## What is a .bib file?

A .bib file is a plain‑text **bibliography database** in *BibTeX* format.  
Each entry (article, book, thesis, web page…) is declared once, for example:

@article{black1973pricing,

author = {Black, Fischer and Scholes, Myron},

title = {The Pricing of Options and Corporate Liabilities},

journal = {Journal of Political Economy},

year = {1973},

volume = {81},

number = {3},

pages = {637--654}

}

* In the LaTeX body you **cite** with \parencite{black1973pricing} (or \cite, \footcite, etc.).
* The back‑end program (bibtex or biber) reads the .bib, formats the citations and the reference list according to the style you chose (apa, IEEEtran, *etc.*).

Advantages:

* One canonical record per source (no copy‑paste duplication).
* You can swap citation styles without rewriting the document.
* Many academic databases let you export BibTeX directly.

## preamble.tex vs. mystyle.sty

| **Aspect** | **preamble.tex (Input File)** | **mystyle.sty (Package File)** |
| --- | --- | --- |
| How you include it | \input{preamble} | \usepackage{mystyle} |
| Typical contents | Packages, macros **specific to one project** | Re‑usable macros/environments shared across multiple projects |
| Location | Lives alongside main.tex | Lives in project root or in local TeX tree |
| Distribution | Not intended for CTAN upload | Could be published and versioned as a package |

**Rule of thumb**

* Keep project‑local settings in preamble.tex (fast, zero overhead).
* If you find yourself reusing the same macros in different documents, refactor them into mystyle.sty and load it with \usepackage.

For a single handbook, preamble.tex is perfectly fine.

## What do \mainmatter, \part, \backmatter, \printbibliography do?

\mainmatter

\part{Foundations of Financial Markets}\label{part:foundations}

...

\backmatter

\printbibliography

**\mainmatter** (book/report classes)

* Resets page numbering to 1 in **arabic numerals**.
* Switches chapter/section counters to their “main” style.
* Place it after your front‑matter (title page, TOC, glossaries).

**\part{...}**

* Top‑level division above \chapter.
* Starts on a right‑hand (recto) page and gets its own typeset “Part I” page.
* The \label{...} lets you later refer to it via \cref{part:foundations} or \nameref.

**\backmatter**

* Turns OFF chapter numbering (useful for appendices, bibliography, index).
* Page numbering continues but headings change style (often smaller).

**\printbibliography** (from biblatex)

* Generates the reference list, applying the selected citation style.
* Place it in the back‑matter so it is not counted as a numbered chapter.

## Minted – syntax-highlighted code listings

minted is a LaTeX package that delegates highlighting to **Pygments**, a Python library covering 300+ languages.

**How it works**

1. LaTeX writes the code snippet to a temporary file.
2. During compilation, it calls Pygments (requires -shell-escape) which returns colorized TeX code.
3. The formatted code is inserted into the PDF.

\begin{minted}[fontsize=\small,linenos]{python}

def black\_scholes\_call(S, K, r, sigma, T):

# vectorised numpy implementation

...

\end{minted}

Pros: beautiful output, language‑agnostic.  
Cons: needs Python and shell access; not allowed on some corporate TeX servers.

## \label{} vs \cref{}

* **\label{mytag}** marks a structural element (section, figure, equation…).
* **\ref{mytag}** later prints the raw number (e.g. “3.2”).
* **\cref{mytag}** (from the cleveref package) prints an **intelligent reference** that automatically adds the correct prefix and handles plurals:

| **Code in text** | **Output** |
| --- | --- |
| \cref{eq:bs} | “(Equation 2.1)” |
| \cref{fig:lob,fig:flash} | “Figures 1.3 and 1.4” |

Big win: you no longer hard‑code “Fig.” vs “Eq.” vs “Sec.” — cleveref figures it out.

## pgfplots

*TikZ/PGF* is the native graphics language of LaTeX. pgfplots is a high‑level package built on top of TikZ for producing **publication‑quality plots** (lines, scatter, heat‑maps) entirely inside LaTeX.

* Benefits: perfect font consistency, vector graphics, no external images.
* Drawbacks: heavier compile times, learning curve for TikZ syntax.

Typical usage:

\begin{tikzpicture}

\begin{axis}[

xlabel={$S$},

ylabel={Option Price $C(S)$},

]

\addplot[samples=100,domain=50:150]{max(x-100,0)};

\end{axis}

\end{tikzpicture}

## pythontex and sympytex

| **Package** | **Purpose** | **Typical Use Case** |
| --- | --- | --- |
| pythontex | Run arbitrary Python during LaTeX compilation and insert the results. | Auto‑generate data tables, numeric examples, or figures without manual copy‑paste. |
| sympytex | Evaluate **Symbolic** mathematics via sympy and insert the formatted expression. | Produce algebraic simplifications, derivatives, integrals inline. |

Both require -shell-escape; both cache results so repeated compilations are fast.  
If you are comfortable with Python, pythontex is often the most flexible choice.