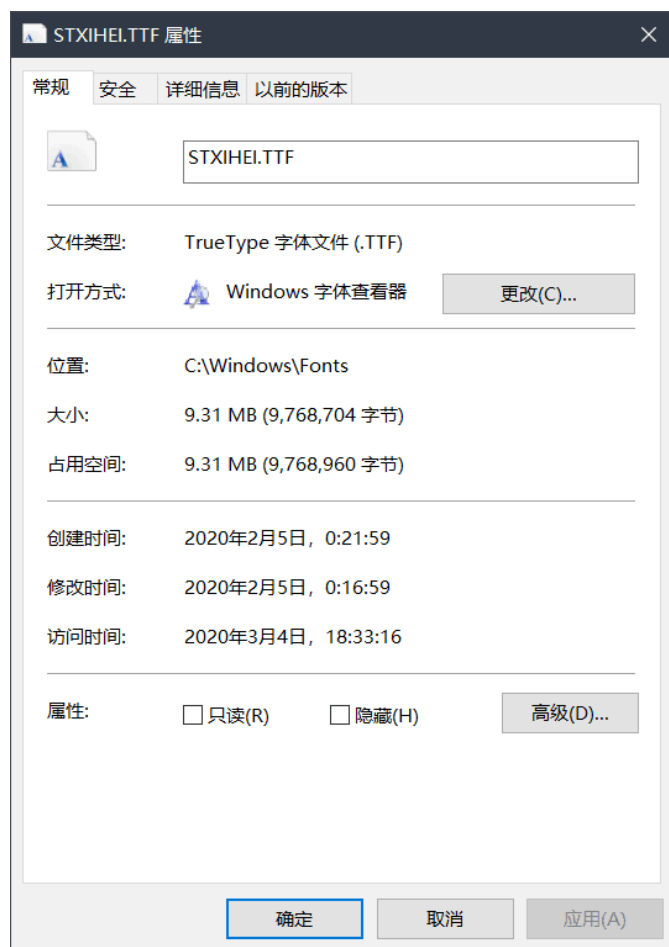


图 7-18 字体属性

之后，重启电脑，刷新字体缓存，确认编译状况。

如果  $\text{L}^{\text{T}}_{\text{E}}\text{X}$  编译器依旧找不到相关字体，我推荐重新安装 Windows 系统，或者将定义“华文细黑”字体的地方（毕业论文模板 `main.tex` 第 61 行至 63 行）与使用“华文细黑”字体的地方（毕业论文模板 `misc/0_cover.tex` 第 43 行）删除，用默认字体替代。

## 7.2 出现参考文献样式找不到的编译失败提示

开题报告与毕业论文的参考文献均使用了 `biblatex-gb7714-2015` 宏包生成符合《GB/T 7714-2015 信息与文献参考文献著录规则》规定的参考文献。这一参考文献宏包仅适用于最新版本的  $\text{T}_{\text{E}}\text{X}$ Live 发行版（ $\text{T}_{\text{E}}\text{X}$ Live 2019）。如果你在编译过程中出现了类似如下的报错：

```
Error: Style 'gb7714-2015' not found.
```

那么就是由于你的 TeXLive 发行版中没有包含这一宏包。需要你手动将 TeXLive 更新为 2019 版本，或手动下载相应的宏包。（跨版本升级 TeXLive 可能出现一些问题，推荐卸载重新安装。）

之后，在开始菜单中寻找 TeXLive Manager，点击打开，并搜索 biblatex-gb7714-2015，有如下输出表明你的宏包安装成功。

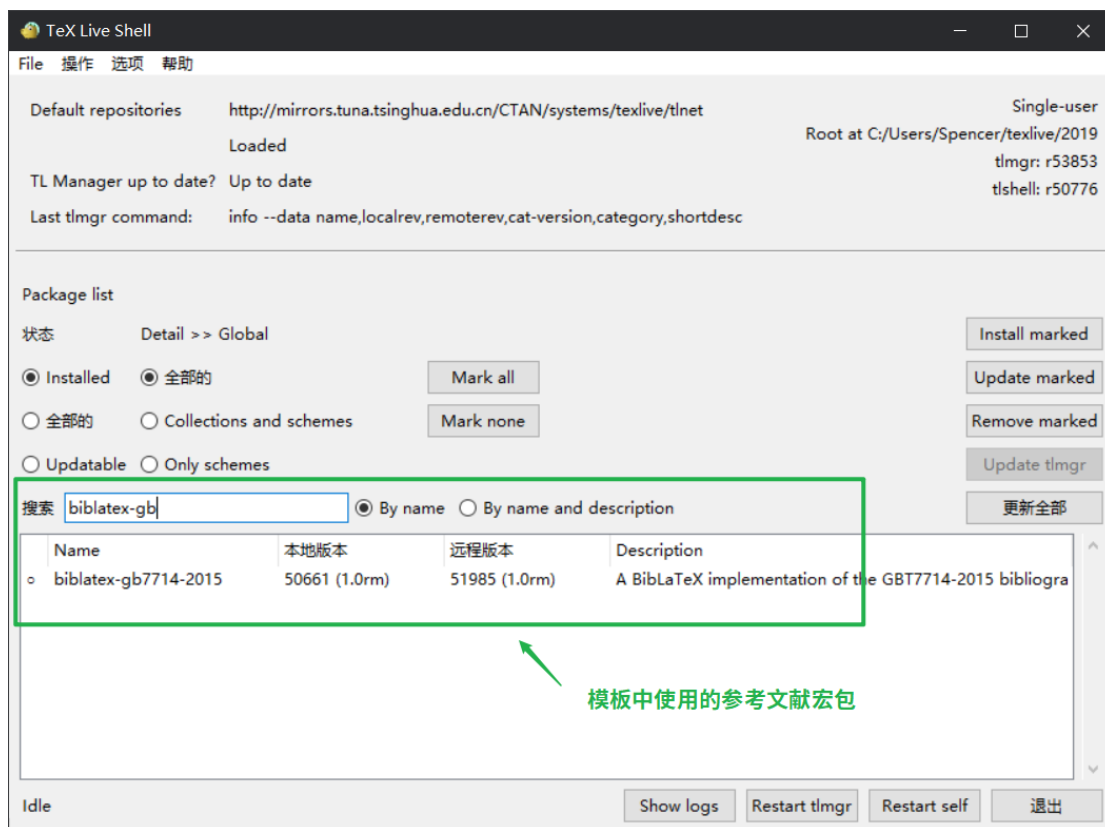


图 7-19 宏包安装成功

## 7.3 无法使用代码高亮 minted 宏包

在论文中不可避免的要加入“代码块”。一般我们代码高亮使用的宏包都是 minted，如果你发现插入 minted 环境后，编译失败，你可以尝试如下的方法解决：

### 7.3.1 排查是否正确安装 Python 与 pygments 包

如果你出现了类似如下的编译报错：

1

```
"Package minted Error: You must have `pygmentize' installed to
↪ use this package."
```

那么是由于你的 Python 环境中缺少必要的库。`minted` 背后事实上用的是 Python 的 `pygments` 库进行代码渲染和高亮，因此你也必须安装 Python 环境和 `pygments` 库。你可以使用 Python 的官方包管理工具 `pip` 安装 `pygments`：

```
pip install pygments
```

之后，你需要确认可执行文件 `pygmentize.exe` 位于系统路径上。用如下的方法进行验证，如果出现类似输出，说明你的安装应该没有问题：

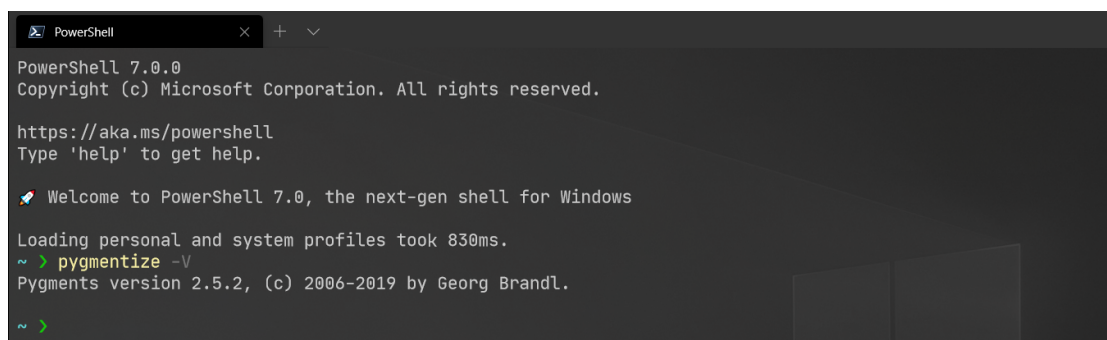


图 7-20 Pygmentize 安装成功

### 7.3.2 添加额外的编译参数

如果你出现了类似如下的编译报错：

```
1 Package minted Error: You must invoke LaTeX with the
   ↪ -shell-escape flag.
```

这是由于你的编译命令中没有添加 `-shell-escape` 参数，`minted` 宏包需要这一参数才能正确编译  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  文档。以 VS Code 的  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  Workshop 为例，如果你使用 `xelatex` 编译，那么你需要将 `xelatex` 的编译参数修改如下：

```
{
  "name": "xelatex",
  "command": "xelatex",
  "args": [
    "-synctex=1",
    "-interaction=nonstopmode",
    "-file-line-error",
    "-pdf",
  ]
}
```



```
    "-shell-escape",
    "-outdir=%OUTDIR%",
    "-cd",
    "%DOC%"
  ],
  "env": {}
}
```

如果你使用 `latexmk` 编译，那么你需要将 `latexmk` 的编译参数修改如下：

```
{
  "name": "latexmk",
  "command": "latexmk",
  "args": [
    "-synctex=1",
    "-interaction=nonstopmode",
    "-file-line-error",
    "-xelatex",
    "-shell-escape",
    "-outdir=%OUTDIR%",
    "-cd",
    "%DOC%"
  ],
  "env": {}
}
```

### 7.3.3 删除 `minted` 宏包的缓存文件夹

如果你出现了类似如下的编译报错：

```
! Undefined control sequence.
```

则可能是由于 `minted` 缓存导致。一般如果你编译过带有 `minted` 环境的 `LATEX` 项目，根目录都会有一个名称为 `_minted_doc` 的缓存文件夹。你可以尝试将这一文件夹删除，重新编译，排查问题。

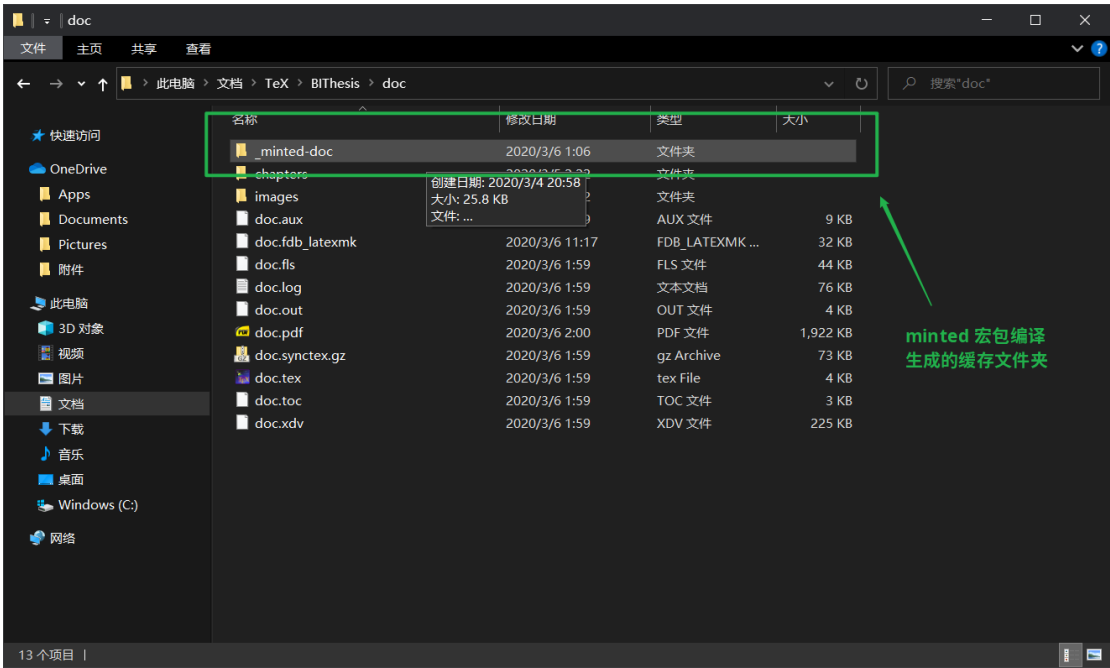


图 7-21 删除 \_minted-doc 缓存文件夹

7.4 编译过慢，一次更改需要编译半分钟

如果你觉得模板中 `xelatex -> biber -> xelatex -> xelatex` 四步编译太慢，每次都全量编译，需要等待半分钟才能出结果，你可以尝试使用 `latexmk` 进行编译。`latexmk` 每次会根据你 `LaTeX` 文档的更改，增量编译，从而加快对原文档进行微小变化后（比如只修改一个字）的编译速度。

另外，从我自己使用来看，`TeXstudio` 的编译速度一般都比 `VS Code` 快一些，如果你觉得 `VS Code` 的 `LaTeX Workshop` 编译太慢，可以考虑尝试使用 `TeXstudio`。

## 第 8 章 致谢

BIT<sub>HESIS</sub> 模板是一个团队项目。从一个开题报告的启发，到最终完成程度接近 100%，格式几乎完全匹配的毕业论文；从一个简陋的 README，到近一万两千字的在线版本 Wiki 文档和近一万字的 L<sup>A</sup>T<sub>E</sub>X PDF 文档（本文档）……这些离不开团队成员（2016 级计算机学院本科生武上博、2016 级计算机学院本科生王赞）两位同学近一个月的努力与维护，他们见证了 BIT<sub>HESIS</sub> 的从无到有。

作为 BIT<sub>HESIS</sub> 的创造者，我（武上博）由衷的感谢北京理工大学教务部、北京理工大学计算机学院教务处为我们项目提供的强大背书，没有支持我们的领导和老师们的助力，本项目将毫无意义；感谢最初打开、阅读、与分享这一项目的介绍公众号文章的各位老师与同学，没有愿意分享这一项目的你们，本项目将无人问津；感谢愿意为这一项目在 GitHub 上做出贡献、提出 pull request 的诸位同学，没有你们的贡献，本项目也不会有这么多的特色；感谢因为各种问题 in GitHub 的 issue 上进行询问的同学，没有你们的提问，本项目同样也不会这样完善。当然，最终，也要感谢使用 BIT<sub>HESIS</sub> 的同学们，你们为我们的工作赋予了意义，为我们的后续维护提供了动力，感谢你们，感谢所有人。

本文档是 BIT<sub>HESIS</sub> 的官方参考手册，由 2016 级计算机学院武上博同学、2016 级计算机学院王赞同学撰写，感谢 2016 级计算机学院唐誉铭同学、2017 级经济与管理学院詹熠莎同学提供的宝贵意见。