LaTeX 学习笔记

司英成

2021 年 12 月 19 日

摘要

本文是根据 github 上的一个 latex-cookbook(https://github.com/xinychen/latex-cookbook) 学习 LaTeX 的基本使用，本文也使用 LaTeX 编写。

目录

第一章 横空出世的 LaTeX **9**

1.1 Tex 和 LaTeX 9

1.2 引领浪潮的 LaTeX 10

1.2.1 LaTeX 的出现 10

1.2.2 LaTeX 的特点 11

1.2.3 LaTeX 编辑器 13

1.3 应运而生的在线系统 14

1.3.1 LaTeX 在线系统的出现 14

1.3.2 LaTeX 在线系统的特点 15

1.4 LaTeX 问答社区 17

1.4.1 问答社区的介绍 17

1.4.2 高频访问问题 18

1.5 关于 LaTeX 的开源项目 18

1.5.1 开源社区 GitHub 18

1.5.2 学位论文 LaTeX 模板 19

1.5.3 LaTeX 绘制图形 20

1.5.4 LaTeX 制作简历 21

1.5.5 LaTeX 制作幻灯片 21

1.5.6 LaTeX 制作海报 22

1.5.7 推荐指导本书的开源项目 22

1.6 LaTeX 制作中文文档 22

1.6.1 使用 CJKutf8 宏包 23

1.6.2 使用 CTEX 宏包 23

第二章 LaTeX 基础介绍 **25**

2.1 导言 25

2.2 LaTeX 语法规则 25
2.2.1 命令 26
2.2.2 环境 26
2.2.3 宏包 27
2.3 LaTeX 代码结构 27
2.3.1 前导代码 28
2.3.2 主体代码 29
2.4 文档类型介绍 30
2.4.1 基本介绍 30
2.5 简单文档的制作 31
2.5.1 制作封面 31
2.5.2 开始创建文档 31
2.6 一些基本命令 33
2.6.1 全局格式设置 33

引言

1977 年，计算机科学家克努斯博士[1]开发了一款名为 TeX 的文档排版系统，作为一种计算机程序语言，它能够专门用于制作各类技术文档，并且对制作包含数学公式的技术文档具有良好的适用性。克努斯博士开发 TeX 其实存在一些意外：上世纪 70 年代，克努斯博士在修改自己的著作时，由于当时的排版质量差到让他难以容忍，所以他便转而开始思考能否开发出高质量的文档排版系统。

在使用过程中，TeX 制作文档的方式非常特殊，与今天常用的办公软件 Word 等截然不同，它是完全使用计算机程序语言来制作文档的。由于其对计算机语言的高度依赖，这款系统的使用门槛较高，但也具有很多优点，其中最为人称道的优点是它可以书写大量复杂的数学表达式。基于 TeX，兰波特博士 [2]于 1985 年开发了另一款文档排版系统，名为 LaTeX，兰波特博士设计这款系统初衷是让人们从排版样式这些繁琐的细节中解放出来，从而将精力集中在文档结构和文档内容上，这一做法很快便让 LaTeX 取代了 TeX。后来，LaTeX 的众多开发者对 LaTeX 最初版本进行了更新和提升，也就是我们今天一直在用的 LaTeX。

[1]Donald E. Knuth，直译名为唐纳德 · 尔文 · 克努斯，中文名为高德纳，美国计算机科学家，现代计算机科学的先驱人物，在计算机科学及数学领域著有多部影响深远的著作，于 1974 年获得图灵奖。

[2]Leslie Lamport，直译名为莱斯利 · 兰波特，美国计算机科学家，于 2013 年获得图灵奖，他获得图灵奖的原因并非在于开发了 LaTeX，而是源于他在所研究的学术领域做出的突出贡献。

第一章　横空出世的 LaTeX

1.1　Tex 和 LaTeX

TeX 是一种专门用于文档排版的计算机程序语言，同时也是一款文档排版系统，它几乎和微软推出的 Office 办公软件同时出现，后来成为人们制作文档的两种最佳工具。TeX 和 Office 制作文档的方式截然不同，Office 的使用门槛并不高，只要掌握一些基本操作就能够制作文档；而 TeX 则需要一定的计算机编程基础，除了一些基本命令，还要掌握 TeX 环境和一些特定的宏包。实际应用中，TeX 以其高质量、高效率的排版输出，特别是数学公式的排版能力而闻名，被科研工作者广泛用于科技文档的制作。

TeX 是怎么出现的呢? 有时候，新生事物的出现往往会伴随着一定的契机和巧合。在 20 世纪 70 年代末，克努斯博士正准备出版其著作《计算机程序设计艺术》时，他发现出版社提供的排版效果不太理想，当时的计算机排版技术也十分粗糙，这严重影响了他的著作的印刷质量，于是，他计划花费几个月的时间开发出一套更有效的文档排版系统，具体的开发目标是实现高质量的书籍排版。

由于克努斯博士此次在数学公式的排版上下足了功夫，就在他启动这项计划不久后，他收到了美国数学协会 (American Mathematical Society, AMS) 的邀请，克努斯博士在此次邀请中汇报的内容是“基于 TeX 排版，如何让计算机服务于数学”，这次汇报成功吸引了一大批数学家的目光。由于 TeX 在数学公式排版方面的优秀表现，比如数学公式的自动间距调整，TeX 后来摇身一变成为了书写数学公式的“利器”。

为了提升 TeX 的开发质量，克努斯博士悬赏奖励任何能够在 TeX 中发现程序漏洞的人，也就是我们一般认为的“找 bug”。每一个 bug 的奖励金额从 2.56 美元（16 进制的 100 美分）开始，以后每发现一个 bug，都会翻倍，直到 327.68 美元封顶。然而，克努斯博士从未因此而损失大笔金钱，因为 TeX 中的 bug 极少，而真正发现 bug 的人在获得支票后往往因其纪念价值而不愿兑现。

随着时间的推移，TeX 也派生出了很多优秀的软件，其中最著名的派生软件便是 LaTeX。另外，美国数学学会也发布了 TeX 版本的数学公式宏包，其中，以 ams 命

图 1.1: 克努斯博士，注：图片来源为克努斯博士的维基百科

名的宏包就有 *amsfonts*、*amsmath*、*amssymb* 等，这些宏包都可以在 LaTeX 上进行使用，在 LaTeX 上使用这些宏包可以编辑出各种数学公式。

参考资料

TeX 的维基百科介绍: https://zh.wikipedia.org/wiki/TeX.

1.2 引领浪潮的 LaTeX

1.2.1 LaTeX 的出现

LaTeX 是一款高质量的文档排版系统，LaTeX 在读法上一般发作 Lay-tek 或者 Lah-tek 的音，而不是大家普遍认为的 Lay-teks。LaTeX 的历史可以追溯到 1984 年，兰波特博士作为早期开发者在这一年发布了 LaTeX 的最初版本。事实上，LaTeX 完全是兰伯特博士的意外所得，他当年出于自己写书的需要，在早先发布的文档排版系统 TeX 基础上新增了一些特定的宏包，为了便于自己日后可以重复使用这些宏包，他将这些宏包进行规整，于是，便有了相应的标准宏包 (standard macro package)。谁曾想，正是这些不经意间开发出来的宏包，在经过后续封装和发布使用手册之后，形成了 LaTeX 的雏形。

在很长一段时间里，LaTeX 的版本其实没有多少大的更新，从技术层面来说，LaTeX 实在没有什么可供更新的地方了，它最初的面貌已趋近于完美且深入人心。LaTeX 的最初版本是由兰伯特博士于上世纪 80 年代初开发出来的，目前，广泛使用

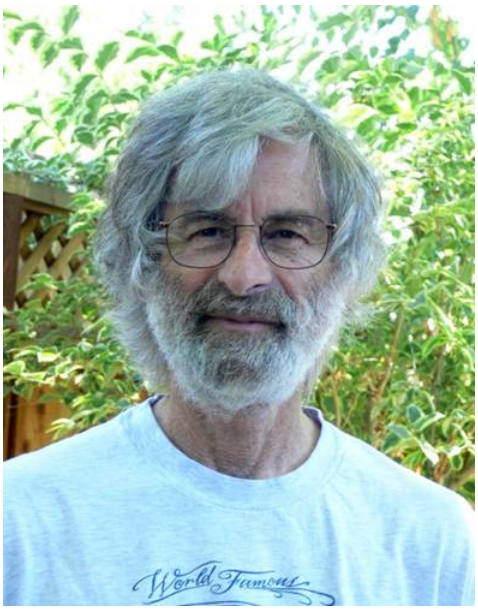

图 1.2: 兰伯特博士，注：图片来源为兰伯特博士的维基百科

的版本 LaTeX2e 是在 1994 年发布的，发布后一直没有大的更新，甚至发布后的首次更新出现在二十多年后的 2020 年。尽管 LaTeX2e 的后续版本更新工作早在上世纪九十年代初就已经开展了，但时至今日，新版的 LaTeX 仍未进入人们的视野。从开发者兰伯特博士的角度来看，开发 LaTeX 的目的是为了降低 TeX 的使用门槛、更容易地发挥 TeX 强大的排版功能，提供一款高质量、解释性强的计算机程序语言，所以 LaTeX 最初的风格就是精简，这也是为什么 LaTeX 在日后可供提升的地方不是很多的原因。

1.2.2 LaTeX 的特点

由于种种原因，时至今日，TeX 几乎淡出了人们的视线，不过我们现在依旧能看到：在使用 LaTeX 制作文档时，通常需要创建一个以.*tex* 为拓展名的文件。对于很多人来说，日常制作各类文档的首选可能是 Word 等软件，它简单好用、所写即所见，但当我们制作几十页甚至上百页的文档时，Word 的劣势就会展露无疑，因为我们需要投入大量的时间和精力来对文档内容进行排版。反观 LaTeX，它对文档的排版都是自动完成的，我们根本不需要像 Word 那样完全手动调整格式，另外，使用 LaTeX 插入各种图形、表格、公式、文献时，相应的索引出错的可能性也非常小，这些优点都是 Word 所无法比拟的。

在上个世纪 80 年代和 90 年代，LaTeX 的用户群体非常庞大，然而，在世纪之交，随着微软推出的一系列 Windows 操作系统快速发展，例如红极一时的 XP 系统，

相应的办公软件 Microsoft Office 也以其便捷性吸引了人们的视线，致使大量 LaTeX 用户转而使用 Microsoft Office。即便如此，时至今日，LaTeX 的用户群体依旧十分庞大，这主要得益于 LaTeX 强大的文档排版能力，虽然 LaTeX 复杂的语法结构、不容配置的编译环境让很多初学者望而却步，但 LaTeX 能让用户更专注于内容创作，而非锦上添花的“排版”，这一显著特点契合了人们对质量和效率的追求，使得 LaTeX 在文档排版、论文撰写等方面占有重要地位。在此基础上，具体来说，使得 LaTeX 历久弥新的关键可以归纳为以下五点：

1. LaTeX 是专门用于制作文档的计算机程序语言。在众多计算机程序语言中，LaTeX 可以制作排版连贯性极好的专业文档。

2. 独特的创作方式。尽管 LaTeX 沿用了 TeX 排版系统，但使用 LaTeX 制作文档时，内容创作和文档生成却是分开的，有需要的时候，我们可以预览创作的文档。因此，在创作的过程中，创作者不再像使用办公软件 Word 那样，既要关注创作内容，又要同步关注繁琐的排版和格式，使用 LaTeX 制作文档能在真正意义上让创作者专注于创作内容本身。更值得一提的是，当文档篇幅较大时，使用 LaTeX 无疑会让我们节省大量的时间和精力。

3. 简单的逻辑结构。使用 LaTeX 制作文档时，创作者可以通过一些非常简单的逻辑结构进行创作，如 chapter（章）、section（节）、table（表格）。因此，LaTeX 的使用门槛并不像真正的程序语言那么高。很多人或许在使用 LaTeX 的过程中都不会用到 for 等基本的循环语句。

4. 对数学公式以及特殊符号的支持程度。众所周知，LaTeX 在开发之初，是作为数学和计算机相关研究人员的创作工具，这类群体喜欢使用 LaTeX 的原因无外乎是 LaTeX 可以通过一些简单的代码生成复杂的数学表达式和特殊符号。

5. 编译以*.tex* 为拓展名的 LaTeX 文件后会得到一个 PDF 文档，PDF 文档不存在跨平台、兼容性等问题，可以在各种操作系统上打开。

当然，除了上述五点，实际上可能还有十分重要的一点，那就是 LaTeX 能够制作各类文档，从科技论文、技术报告、著作、学位论文、幻灯片甚至到科技绘图一应俱全，当然它也支持嵌入图片、绘制图形、设计表格、插入参考文献等。

从 LaTeX 的出现到当下，它已经形成了一套非常高效的文档制作环境：

- 文档类型 (document class)。文档类型是文档排版样式的基调，这些类型包括文章 (article)、报告 (report)、幻灯片 (beamer) 等，在.tex 文件中申明文档类型后，我们就可以开始文档创作了。

- 宏包 (package)。它是 LaTeX 中的重要辅助工具，也可以把它理解为一般意义上的工具包。在使用时，调用宏包的基本命令为*\usepackage{}*，举例来说，包含颜色命令的宏包为 *color*，其调用语句为*\usepackage{color}*。随着 LaTeX 的发展，越来越多的宏包被开发出来，这些宏包能满足特定的需求（如制表、插图、绘图），同时也能让 LaTeX 代码变得更加简洁，我们只需要用简单的*\usepackage{}*命令就能调用所我们需要用到的宏包。

- 模板 (template)。LaTeX 的发展催生了很多视觉和审美效果极好的模板，包括论文模板、幻灯片模板、报告模板甚至著作模板，这些模板在一定程度上能减少创作者在文档排版上的时间开销，也有很多学术刊物会给投稿作者提供相应的 LaTeX 模板。

通过对比 LaTeX 和 Word，我们还会看到：

- 第一，LaTeX 的.tex 源文件是无格式的，编译之后，根据特定的模板和指定的格式形成最终的 PDF 文档，因此，使用 LaTeX 制作各类文档能够很方便地切换模板和修改格式；

- 第二，LaTeX 对公式、图表以及文献引用的支持是 Word 所无法比拟的，尤为特殊的是，当文献数量达到上百篇时，在 Word 中修改参考文献可能是“牵一发而动全身”，费时耗力，而 LaTeX 根据已经整理好的*.bib* 文件可自动完成文献引用和生成。

1.2.3 LaTeX 编辑器

实际上，配置 LaTeX 环境包括两部分，即编译器和编辑器，对应的英文表达分别是 editor 和 complier，两者不是一回事。LaTeX 编译器又称为 LaTeX 编译工具，可根据系统安装相应的编译工具：

- Linux 系统：可安装 TeX Live，该编辑器拥有 LaTeX 编辑器；

- Mac OS 系统：可安装 Mac TeX，该编译器拥有完整的 TeX/LaTeX 环境和 LaTeX 编辑器；

- Windows 系统：可安装 MiKTeX 或 TeX Live，两者都拥有完整的 TeX/LaTeX 环境和 LaTeX 编辑器。

目前，我们可以接触到很多 LaTeX 编辑器，这些编辑器的界面大致有两部分组成，即 LaTeX 源码编译区域和 PDF 文档预览区域。前面也提到了几款 LaTeX 编译器，但如果想要提高 LaTeX 的使用体验，以下几款 LaTeX 编辑器比较受人推崇：

- TeXworks：这是 TeX Live 自带的一款轻量级编辑器。
- TeXstudio：这款编辑器集代码编译与文档预览于一身。
- WinEdt：这是 CTeX 自带的一款编辑器。
- VS Code：这是微软推出的一款免费文本编辑器，功能包括文本编辑、日常开发等。
- Atom：这是一款开源的跨平台编辑器（GitHub 网址为 https://github.com/atom/atom），支持多种程序语言。

1.3 应运而生的在线系统

1.3.1 LaTeX 在线系统的出现

上世纪 80 年代，LaTeX 作为一件新生事物，在发布之初便引起了人们极大的兴趣，虽然在制作文档方面拥有很多办公软件都无法比拟的强大优势，尤其在数学公式编写及高效排版上具有很大优势，但是由于其较高的使用门槛（使用计算机程序语言进行编译）和安装成本（本地安装需要花费大量的时间配置相应的环境），在很长一段时间里，LaTeX 主要用户都是科研工作者。然而，LaTeX 在线系统的出现已实实在在地改变了这一尴尬局面。

随着信息技术快速发展、互联网深度普及，人们的工作生活方式也在发生着很大改变，很多过去安装在本地的操作软件都被搬到了浏览器上，人们无须在个人计算机上安装各类办公软件就能进行办公，这带来了极大的便利。不过这类在线系统也存在一些先决条件，例如，出于计算资源方面的考虑，通常要求在线系统的类型不能是计算密集型，因为计算密集型的在线系统往往需要大量的计算资源作支撑。反观 LaTeX，尽管我们可以认为 LaTeX 是一种计算机程序语言，但实际上，其对计算资源的需求并不是很大。

在过去，受网速限制，使用线上系统几乎是一件难以想象的事。然而，在线系统的兴起并非空穴来风，一方面是目前的网速已经跟过去发生了质的变化，另一方面则是上网成本在急剧降低，互联网触手可及，已经成为人们日常生活和工作中不可或缺的一部分。以前，我们可能已经习惯了在本地计算机上安装和使用各类软件或者集成开发环境，不过以 LaTeX 为例，在本地计算机上安装的集成开发环境也有很多缺陷：

- 第一，我们需要为安装 LaTeX 编辑器腾出很大的存储空间；

- 第二，某些特定的宏包需要额外安装和配置，但安装过多宏包之后又会使 LaTeX 变得很臃肿，甚至是不友好；
- 第三，当我们在本地计算机使用 LaTeX 制作文档时，我们很难与合作者进行协同创作。

在这个背景下，一些成熟的 LaTeX 在线系统逐渐走进人们的视野，并受到很多用户的喜爱，其中，最为著名的 LaTeX 在线系统便是 *overleaf.com*。这些 LaTeX 在线系统不仅支持各种语言、各种拓展宏包等复杂的 LaTeX 环境，同时也支持实时编译和实时预览编译后的文档，就算是换一台电脑，也丝毫不会影响创作过程，创作完成之后，可以选择下载压缩文件包（如.zip），也可以只导出 PDF 文档，毫无疑问，这些人性化的设计都是为了让 LaTeX 更加便捷和高效。除此之外，现有的 LaTeX 在线系统还提供大量的 LaTeX 模板库，科技论文、毕业设计、幻灯片、海报、简历等参考模板一应俱全，就连 LaTeX 使用文档也数不胜数。

在线系统 overleaf

Overleaf 是一个初创的科技企业，它的主要业务是构建现代化协作创作工具，即 LaTeX 在线系统，旨在让科学研究变得更加便捷和高效。目前，Overleaf 已合并另一款著名的 LaTeX 在线系统 ShareLaTeX，在全球范围内拥有超过 600 万用户，这些用户大多是来自于高校和研究机构的研究人员、老师以及学生，只要打开网址 overleaf.com，用户无需在本地计算机配置 LaTeX 环境就可以创建各种 LaTeX 项目。

关于 Overleaf 的介绍可参考 https://www.overleaf.com/about。

1.3.2 LaTeX 在线系统的特点

以 Overleaf 为例，该 LaTeX 在线系统往往具备以下几点特征：

- 免费和开源。可以免费注册和使用，不用下载和安装 LaTeX 编辑器，这一点对于初学者来说无疑是非常友好的；
- 使用简单。不管是在计算机、手机还是其他终端上，我们只需要使用浏览器打开 overleaf.com 就可以开始创作，另外，由于 Overleaf 界面非常简洁，所以用户使用起来也非常便利；
- 支持实时在线编辑。有各类 LaTeX 插件，编辑功能十分完善，且具有实时编译和预览功能；

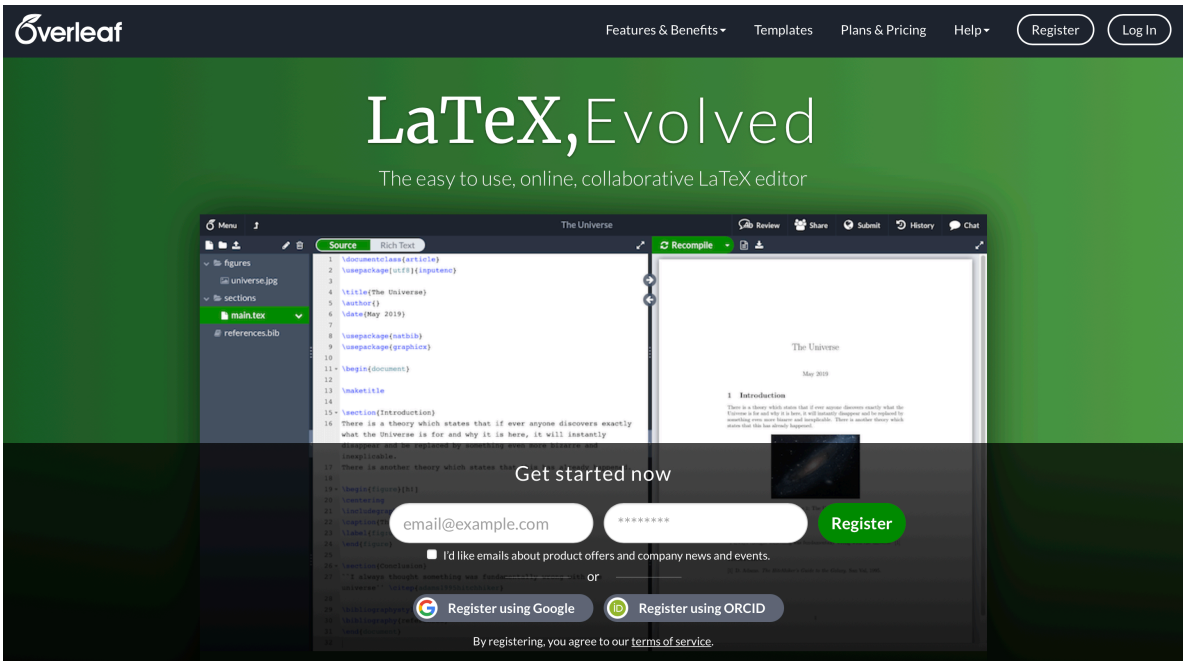

图 1.3: Overleaf 首页，图片来源于 Overleaf 官网

- 支持在线协作。创作文档时，我们可以将文档项目分享给合作者进行协作，Overleaf 支持实时编译，不会出现版本控制混乱等问题；
- 支持双向定位。可以在 LaTeX 代码与 PDF 文档内容之间进行双向定位；
- 提供丰富的模板库。Overleaf 有着非常庞大的模板库，不仅有正式的学术论文、学位论文和书籍的参考模板，还有很多美观的报告、简历、幻灯片模板。就论文写作来说，用户可以在 Overleaf 官网找到众多期刊的 LaTeX 模板，根据使用说明，用户很容易就能用于撰写自己的论文；
- 提供大量的帮助文档。LaTeX 提供了齐全的帮助文档，从 LaTeX 快速入门、基础操作到编译数学公式，应有尽有、一应俱全，且这些文档内容具有很强的实操性。

LaTeX 在线系统的出现大大降低了 LaTeX 的使用门槛，也为用户省去了繁琐的安装和配置过程。其实，LaTeX 在线系统的出现并非个例，很多办公软件为迎合用户需求和时代发展趋势，陆续转变了产品研发思路，包括微软在线 Office 系统、腾讯在线文档等在内的很多在线系统都走进了人们的视野，这些在线系统能够在线备份、满足人们对随时随地办公的需求，在确保便捷和高效的同时，在线和共享的理念正在潜移默化地影响着人们的办公模式。

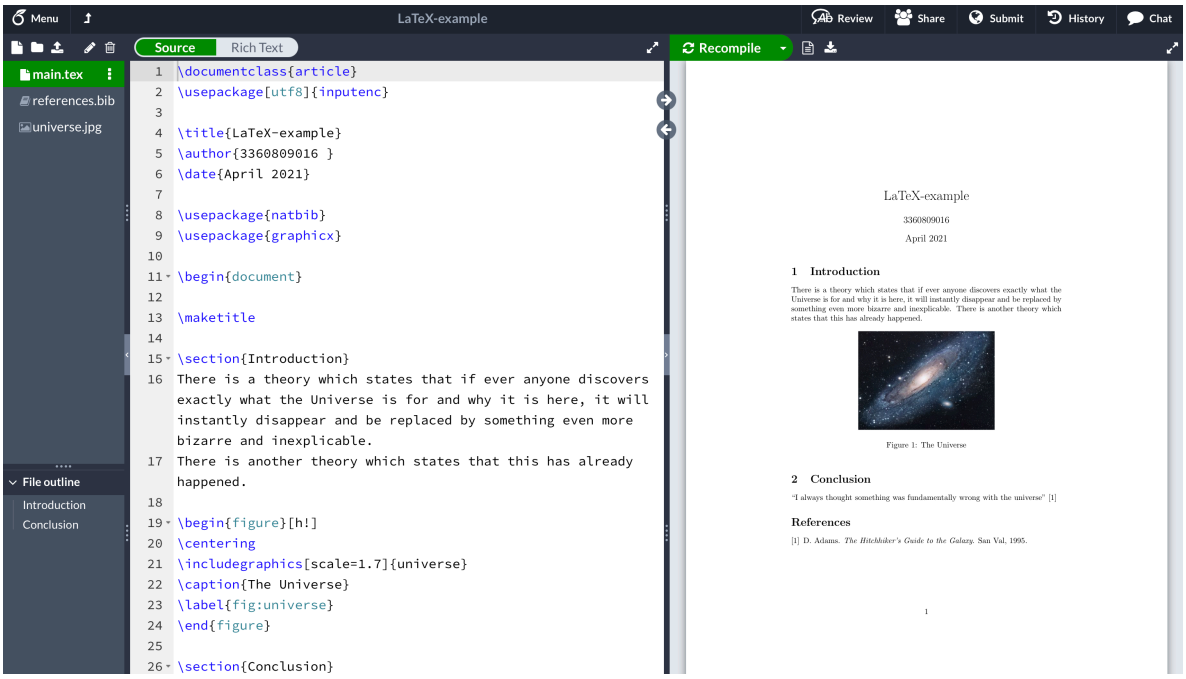

图 1.4: Overleaf 编辑器界面，主要由代码区域和文档预览区域组成

1.4 LaTeX 问答社区

在 LaTeX 刚被发布和推出的那个年代，相关资源如使用手册、教程等远没有今天这么丰富，同时，获取渠道也没有今天这么便捷。在互联网触手可及的今天，我们能通过一个浏览器访问到各种相关学习素材，遇到代码 bug，也能在一些专业的问答社区找到最佳的解决方案。毫无疑问，对于今天的我们来说，如何利用好互联网资源至关重要。

1.4.1 问答社区的介绍

对于从事计算机相关的技术人员来说，专业的技术问答社区往往是不可多得的资源，它能帮助很多技术人员提升个人编程能力、学习新技术，同时解决一些技术困扰，例如解决编程时遇到的 bug。对于各类计算机程序语言而言，*Stack Exchange* 作为一个著名的技术问答社区，覆盖了大量编程相关的技术问题和优质回答，就算是一些较为细致的问题，我们往往都能找到想要的答案。Stack Exchange 问答社区按计算机程序语言类型进行划分，我们所关心的 LaTeX 问答通常被分配在 TeX Stack Exchange 社区（网址为 https://tex.stackexchange.com/）。截至目前，TeX Stack Exchange 已经覆盖了关于 TeX、LaTeX 以及其他排版系统的用户在使用中遇到的诸多问题，并且这些问题多与 LaTeX 相关。

Stack Exchange

网址为 https://stackexchange.com/，它与 Stack Overflow 问答社区在全球范围内拥有广泛的用户群体

除了 Stack Exchange 这种涵盖了多种计算机程序语言的技术问答社区，LaTeX forum 社区（https://latex.org/forum/）是一个专门面向 LaTeX 的技术交流平台，它拥有非常活跃的用户群体和丰富的问答资源，并拥有超过 10 万篇分门别类的技术帖子，我们可以根据浏览量从该平台一览高频访问问题。

实际上，不管是 LaTeX 初学者还是高级用户，在遇到 LaTeX 使用问题时，去问答社区寻找解决方案都是一种非常有效的方式。TeX Stack Exchange 社区非常活跃，每天都会有大量关于 LaTeX 的问题和回答，且每个问题下面的回答都会根据用户的认可度进行排序，回答一般比较细致。

1.4.2 高频访问问题

顾名思义，高频访问问题是指访问量较高的问题。LaTeX forum 社区已经将众多问答帖子进行了分门别类整理，针对某一特定话题，展开内容即可看到各类问题的访问情况。

1.5 关于 LaTeX 的开源项目

实际上，不管是 TeX 还是 LaTeX，它们作为计算机程序语言是完全开源的，我们可以轻松获取到 TeX 和 LaTeX 的源代码。近些年来，互联网催生了一些用户体验以及评价很好的开源社区，比如当下非常受欢迎的开源平台 GitHub，它们都在实实在在地影响着我们对于计算机技术开放的态度。开源社区有很多优质的“仓库”(repository)，这些仓库会提供诸如源代码、工具、案例甚至使用文档。与 LaTeX 在线系统相似的是，这些开源社区非常支持协作功能，在有一些优质的仓库中，我们或许能看到几十甚至上百个贡献者，“汇聚集体智慧”也是这些开源社区广受开发者青睐的一个重要原因。

1.5.1 开源社区 GitHub

GitHub 是一个面向开源项目及私有项目的云端托管平台，官方网址为 http://github.com/，旨在为开发者储存、管理代码以及控制代码的更新，只支持 Git 作为唯一的版本库格式进行托管。GitHub 已经走进人们的视野已经有十几年了，它于 2008 年 4 月 10 日

正式上线，除了 Git 代码仓库托管和基本的网页管理界面以外，还提供了在线文件编辑器、版本控制、协作报表等功能。截至 2021 年初，GitHub 在全球范围内拥有的注册用户已经超过 5000 万，托管项目的总量更是超过 1 亿，是很多开发者远程协作的重要工具。2018 年 6 月，GitHub 被微软以 75 亿美元的价格收购。

GitHub 本着开源、共享、协作的理念正影响着开发社区生态，吸引了包括微软和谷歌等国外技术巨头在这里进行开源，另外，我们也能在 GitHub 上看到一些国内互联网企业的身影。实际上，GitHub 开源社区的项目类型及展现形式也异常丰富，既有大量的应用型项目，也有很多研究探索型项目。

为了方便开发者追踪整个开源社区的开发动态，GitHub 根据计算机程序语言类型提供了趋势分析，这个功能称为 GitHub trending，它会定期更新趋势榜、帮助开发者追踪当下较为流行的 GitHub 仓库。

GitHub 开源社区提供了很多关于 LaTeX 的优质开源项目，这些项目使得 LaTeX 的用户群体越来越广泛。实际上，LaTeX 在线系统 Overleaf 也是开源的，它托管在 GitHub 上，网址为 https://github.com/overleaf/overleaf。

为便于读者认识和掌握 LaTeX，以下简单归纳一下关于 LaTeX 的各类开源项目。需要注意的是，优质的开源项目是非常多的，限于篇幅，这里只列举一部分，感兴趣的读者不妨在 GitHub 平台上探寻更多优质项目。

1.5.2 学位论文 LaTeX 模板

在 GitHub 开源社区中，我们可以找到很多国内外高校官方和非官方的学位论文 LaTeX 模板，学位论文类型包括本科毕业设计、硕士学位论文和博士学位论文等。以下将列举一部分国内高校的学位论文 LaTeX 模板开源项目：

- 开源项目 https://github.com/tuna/thuthesis 提供了清华大学学位论文 LaTeX 模板，模板库涵盖本科综合论文训练、硕士论文、博士论文以及博士后出站报告，截至目前，已在 GitHub 上获得超过 3000 次标星。

- 开源项目 https://github.com/mohuangrui/ucasthesis 提供了中国科学院大学学位论文 LaTeX 模板，包括本科、硕士和博士学位论文模板，截至目前，已在 GitHub 上获得超过 2000 次标星。

- 开源项目 https://github.com/mohuangrui/ucasthesis 提供了中国科学院大学学位论文 LaTeX 模板，包括本科、硕士和博士学位论文模板，截至目前，已在 GitHub 上获得超过 2000 次标星。

- 开源项目 https://github.com/ustctug/ustcthesis 提供了中国科学技术大学学位

论文 LaTeX 模板，包括本科毕业论文（设计）和研究生学位论文，截至目前，已在 GitHub 上获得近 1000 次标星。

- 开源项目 https://github.com/TheNetAdmin/zjuthesis 提供了浙江大学学位论文 LaTeX 模板，包含本科、硕士和博士学位论文模板，也提供了英文版的硕博士学位论文模板，截至目前，已在 GitHub 上获得近 1000 次标星。
- 开源项目 https://github.com/dustincys/hithesis 提供了哈尔滨工业大学学位论文 LaTeX 模板，包括本科、硕士、博士开题、中期和毕业论文，也提供了博后出站报告和英文毕业论文格式，截至目前，已在 GitHub 上获得近 1000 次标星。
- 开源项目 https://github.com/x-magus/ThesisUESTC 提供了电子科技大学毕业论文 LaTeX 模板，截至目前，已在 GitHub 上获得超过 500 次标星。

上述开源项目提供的学位论文模板绝大多数支持在 LaTeX 在线系统 overleaf.com 上直接进行编辑和使用。

1.5.3 LaTeX 绘制图形

目前，LaTeX 有很多关于绘制图形方面的宏包，这些宏包会提供图形的基本元素、颜色等。由于 LaTeX 绘制出来的图形可以以 PDF 文档保存，所以一般不会出现分辨率不足等问题。这里对 LaTeX 绘制图形相关的一些开源项目稍作整理：

- 工具类:
  - *BayesNet*：用于绘制贝叶斯网络、图模型和因素图形的 TikZ 库。开源网址为 https://github.com/jluttine/tikz-bayesnet。一般而言，使用 BayesNet 绘制特定图形时，需要使用以下两行命令调用该库：*\usepackage{tikz}* 和 *\usetikzlibrary{bayesnet}*
  - *matlab2tikz*：将由 Matlab 代码绘制的图形转换成 PGFPlots 图形。开源网址为 https://github.com/matlab2tikz/matlab2tikz。
  - *tikzplotlib*：将由 Python 绘图工具 matplotlib 绘制的图形转换成 PGFPlots 图形。开源网址为 https://github.com/nschloe/tikzplotlib。
  - *svg2tikz*：将 SVG 图形转换成 TikZ 代码。开源网址为 https://github.com/xyz2tex/svg2tikz。
- 实例类:
  - *TeXample*：该项目提供了大量的 TikZ 绘图实例，开源网址为 https://texample.net/tikz/。

- 开源项目 https://github.com/walmes/Tikz 提供了大约 200 个关于统计学的 TikZ 绘图实例。
- 开源项目 https://github.com/MartinThoma/LaTeX-examples/tree/master/tikz 提供了大约 350 个 TikZ 绘图实例。
- 开源项目 https://github.com/PetarV-/TikZ 提供了大量的 Tikz 绘图实例，囊括了各类神经网络模型的图形。
- TeX Stack Exchange 社区中提供的可视化效果很好的 TikZ 科技绘图实例，网址为 https://tex.stackexchange.com/questions/158668/nice-scientific-pictures-show-off。
- 开源项目 https://github.com/FriendlyUser/LatexDiagrams 提供了大量的 TikZ 绘图实例，包括流程图、图模型。
- 开源项目 https://github.com/alemelis/tikz_drawings 提供了一些 TikZ 绘图实例。

1.5.4 LaTeX 制作简历

LaTeX 可用于制作各类文档，这也包括简历。

- 开源项目 https://github.com/sb2nov/resume 提供了简历模板。
- 开源项目 https://github.com/xdanaux/moderncv 提供了 moderncv 简历样式。
- 开源项目 https://github.com/jankapunkt/latexcv 提供了一些美观大方且容易使用的简历模板，支持包括中文在内的多种语言编译。
- 开源项目 https://github.com/hijiangtao/resume 提供了中文简历的模板。

1.5.5 LaTeX 制作幻灯片

- 开源项目 https://github.com/matze/mtheme 提供了现代风格主题的幻灯片模板。
- 开源项目 https://github.com/wzpan/BeamerStyleSlides 提供了很丰富的模板，包含了 PowerPoint 和 Keynote 两套格式。
- 开源项目 https://github.com/XiangyunHuang/awesome-beamers 提供了多种制作幻灯片的模版，同时也支持中文制作。

- 开源项目 https://github.com/josephwright/beamer 提供了多种风格的幻灯片模版。

1.5.6 LaTeX 制作海报

- 开源项目 https://github.com/jkjaer/aauLatexTemplates 提供了 beamer 海报模板。

- 开源项目 https://github.com/mloesch/baposter 提供了简洁的海报模版。

- 开源项目 https://github.com/HolgerGerhardt/TeXTemplates 提供了适合做学术报告的海报模版，同时也包含幻灯片模版。

1.5.7 推荐指导本书的开源项目

开源网址为 https://github.com/xinychen/latex-cookbook。同时该作者还有图形绘制和幻灯片制作的开源项目:

- LaTeX 图形绘制：https://github.com/xinychen/awesome-latex-drawing 提供大量绘图实例，实例包括贝叶斯网络、框架图、流程图、机器学习模型示意图。

- LaTeX 幻灯片制作：https://github.com/xinychen/awesome-beamer 提供大量使用 Beamer 制作幻灯片的实例。

参考资料

开源项目 https://github.com/xiaohanyu/awesome-tikz。

1.6 LaTeX 制作中文文档

LaTeX 最初只提供英文的编译环境，随着其在文档编辑领域的优势越来越深入人心，更多国家的学者希望用到这一工具，因此出现了许多支持各国语言的工具包或编写环境，作为全世界使用人数最多的语言，LaTeX 有 *CJKutf8* 宏包、*CTEX* 宏包等多种方式可以实现中文文档编辑，均能在开源网站 Overleaf 中调用并实现中文文档制作。

1.6.1 使用 CJKutf8 宏包

CJKutf8 宏包提供了两种中文简体字体制作中文文档：使用 *\begin{CJK}{UTF8}{gbsn}* 和 *\end{CJK}* 环境将以宋体（gbsn）制作文档，而使用 *\begin{CJK}{UTF8}{gkai}* 和 *\end{CJK}* 环境则以楷体（gkai）制作文档内容。在默认的 pdfLaTeX 编译环境中即可得到文档编译结果。

【例】使用 *CJKutf8* 红包制作中文文档：

```
\documentclass{article}
\usepackage{CJKutf8}

\begin{document}

\begin{CJK}{UTF8}{gbsn} %字体是gbsn
你好，LaTeX！。
\end{CJK}

\end{document}
```

Listing 1.1: CJKutf8 示例

1.6.2 使用 CTEX 宏包

CJKutf8 宏包只提供了两种字体，可选择的余地太小。如果想要使用更丰富的字体编辑 Latex 中文文档，可以调用 *CTEX* 宏包、并设置 UTF8 选项使其支持 utf-8 编码。在 Overleaf 中使用 CTEX 宏包时，需要先将编译环境从 *pdfLaTeX* 调整为 *XeLaTeX*。【例】使用 *CTEX* 宏包制作中文文档：

```
\documentclass[UTF8]{ctexart}

\begin{document}

{\kaishu 你好，LaTeX！（楷体）}

{\songti 你好，LaTeX！（宋体）}

{\heiti 你好，LaTeX！（黑体）}

```

```
{\fangsong 你好，LaTeX! （仿宋）}。

\end{document}
```

Listing 1.2: CTEX 示例

目前在 Overleaf 上已经出现了许多中文文档 LaTeX 模板，除了一些学位论文模板，一些中文学术期刊如《计算机学报》也提供了科技论文的 LaTeX 模板。

期刊模板：

- 《中国科学：信息科学》https://www.overleaf.com/project/5e99712a0916c900018d11af
- 《中国科学：信息科学》https://www.overleaf.com/project/5e99712a0916c900018d11af

中文学位论文模板：

- 《浙江大学学位论文模板》https://www.overleaf.com/project/610fa05007d0073d5405a04f
- 《武汉大学博士学位论文模板》https://www.overleaf.com/project/610fa09e07d007fa5605a1e9
- 《武汉大学博士学位论文模板》https://www.overleaf.com/project/610fa09e07d007fa5605a1e9
- 《南京大学研究生毕业论文模板》https://www.overleaf.com/project/610fa1d007d00704c305a3eb

另外，开源项目 https://github.com/MCG-NKU/NSFC-LaTex 提供了国家自然科学基金申报书的 LaTeX 模板。

第二章　LaTeX 基础介绍

2.1　导言

LaTeX 作为专门用于制作文档的计算机程序语言，其语法规则是由命令 (command) 和环境 (environment) 构成，基于一些基本命令，如\usepackage{}，我们能调用制作文档所需的宏包。一般而言，在制作文档时，LaTeX 的代码结构分为前导 (preamble) 和主体 (body) 两部分，前导部分的代码主要用于申明文档类型、排版样式、所使用的宏包等；主体部分主要用于确定标题、章节、目录等文章结构及创作文档内容。

掌握一些常用命令之后，我们便可上手编辑简单的文档。从功能上来说，LaTeX 能满足人们对文档制作多样性的要求。我们可以用 LaTeX 制作各类文档，包括科技文档、技术报告、学位论文、书籍著作、幻灯片、个人简历、信件以及海报。不同的文档类型在文档大小、排版、章节样式方面略有不同，所以在使用 LaTeX 制作文档时，我们首先需要对文档类型进行申明，然后再进行文档内容的创作。

本章节要介绍的内容主要包括：Latex 语法规则、Latex 代码结构、文档类型的介绍、简单文档的制作以及全局格式的设置。

2.2　LaTeX 语法规则

对于任何一种计算机程序语言，严格的语法规则都是让程序语言实现功能的保障。不同于一般性的计算机程序语言，LaTeX 的语法规则非常简单，是由命令和环境构成。在 LaTeX 中，不管是命令还是环境，都离不开计算机符号反斜线\，例如调用宏包会用到 \usepackage{宏包名称}，这里的花括号 {} 也是非常常用的一个符号。在调用宏包后，命令的用法都是由宏包约定的。

2.2.1 命令

LaTeX 中有很多命令，它们用法大同小异，功能却千差万别。既有申明文档类型的命令，如\documentclass{article}，也有输入特殊符号的命令，例如\copyright。一般而言，LaTeX 中的命令通常由三部分组成\命令名 [可省略参数]{不可省略参数}，具有以下特点：

- 通常都是以反斜线作为开始，通过紧跟的既定字符（命令名）实现相应的功能，例如\LaTeX 和\copyright 可生成特殊字符。
- 一些命令需要申明一些参数,通过大括号中的不可省略参数进行申明,例如\color{blue}命令中需要申明具体的颜色名称。
- 一些命令包括一些选项和一些参数一般情况下是默认的，有需要时可以通过中括号中的可省略参数进行调整，例如在\documentclass[a4paper]{article} 中，中括号 [] 作为额外的选项，既可以自行设置，也可以选择默认设置。
- 有些命令可以用反斜线作为终止，例如\copyright。

2.2.2 环境

这里所说的“环境”是指编译环境，使用 LaTeX 时，当我们想要编译出期望的效果，例如列表、插入图形、制作表格，我们就需要用到一些环境。举例来说，我们可以通过

```
1    \begin{itemize}
2       \item item 1 % 条目1
3       \item item 2 % 条目2
4    \end{itemize}
```

制作一个简单的无序列表，这里的 *itemize* 表示无序列表环境，*\begin{}* 和 *\end{}* 表示环境的初始化和结束。当然，这些环境并非“一成不变”，我们也可以设置一些参数，从而改变编译之后的文档效果，例如，可以通过

```
1    \begin{spacing}{1.3}
2       paragraph 1 % 第1段
3
4       paragraph 2 % 第2段
5    \end{spacing}
```

将两段话之间的行间距设置为 1.3 倍。

【例】使用基本命令\documentclassarticle 和文档环境\begin{document} 和\end{document} 创建一个简单文档。

```
\documentclass{article}

\begin{document}

Hello, LaTeXers! This is our first LaTeX document.

\end{document}
```

2.2.3 宏包

宏包是 LaTeX 的重要组成部分，用来扩展和增强 LaTeX 的功能，是支撑 LaTeX 实现一系列复杂文档编辑和排版的关键所在。宏包与 LaTeX 的关系和浏览器插件与浏览器的关系类似，通过调用不同的宏包可以实现一些复杂排版功能，例如插入表格、插入公式和特殊符号、插入程序源代码、设置文档样式等。一个宏包通常会提供一组 LaTeX 命令，有些特殊命令只能在调用宏包后使用。在 LaTeX 中，调用宏包的形式大同小异，如果想要使用某一特定宏包，最简单的办法就是用\usepackage{宏包名} 命令对相应宏包进行调用。

【例】使用\usepackage{color} 命令调用颜色宏包调整文本字体颜色。

```
\documentclass{article}
\usepackage{color} % 调用颜色宏包
\begin{document}
\textcolor[rgb]{1,0,0}{Hello, LaTeXers! This is our first LaTeX document.}
\end{document}
```

2.3 LaTeX 代码结构

在使用 LaTeX 进行文档编辑时，我们通常在拓展名为*.tex* 的源文件中书写代码，然后通过编译，生成一个 PDF 格式的文档。LaTeX 源文件的代码结构主要包含两个

部分，即前导代码（preamble）和主体代码（body），其结构示例如下：

```
\documentclass[]{}

..... % 前导代码 (preamble)

\begin{document}

..... % 主体代码 (body)

\end{document}
```

2.3.1 前导代码

前导代码是指从源文件第一行代码到\begin{document} 之间的所有命令语句，一般为 LaTeX 代码的第一部分。在前导代码部分，既可设置文档类型全局参数，包括字体大小、纸张大小、文字分栏、单双面打印设置等，也可声明主体代码中需要用到的宏包，如插入图形、新增表格所要用到的宏包。当全局格式没有特殊申明时，前导代码中的文档类型申明语句可以简写成\documentclass{B}，其中，位置 B 的作用在于申明文档类型，如 article（常规文档）、book（书籍）、report（报告）、beamer（幻灯片）等。

下面以常规文档（article）为例，简要介绍各常用全局参数的设置方式。

字体大小

article 类型文档字体大小默认为 *10pt*，可在\documentclass[可省参数]{article} 的中括号中根据需要设置成 11pt 和 12pt。

通常，我们按照\documentclass[fontsize = 12pt]{article} 这样的形式设置文档参数，有时候，为了方便起见，可以把文档类型申明做一定的简写，如\documentclass[12pt]{article}。需要注意的是，字体大小设置中的基本单位 pt 是英文单词 point 的缩写，是一个物理长度单位，**指的是 72 分之一英寸，即 1pt 等于 1/72 英寸**。

纸张大小

article 类型文档纸张大小默认为 *letterpaper*，同样可在\documentclass [可省参数]{article} 的中括号中根据需要设置成 *a4paper*、*a5paper*、*b5paper*、*legalpaper* 和 *executivepaper*。

文字分栏

article 类型文档的文字分栏默认为 *onecolumn*（不分栏），也可以使用 *twocolumn* 参数设置为 twocolumn（两栏）。

单双面打印设置

article 类型文档打印时默认单面打印，同样可以使用\documentclass[可省参数]{article} 中括号中的可选参数，通过添加 *twoside* 参数进行双面打印的设置。

文档属性设置小结

基于上面介绍可供调整的参数，我们可以进行任意无序组合，例如\documentclass [a4paper, 11pt, twoside]{article} 对应着文档类型为 article、纸张大小为 A4、字体大小为 11pt 的双面文档。

2.3.2 主体代码

主体代码为\begin{document} 及\end{document} 之间所有的命令语句和文本，一般由文档的创作内容构成，按照一般书写顺序，主要包含文档标题、目录、章节、图表及具体文字内容等，通常配合一些基本命令的使用，来形成我们期望的文档。下面给出了一个简单例子，让读者对如何制作一个包含标题、章节及其文字内容的简单文档能有一个比较粗略的认识，更具体的语法命令我们将在后续章节中依次介绍。

```
1    \documentclass{article}
2    \title{LaTeX cook-book}
3
4    \begin{document}
5    \maketitle
6    \section{Introduction}
7
8    Hello, LaTeXers! This is our first LaTeX document.
9
10   \end{document}
```

2.4 文档类型介绍

在 LaTeX 代码结构中，申明文档类型往往是制作文档的第一步，也是最基本的一步。事实上，不同文档类型对应的文档样式略有不同，但制作不同类型的文档时，LaTeX 中的绝大多数命令和环境却是通用的，完成对文档内容的创作后，使用文档类型的申明语句可以让我们在不同类型的文档间切换自如。

2.4.1 基本介绍

对 LaTeX 熟悉的读者会知道，LaTeX 实际上支持非常多的文档类型，例如，撰写科技论文会用到的 *article*（常规文档）类型、制作演示文稿会用到的 *beamer*（幻灯片）类型。在众多文档类型中，常见的文档类型包括 *article*（常规文档）、*report*（报告）、*book*（书籍）、*beamer*（幻灯片）等，如果使用支持中文编译的 *ctex* 文档类型，还会有 *ctexart*（中文常规文档）、*ctexrep*（中文报告）、*ctexbook*（中文书籍）等，文档类型会直接决定整个文档的样式和风格。

使用 LaTeX 制作文档时，申明文档类型是作为前导代码，其一般格式为：

```
1    \documentclass[A]{B}
```

在这一申明语句中，位置 A 的作用主要是设置控制全文的文档参数，我们可以调整全文的字体大小、纸张大小、分栏设置等，因为各种文档类型都有一整套的默认参数，所以一般情况可以省略掉位置 A。在位置 B，我们需要键入特定的文档类型，例如，\documentclass[a4paper, 12pt]{article} 即表示申明一个纸张大小为 A4、字体大小为 12pt 的常规文档。

以下将逐一介绍比较常用的三种文档类型，包括 article（常规文档）、report（报告）、book（书籍）。其中，report 和 book 这两种文档类型的文档结构是一致的，可以使用的结构命令有\part{}、\chapter{}、\section{}、\subsection{}、\subsubsection{}、\paragraph{}、\subparagraph{}，举例来说，包含颜色命令的而 article 文档类型中除了没有\chapter{} 这一结构命令之外，其他都与 report 和 book 文档类型是一样的。

- article 是 LaTeX 制作文档时最为常用的一种文档类型，撰写科技论文往往会用到 article 文档类型。
- report 主要是面向撰写各类技术报告的文档类型。
- book 是用于制作书籍等出版物的文档类型。

2.5 简单文档的制作

LaTeX 不但适合制作篇幅较大的文档，在制作篇幅较小的文档比如手稿、作业等时也十分方便。在 LaTeX 的各类文档中，最为常用的文档类型为 article (文章)，以下将介绍如何制作一个简单文档。

2.5.1 制作封面

添加标题、日期、作者信息一般是在\begin{document} 之前，格式如下：

```
% 输入空格表示空的
\title{标题}
\author{作者名字}
\date{日期} % 如果不设置则会自动设置为编译时的时间，如果不想展示日期则使用\data{}
```

如果要显示添加的相关信息，需要在\begin{document} 之后使用\maketitle 命令。

2.5.2 开始创建文档

在 LaTeX 中，以\begin{document} 命令为分界线，该命令之前的代码都统称为前导代码，这些代码能设置全局参数。位于\begin{document} 和 \end{document} 之间的代码被视为主体代码，我们所创作文档的具体内容也都是放在这两个命令之间。

设置章节

文档的章节是文档逻辑关系的重要体现，无论是中文论文还是英文论文都会有严谨的格式，章、节、段分明。在 LaTeX 中，不同的文档类设置章节的命令有些许差别，\chapter 命令只在 book、report 两个文档类中有定义，article 类型中设置章节可以通过\section{name} 及\subsection{name} 等简单的命令进行实现。

段落

段落是文章的基础，在 LaTeX 中，可以之间在文档中间键入空行文本作为段落，也可以使用 \paragraph{name} 和\subparagraph{name} 插入带标题的段落和亚段落。

生成目录

在 LaTeX 中,我们可以通过一行简单的命令便可以生成文档的目录,即\tableofcontents。命令放在哪里，就会在哪里自动创建一个目录。默认情况下，该命令会根据用户定义的篇章节标题生成文档目录。目录中包含\subsubsection 及其更高层次的结构标题，而段落和子段信息则不会出现在文档目录中。注意如果有带 * 号的章节命令，则该章节标题也不会出现在目录中。如果想让文档正文内容与目录不在同一页，可在\tableofcontents 命令后使用\newpage 命令或者\clearpage 命令。

类似对章节编号深度的设置,我们通过调用计数器命令\mintinline{tex}{\setcounter}也可以指定目录层次深度。例如:

- \setcounter{tocdepth}{0} 目录层次仅包括 part
- \setcounter{tocdepth}{1} 目录层次深入到 section
- \setcounter{tocdepth}{2} 目录层次深入到 subsection
- \setcounter{tocdepth}{3} 目录层次深入到 subsubsection，默认值

除此之外,我们还可以在章节前面添加\addtocontents{toc}{\setcounter{tocdepth}{}}命令对每个章节设置不同深度的目录。另外还有一些其他的目录格式调整命令，如果我们想让创建的目录在文档中独占一页，只需要在目录生成命令前后添加\newpage;如果我们需要让目录页面不带有全文格式,只需要在生成目录命令后面加上\thispagestyle{empty}命令；如果我们想设置目录页之后设置页码为 1，则需要在生成目录命令后面加上\setcounter{page}{1} 命令。

如果我们想要创建图目录或表目录，分别使用\listoffigures、\listoftables 命令即可，与创建章节目录的过程类似，这两个命令会根据文档中图表的标题产生图表目录，但不同之处在于，图目录或表目录中所有标题均属于同一层次。

【例】基于当前已掌握的知识的一个简单的 article:

```
1 \documentclass[a4paper,utf8,12pt]{ctexart}
2
3 %% 导言区
4 \usepackage{hyperref} % 超链接
5 \hypersetup{hidelinks} % 超链接隐藏默认的红框
6 \title{\LaTeX Cookbook} % 设置标题
7 \author{XXX} % 作者
8 \date{2021/12/19} % 设置日期,不设置默认生成编译当天日期
9
```

```
\tableofcontents % 生成目录

%% 文档主体
\begin{document}
\section*{摘要} % 默认不生产目录; 带*号的不设置编号

\section{XX背景介绍} % 第一节

\section{XX原理} % 第二节
\subsection{XX原理论证} % 2.1 小节
\subsection{基于XX的优化} % 2.2 小节
\paragraph{一个段落...}

\paragraph{另一个段落...}

段落也可以使用空行生成,比如这是第三个段落...

这是第四个段落...

\end{document}
```

2.6 一些基本命令

当我们运用 LateX 进行文档编辑时，需要用到一些基本命令。

2.6.1 全局格式设置

在前面，我们介绍了一些全局设置的命令在申明文档类型时可以进行的一些全局参数设置，使用方法为在\documentclass[可省参数]{article} 的中括号中根据需要进行参数设置，然而有些全局设置需要用到一些其他方法进行调整。例如，纸张方向、页边距等需要调用宏包进行参数调整。同样地，我们以 article 类型文档进行举例说明。

纸张方向

article 类型文档的纸张方向默认为 *portrait*（纵向），也可以设置成 *landscape*（横向）。在文档中可以调用 *lscape* 宏包中的\begin{landscape} \end{landscape} 环境将默认的纵向文档调整为横向。

How to change certain pages into landscape/portrait mode

https://tex.stackexchange.com/q/337/227605

页边距

article 类型文档的页边距可以通过调用 *geometry* 宏包进行调整

```
\usepackage{geometry} % 使用页面设置宏包
\geometry{left=3.0cm,right=3.0cm,top=2.5cm,bottom=2.5cm} % 设置页边距
```

章节标题的字体格式

article 类型文档的章节标题的字体格式可以通过调用 *sectsty* 宏包进行调整。一些设置例举：

```
\usepackage{sectsty}
\sectionfont{\fontfamily{phv}\fontseries{b}\fontsize{11pt}{20pt}\selectfont} % 一级标题字体格式设置
\subsectionfont{\fontfamily{phv}\fontseries{b}\fontsize{11pt}{20pt}\selectfont} % 二级标题字体格式设置
\subsubsectionfont{\fontfamily{phv}\fontseries{b}\fontsize{11pt}{20pt}\selectfont} % 三级标题字体格式设置
```

图、表、公式格式全局设置

当我们需要批量设置图、表及公式的格式时，可以通过调用 caption 宏包进行全局设置。例：

```
\usepackage[labelfont=bf,labelsep=period,font={bf,sf,normalsize}]{caption}
```

Changing/Defining Fonts for an Entire Document

https://tex.stackexchange.com/q/337/227605

自定义命令全局设置

有时，我们需要也可以使用一些自定义命令来更改全局设置。例如在更改整个文档的字体格式时，我们也可以使用：

```
\renewcommand{\sfdefault}{\fontencoding{T1}\fontfamily{phv}\selectfont}
\renewcommand{\familydefault}{\sfdefault}
```

等命令。

在更改目录标题时，我们可以使用：

```
\renewcommand{\contentsname}{new name of Contents}
```