Guida a LaTeX

GuIT - Gruppo Utilizzatori Italiani di TeX e LaTeX

Tutto il necessario per iniziare con LaTeX

versione 0.1.13

August 06, 2026

Indice

1. Guida a LaTeX 5
   1.1. Cos'è LaTeX 5
   1.2. Contesto 5
2. Libero e multipiattaforma 5
3. Differenza tra TeX e LaTeX 5
4. Vantaggi e svantaggi di LaTeX 6
   4.1. Area file 6
   4.2. Area contenuti 6
   4.3. Area tipografica 7
   4.4. Area economica e sociale 7
   4.5. Area apprendimento 7
   4.6. Area utilizzo 7
   4.7. Area collaborazione 8
5. Informazioni pratiche 8
6. File di testo 8
   6.1. Testo puro e testo tipografico 8
   6.2. Nomi dei file 9
   6.3. Creare un file di testo 9
7. Gli editor di testo 9
   7.1. Editor per LaTeX 10
   7.2. Digitare i caratteri speciali 10
8. Riga di comando 10
   8.1. Perché il terminale 10
      8.1.1. Compilazione 10
      8.1.2. Procedure per TeX Live 10
      8.1.3. Utilità generale 10
   8.2. Cos'è il Terminale 10
   8.3. Apri una shell 11
   8.4. Prima esecuzione 11
   8.5. Percorsi 11
   8.6. Qualche comando utile 12
      8.6.1. Creare un file di testo 12
      8.6.2. `texdoc` 12
      8.6.3. `tree` 13
      8.6.4. `pdfcrop` 14
9. Installazione locale di TeX Live 14
   9.1. Ciclo di sviluppo di TeX Live 15
   9.2. Schemi di installazione 15
10. TeX Live su Windows 15
   10.1. Passo 1: file di installazione 15
   10.2. Passo 2 (raccomandato): sicurezza 16
   10.3. Passo 3: Avvio dell'installazione 16
   10.4. Passo 4: installazione 17

10.5. Passo 5: Scaricamento pacchetti ... 18
10.6. Passo 6: test dell'installazione ... 19
10.7. Passo 7: aggiornamenti ... 19
10.8. Passo 8 (raccomandato): verifica dei pacchetti ... 19
11. TeX Live su macOS ... 20
11.1. Passo 1: scarica MacTeX.pkg ... 20
11.2. Passo 2: installazione ... 20
11.3. Passo 3: test ... 21
11.4. Passo 4: aggiornamenti ... 21
11.5. Altre utilità ... 21
12. TeX Live su Linux ... 21
12.1. Passo 1: archivio ... 22
12.2. Passo 2 (raccomandato): verifica `SHA-512` ... 22
12.3. Passo 3: avvio ... 22
12.4. Passo 4: temp file ... 23
12.5. Passo 5: configurazione ... 23
12.6. Passo 6: Test di funzionamento ... 23
12.7. Passo 7: aggiornare TeX Live ... 23
13. TeX Live sicuro su Linux ... 24
13.1. Passo 1: pulizia ... 24
13.2. Passo 2: firma GPG ... 24
13.3. Passo 3: archivio ... 25
13.4. Passo 4: sicurezza ... 25
13.4.1. Verifica del checksum ... 25
13.4.2. Verifica dell'archivio ... 26
13.5. Passo 5: estrazione e installazione ... 26
13.6. Passo 6: eliminazione temp-tl ... 27
13.7. Passo 7: configurazione ... 27
13.8. Passo 8: Test installazione ... 27
13.9. Passo 9: aggiornare TeX Live ... 27
14. LaTeX online ... 28
14.1. Overleaf ... 28
14.2. Cocalc ... 29
14.3. Per approfondire ... 29
15. Primo documento ... 29
16. Matematica ... 30
17. Compilare il sorgente ... 30
17.1. Estensione e codifica ... 31
17.2. Compilare con TeXworks ... 31
18. Compositori ... 32
19. Preambolo e corpo ... 32
20. Pacchetti ... 33
20.1. Pacchetti di uso comune ... 33
20.2. `babel` ... 34
20.3. `graphicx` ... 34

20.4. Gestione della pagina 34
20.5. amsmath 35
20.6. Grafica programmata 35
20.7. Oggetti flottanti 35
20.8. Liste 35
20.9. Tabelle 35
20.10. Varie ed eventuali 35
21. Documentazione di sistema 35
22. Organizzazione di un progetto 36
22.1. Struttura dell'archivio 36
22.2. Assemblare file sorgente 36
22.3. Il comando \input{} 36
22.4. Il comando \include{} 37
23. Il linguaggio di markup 38
23.1. Il carattere di escape 38
23.2. Le macro di LaTeX 39
23.3. Tutto così semplice? 40
24. Macro 40
24.1. Ambienti 40
24.2. Spazi e ritorni a capo 41
24.3. Commenti 42
25. Formattazione del testo 43
25.1. Evidenziare testo con \emph{} 43
25.2. Le macro per il testo 44
25.3. Incisi 44
25.4. Sezionamento del testo 44
25.5. Riferimenti incrociati 45
25.6. Oggetti mobili 46
25.7. Oggetto mobile table 47
25.8. Oggetto mobile figure 47
25.9. Indici (o sommari) 48
25.10. Indice analitico 48
25.11. Elenchi 48
26. Esempi di codice 49
27. Informazioni sulla guida 50
27.1. Contribuire alla guida 50
27.2. Contributori 51
27.3. Citare la guida: entry biblatex 51
27.4. Produzione della guida 51
27.5. Roadmap 51
27.6. Licenza d'uso 52

1. Guida a LaTeX

Lo scopo di questa guida è facilitare l'utente che inizia a usare LaTeX. La forma è quella agile di un documento online integrato nel sito GuIT facilmente consultabile.

La ricerca di un modo efficace per presentare un sistema indubbiamente complesso come LaTeX non può che basarsi su questi capisaldi:

- imparare LaTeX significa comprenderne la filosofia
- e utilizzarlo comporta l'esecuzione di alcune procedure al PC

1.1. Cos'è LaTeX

LaTeX è uno strumento per la tipografia digitale libero. Si è evoluto dalle origini di TeX in un lungo arco di tempo coprendo esigenze sempre più ampie.

Diamone una definizione pratica:

> LaTeX è un programma di composizione tipografica che produce un documento PDF a partire da un file di testo chiamato *sorgente* contenente una *descrizione* del documento stesso espressa con uno specifico linguaggio.

Compito dell'utilizzatore è costruire il file sorgente con un editor di testo. Compito di LaTeX è comporlo in un documento di alta qualità tipografica.

1.2. Contesto

Stiamo parlando di documenti a stampa, dai più semplici a quelli più complessi, nei particolari contesti in cui l'Autore o gli Autori utilizzano in prima persona un software specifico sia per elaborare i contenuti sia per dargli la forma stilistica appropriata.

Lo stile può essere deciso dall'Autore stesso oppure da altri soggetti editoriali. Nel caso di LaTeX ci sono due differenti situazioni:

- o è l'Autore a scrivere il codice LaTeX deputato a dare al documento finale lo stile stabilito,
- oppure è un ente terzo che scrive quel codice di stile mettendolo a disposizione dell'Autore.

In questa guida considereremo principalmente il primo caso e auguriamo al lettore il classico *Happy TeXing*!

2. Libero e multipiattaforma

Il sistema TeX è libero, ciò significa che è possibile utilizzarlo liberamente anche nell'ambito professionale o aziendale, e che il codice è consultabile compreso quello dei pacchetti di estensione. È anche multipiattaforma ovvero è disponibile per molti sistemi operativi compresi Windows, macOS e Linux.

Tutti i programmi e la relativa documentazione sono diffusi via Internet da un insieme di server chiamati *mirror* che fanno capo a CTAN acronimo di *Comprehensive TeX Archive Network*.

3. Differenza tra TeX e LaTeX

Qui dobbiamo raccontare un po' di storia. In origine alla fine degli anni 70 Donald Knuth creò TeX: il linguaggio e la sua implementazione.

Il nome TEX deriva dalla parola greca τεχνη (téchne) che ha il doppio significato di *arte* e *tecnica* a simboleggiare l'unione dei due concetti: l'arte tipografica e la tecnologia informatica. L'ultima lettera deve quindi essere letta con il `ch` di chiave.

Negli anni 80 Leslie Lamport creò un linguaggio macro di alto livello programmato con le primitive di TEX che semplificava il lavoro di impostazione del documento e la costruzione delle sue parti, che chiamò LATEX.

Con la diffusione di LATEX cominciarono a nascere nei vari paesi del mondo i *TEX User Group*. Il GuIT lo è per l'Italia.

4. Vantaggi e svantaggi di LaTeX

Vantaggi e svantaggi non sono a ben vedere così indipendenti tra loro e nemmeno immutabili nel tempo: uno svantaggio può produrre effetti positivi proprio perché ci costringe ad un maggior sforzo creativo, un'idea questa conosciuta come *difficoltà desiderabile*.

Il confronto tra LATEX e i programmi di video scrittura non è quindi assoluto. È invece una *valutazione* basata su esigenze stabilite caso per caso.

Riportiamo per aree tematiche alcune utili considerazioni.

4.1. Area file

Al di la dello strumento usato per crearli, i *documenti digitali* sono file scritti in un certo formato. Per LATEX utilizziamo file in formato testo.

1. **affidabilità**: i sorgenti sono file di testo puro. Possono essere letti e scritti da decine di editor diversi su tutti i sistemi operativi rimanendo leggibili nel futuro per lunghissimo tempo.
2. **indipendenza**: nell'editare i sorgenti siamo indipendenti sia da un dato programma sia dal sistema operativo.
3. **sicurezza**: i file di testo si prestano a essere gestiti dai sistemi di controllo di versione come `git` sulle piattaforme online di sviluppo del codice: backup sicuri, lavoro di gruppo centralizzato, recupero di versioni precedenti in caso di necessità.
4. **organizzazione**: un documento molto complesso può essere facilmente suddiviso in più file che ne copiano la struttura.
5. **automatizzazione**: i programmi possono con facilità leggere e scrivere file di testo per creare report in processi documentali automatici.

4.2. Area contenuti

1. **forma e contenuto**: mentre si scrive non ci si deve preoccupare della forma ma solo dei contenuti. Non ci sono menù con voci predisposte. Siamo liberi di pensare alle relazioni tra parti del testo, alla creazione di rappresentazioni originali e di equilibrare il livello di dettaglio.
2. **stile**: intervenire sull'aspetto del documento non necessita della modifica del codice dei contenuti. LATEX applica le nuove proprietà ai vari elementi: titoli di sezione, tabelle, equazioni matematiche, eccetera.
3. **elaborazione complessa**: le funzioni complesse come la realizzazione del sommario, della bibliografia o dell'indice analitico, sono difficili da realizzare con la maggior parte degli altri programmi mentre in LATEX sono automatici.

4. **nuove funzioni**: se necessario è possibile programmare nuove funzioni per realizzare particolari contenuti secondo le proprie esigenze.

4.3. Area tipografica

1. **tipografia**: si diventa più consapevoli della tipografia rispetto a usare un Word Processor. Sono disponibili stili tipografici professionali.
2. **qualità compositiva**: durante la compilazione il testo può essere composto per interi capoversi e non con la logica riga per riga. Gli algoritmi ottimizzano la cesura in fin di riga e compongono il testo in modo che sia più leggibile.
3. **matematica**: la qualità tipografica delle formule è eccellente, una delle ragioni dell'esistenza di TEX.
4. **precisione**: la precisione di TEX è davvero alta superiore di gran lunga a quello che può apprezzare l'occhio del lettore.

4.4. Area economica e sociale

1. **crescita e condivisione**: essendo LATEX utilizzato da tantissime persone nel mondo condividere il proprio codice personale, le proprie esperienze nei forum o nei meeting diventa inestimabile crescita per tutti i partecipanti.
2. **costo**: il sistema TEX è gratuito. Lo è sia installarlo, che utilizzarlo, che consultarne la documentazione.
3. **accessibilità**: le formule matematiche e il codice di markup è testo puro e può essere reso accessibile facilmente ai non vedenti da strumenti come la barra Braille. Inoltre nel PDF finale possono essere inseriti testi alternativi.

4.5. Area apprendimento

1. **apprendimento ripido**: iniziare a lavorare con LATEX è più difficile che con altri editor "visuali". Non è intuitivo gestire oggetti solamente attraverso comandi testuali. Occorre leggere la documentazione per capire la sintassi e il significato degli argomenti e fare compilazioni di verifica del risultato.
2. **cambio di paradigma**: per chi è abituato potrebbe non essere facile non vedere il documento così come verrà stampato mentre lo si scrive. LATEX richiede maggiore astrazione non essendo una strumento *WYSIWYG* acronimo di *What You See Is What You Get*.
3. **TEX User Group**: l'ampia diffusione di LATEX dà vita ai gruppi locali di utenti, proprio come la nostra associazione GuIT, che aiutano a iniziare e con le richieste particolari.

4.6. Area utilizzo

1. **strutturare il documento**: richiede di intercalare il testo con comandi di composizione che definiscono la struttura logica del testo.
2. **mono funzionale**: non è un ambiente integrato con foglio di lavoro, posta elettronica, calendario degli appuntamenti e agenda degli indirizzi.
3. **risolvere gli errori**: i messaggi di errore spesso non sono semplici da interpretare.
4. **coerenza sintattica**: a volte la sintassi di comandi definiti dai pacchetti possono non rispettare le stesse regole sintattiche di LATEX introducendo livelli di incoerenza nel linguaggio di cui bisogna tener in conto.

5. **caratteri**: occorre digitare caratteri che normalmente non si usano: le parentesi quadre `[`, `]`, le graffe `{`, `}`, l'ampersand `&`, il cancelletto `#`, il backslash `\`, il dollaro `$`, eccetera.

4.7. Area collaborazione

1. **interscambio documenti**: spesso l'uso di LaTeX in ambito professionale o aziendale è limitato dalle scelte dei gruppi di lavoro. Non è fattibile utilizzare per alcuni documenti LaTeX e per altri un Word Processor perché essi devono poter essere scambiati con immediatezza o elaborati a più mani.
2. **aiuto**: in caso si voglia utilizzare LaTeX in azienda non è semplice reperire consulenza esperta.

5. Informazioni pratiche

Produrre documenti di qualità con LaTeX significa lavorare con i file di testo. Si tratta dei così detti *file sorgenti*.

Diamo quì informazioni essenziali su:

- cos'è un file di testo
- con quali programmi si editano e
- come si utilizza il terminale

6. File di testo

Ci proponiamo di imparare cos'è un file di testo e quali programmi utilizzare per editarli. Infatti è proprio così che si utilizza LaTeX: si compilano file di testo detti sorgenti.

> Un file di testo è semplicemnete una sequenza di numeri. Ogni numero rappresenta un carattere attraverso una tabella di *codifica*.

La codifica da utilizzare è UTF-8, uno standard UNICODE multipiattaforma compatibile con ASCII che codifica migliaia di caratteri per i diversi alfabeti delle lingue parlate nel mondo.

I file di testo rappresentano un formato essenziale per l'utilizzo dei moderni sistemi informatici. Il Web che costituisce il principale utilizzo della rete Internet è costruito sul formato HTML che altro non è che un file di testo non molto diverso da un sorgente LaTeX.

6.1. Testo puro e testo tipografico

Confrontando un file di testo puro con il testo composto in un PDF prodotto da LaTeX, notiamo molte differenze. Eccone alcune:

- nel testo composto gli spazi tra le parole non corrispondono al carattere spazio codificato con il numero 32, ma a uno spazio di larghezza variabile che consente una migliore leggibilità
- nel testo composto alcune sequenze di caratteri sono raggruppate in un unico glifo. Si tratta delle *legature*. Per esempio le lettere `fi` sono unite insieme e trasformate in `ﬁ` dove il puntino della `i` e il trattino della `f` sono fusi insieme nel disegno dell'altra lettera
- nel testo composto non esiste il ritorno a capo sulla riga perché il capoverso è costruito verticalmente a *scatole* indipendenti poste una di seguito all'altra

6.2. Nomi dei file

I file, i contenitori di informazione organizzati nella memoria su disco del computer con una struttura ad albero detta *file system*, hanno nomi in cui è raccomandabile evitare il carattere punto `.` e il carattere spazio sostituibile con un trattino `-`.

In coda, i nomi dei file sono caratterizzati da un piccolo gruppo di due o tre caratteri finali detti *estensione* separati da un punto.

> **Nota**
>
> l'estensione generica per un file di testo puro è `.txt`

Possiamo intendere l'estensione come informativa sullo scopo del file di testo. Per esempio, questi sono *tutti* file di testo con una differente estensione utilizzati in LaTeX per compiti specifici:

- `.tex` file sorgente di LaTeX
- `.bib` file bibliografico
- `.log` file contenente le informazioni di compilazione
- `.aux` file ausiliario prodotto da LaTeX per tracciare i riferimenti
- `.toc` file ausiliario prodotto da LaTeX per la composizione dei sommari

> **Nota**
>
> L'estensione nel nome del file potrebbe essere nascosta dal sistema operativo. Su Windows per esempio esiste l'opzione 'Mostra/Nascondi le estensioni per i tipi di file conosciuti' presente nelle finestre che mostrano il contenuto delle directory

6.3. Creare un file di testo

Il modo più semplice per creare un nuovo file di testo è quello di fare un click destro nella finestra in cui va posizionato e dal menù contestuale scegliere la voce **Nuovo** > **Documento di Testo** (le voci del menù possono essere diverse a seconda del sistema operativo).

Oppure si può dare un comando da terminale come descritto nella sezione dedicata della guida.

In alternativa e altrettanto semplicemente, si può far ricorso agli editor di testi. TeXworks per esempio, ha nel menù "File" la voce "Nuovo" attivabile anche con la combinazione di tasti CTRL + N.

7. Gli editor di testo

Negli ultimi anni quello che sembrava ormai un settore stabile senza più novità ha invece ripreso vita con innovative nuove funzionalità: cursore multiplo, ricerca/sostituzione sofisticata, editor modali, integrazione con strumenti esterni, aiuto nella scrittura del codice

tramite Language Server Protocol, integrazione con AI, editazione online in più utenti, plug-in di estensione.

Parliamo degli editor di testo, programmi con cui è possibile scrivere testo puro evoluti in piattaforme efficienti e complete.

7.1. Editor per LaTeX

In progress

7.2. Digitare i caratteri speciali

In progress

8. Riga di comando

8.1. Perché il terminale

8.1.1. Compilazione

I compositori del sistema TeX sono programmi a *linea di comando*. Compilare un sorgente per ottenere il PDF di fatto consiste nell'eseguire uno specifico comando al terminale

Poichè nella maggior parte dei casi l'esecuzione del compositore avviene "dietro le quinte" premendo un pulsante nella finestra dell'editor, possiamo in qualche modo pensare che l'interfaccia grafica di questi programmi sia l'editor stesso.

Sapere che l'azione compilare significa eseguire un comando al terminale,

8.1.2. Procedure per TeX Live

Installare e tenere aggiornata una distribuzione TeX Live sul proprio PC, implica l'esecuzione di comandi al terminale, specie su Linux e Windows.

8.1.3. Utilità generale

se si lavora al computer conoscere il terminale può aiutare a svolgere compiti ripetitivi o elaborazioni sui file in modo automatico e sicuro.

8.2. Cos'è il Terminale

Il *terminale* è una finestra di testo in cui si digitano *comandi* invece di fare click su icone o menù, che richiedono elaborazioni ai programmi o al sistema operativo.

Sinonimi del termine "terminale":

- riga (o linea) di comando
- shell
- prompt dei comandi
- console

> ⚠ **Attenzione**
>
> Questa sezione tratta l'argomento del Terminale in modo sintetico. Tieni conto che è invece molto vasto e articolato. L'evoluzione di questo componente storico dell'informatica è ancora attiva e allarga ulteriormente il campo.

8.3. Apri una shell

Prova ad aprire una finestra di terminale. In questo elenco trovi alcuni modi per farlo a seconda del sistema operativo:

- in **Windows**, premi Start e nella barra di ricerca inizia a scrivere `terminale` poi premi invio.
- in **Linux**, premi CTRL + Alt + T
- in **macOS**, premi Cmd + Spazio e digita "Terminale"

La finestra è molto semplice: noterai un cursore lampeggiante che segnala il punto di inserimento del testo che segue il *prompt* che riporta la directory di lavoro.

8.4. Prima esecuzione

Ora che hai davanti un finestra di terminale prova a dare un comando così da chiarire ogni dettaglio operativo.

Scrivi `pwd` e poi invio. Il comando sta per *print working directory*. Dovresti leggerla come effetto dell'esecuzione le righe sottostanti.

Nella guida indicheremo i comandi così (nota che a destra si attiva il pulsante di copia del contenuto se passi sul riquadro del codice con il mouse):

```
pwd
```

Spesso i comandi accettano opzioni e argomenti. Spesso, ma non è una regola che vale sempre, le opzioni hanno nomi che iniziano con due trattini per esempio `--color green` oppure più brevemente con un trattino solo e una lettera, per esempio `-c green`.

Quasi tutti i comandi stampano un messaggio di aiuto se si da solo l'opzione `--help` o `-h` così da ricordare le possibili varianti.

8.5. Percorsi

Spesso i comandi operano sui file e sono file loro stessi. Sarebbe molto scomodo dover digitare tutto il percorso corrispondente nell'albero del file system perciò ci sono due aiuti:

- la *directory di lavoro* già incontrata prima
- il `PATH` di sistema
- le directory `.` e `..`

Quando esegui un comando il sistema operativo cerca tra i percorsi della variabile `PATH`. Ciò ti permette di digitare il nome del comando senza altre informazioni.

Quando indichi un file che si trova nella directory di lavoro puoi digitarne solo il nome.

Quando hai necessità di imporre l'esecuzione di un comando il cui eseguibile si trova enlla directory di lavoro, scrivi il percorso come `./nome-del-comando`.

La directory superiore a quella della directory di lavoro è rappresentata da `..`. Se hai necessità di spostarti un livello più su usa il comando `cd` che sta per *change directory* dando il doppio punto come percorso.

Se utilizzi il comando `cd` senza argomenti ti sposterai nella directory `home` dove sono contenuti i file di lavoro del tuo personale account.

8.6. Qualche comando utile

8.6.1. Creare un file di testo

Per creare file di testo con comandi da terminale esistono davvero tantissime modalità. In Linux basta dare un comando `touch` seguito dal nome del file:

```
touch mio-sorgente.tex
```

In Windows, se utilizziamo PowerShell si usa `New-Item` o la sua versione breve `ni`, sempre seguito dal nome del file:

```
ni mio-sorgente.tex
```

Si può usare anche la redirezione dell'output con il comando `echo` che stampa il testo dato come argomento. Questi dati poi, possono essere indirizzati verso un file attraverso il simbolo di maggiore `>`.

L'idea funziona su tutti i sistemi operativi macOS, Windows e Linux: per creare un file non vuoto ma contenente i caratteri `Hello World!` possiamo eseguire il comando:

```
echo "Hello World!" > mio-sorgente.tex
```

8.6.2. `texdoc`

I comandi da terminale possono essere interattivi. Per esempio l'utility `texdoc` inclusa in TEX Live per la ricerca di documentazione, ha l'interessante opzione `-l` (trattino elle). La lettera sta per `list` e dice al comando di stampare l'elenco numerato di tutta la documentazione disponibile su un componente e di attendere la decisione dell'utente su quale aprire.

In questa immagine puoi leggere il comando che elenca la documentazione del pacchetto `babel` e l'output generato:

```
$ texdoc -l babel
124 results. Only the top 10 are shown below.
 1 c:\texlive\2026\texmf-dist\doc\latex\babel\babel.pdf
   = User guide
 2 c:\texlive\2026\texmf-dist\doc\latex\babel\babel-code.pdf
   = Code documentation
 3 c:\texlive\2026\texmf-dist\doc\generic\babel-albanian\albanian.pdf
   = Package documentation
 4 c:\texlive\2026\texmf-dist\doc\generic\babel-azerbaijani\azerbaijani.pdf
   = Package documentation
 5 c:\texlive\2026\texmf-dist\doc\generic\babel-basque\basque.pdf
   = Package documentation
 6 c:\texlive\2026\texmf-dist\doc\generic\babel-belarusian\belarusian.pdf
   = Package documentation
 7 c:\texlive\2026\texmf-dist\doc\generic\babel-bosnian\bosnian.pdf
   = Package documentation
 8 c:\texlive\2026\texmf-dist\doc\generic\babel-breton\breton.pdf
   = Package documentation
 9 c:\texlive\2026\texmf-dist\doc\generic\babel-bulgarian\bulgarian.pdf
   = Package documentation
10 c:\texlive\2026\texmf-dist\doc\generic\babel-catalan\catalan.pdf
   = Package documentation
Enter number of file to view, RET to view 1, S to show all results, or any other
 key to exit: 8
```

8.6.3. `tree`

Per visualizzare la struttura dei file esiste il comando `tree`. È utile per stampare l'albero dei sorgenti di un nostro progetto.

Se è installato provate a eseguirlo scrivendone semplicemente il nome seguito dal tasto invio:

```
tree
```

Facendolo dalla directory `home` della guida si ottiene questa rappresentazione:

```
├──00-intro
├──00-text
│   └──images
├──01-install
│   └──images
├──02-firstdoc
│   └──images
├──03-structure
├──11-lang
│   └──images
├──21-produzione
└──91-misc
    └──images
```

8.6.4. `pdfcrop`

Questa utilissima utility di Heiko Oberdiek compresa in TEX Live vi farà risparmiare tantissimo tempo e fatica nel caso servisse scontornare immagini PDF.

Esempio: abbiamo realizzato un grafico a torta che presenta un ampio bordo bianco. Inserito così com'è nel nostro documento con il pacchetto `graphicx` è un brutto risultato: ci sono ampi spazi bianchi attorno all'immagine senza ragione.

Una soluzione è attivare l'opzione di clipping della macro `\inclugraphics` ma occorre dare a mano per tentativi i quattro valori sui singoli lati. Non è solo noiso da fare, diventa impossibile se le immagini con bordo da inserire sono tante.

Con un comando da terminale risolviamo tutto: `pdfcrop` legge l'immagine PDF, calcola il box dei contenuti e scontorna ad esso il file:

```
pdfcrop grafico_torta.pdf
```

Otterrai il file `grafico_torta-crop.pdf` perfettamente scontornato pronto per inserirlo nel documento principale.

Opzione utile è `--margins` (nota il doppio trattino) con la quale puoi aggiungere un contorno al box minimo e, ancora, puoi specificare un nome per il file ritagliato. L'opzione `--help` data da sola al comando stampa tutte le opzioni disponibili accompagnate da una breve spiegazione:

```
pdfcrop --margins 5 input.pdf output.pdf
pdfcrop --margins '5 10 5 20' --clip input.pdf output.pdf
pdfcrop --help
```

> ⚠ **Attenzione**
>
> Non è consigliabile dare a pdfcrop come nome del file di uscita quello del file da ritagliare perché così il file originale viene sovrascritto e perso

Se poi i file da ritagliare sono tanti e sono salvati in una stessa cartella con un unico comando possiamo scontornarli tutti in una volta.

9. Installazione locale di TeX Live

La modalità classica di lavorare con il sistema TEX è quella di installarlo sul proprio computer: significa operare in *locale*.

TEX Live è la distribuzione TEX ufficiale; è completa, multipiattaforma, e scaricabile liberamente dai server di CTAN.

Esistono alternative, quella più nota è forse MiKTEX, ma qui proporremo TEX Live in versione *vanilla*, ovvero quella non pacchettizzata, installata direttamente dalla fonte.

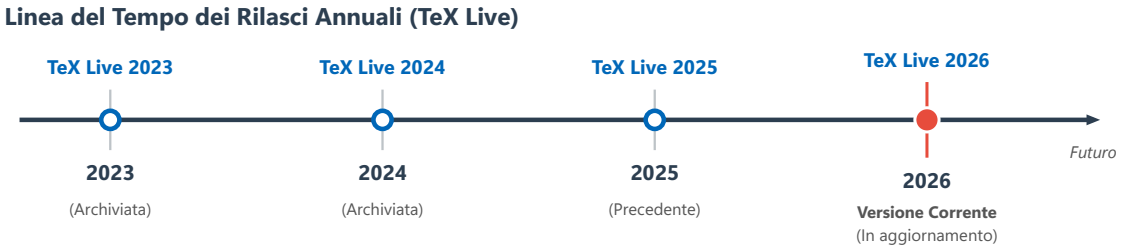
Qui trovi i link diretti alle procedure passo passo per l'installazione per i principali sistemi operativi desktop con le modalità di aggiornamento:

- Windows

- macOS
- Linux:
  - Debian/Ubuntu semplice
  - Debian/Ubuntu crittografica

9.1. Ciclo di sviluppo di TeX Live

La cosa più importante da sapere di TEX Live è che ha un ciclo di sviluppo a salti: ogni anno viene rilasciata una nuova versione. In quel momento, i pacchetti della nostra installazione locale non riceveranno più aggiornamenti.



Siamo noi a dover decidere se e quando passare alla versione successiva. Di solito chi utilizza LATEX in modo continuativo esegue l'upgrade al rilascio della nuova versione di TEX Live per disporre sempre di una piattaforma aggiornata e con le nuove funzionalità.

Ogni rilascio annuale di TEX Live si installa allo stesso modo e non elimina la versione precedente che potrà eventualmente essere eliminata successivamente.

9.2. Schemi di installazione

L'installazione completa (*schema full*) di TEX Live comprende più di 5000 pacchetti, e comprende la documentazione di sistema e i font con licenza libera. Occupa circa 10 GiB su disco.

> ⚠ **Attenzione**
>
> In questa guida è trattata la procedura *Net installation* con lo schema *completo* di TeX Live. Per le altre modalità di installazione o per conoscere gli schemi disponibili consulta questa guida.

10. TeX Live su Windows

L'installazione di TEX Live su Windows in semplici mosse.

10.1. Passo 1: file di installazione

Per prima cosa scarica il file d'installazione `install-tl-windows.exe`. Se desideri eseguire la verifica di sicurezza del file prosegui con il passo 2 altrimenti vai al passo 3.

10.2. Passo 2 (raccomandato): sicurezza

Verifichiamo che l'archivio dell'installer sia sicuro con il confronto tra il suo checksum `SHA-512` e quello atteso.

Da CTAN scarica perciò anche il file `install-tl-windows.exe.sha512`. Esso contiene il checksum atteso e il nome del file da verificare separati da uno spazio.

Crea una nuova cartella, aprila con l'Explorer e copiaci dentro i due file scaricati. Con un click destro in un punto vuoto della finestra apri una sessione di PowerShell scegliendo la voce "Apri nel Terminale".

Nella finestra di PowerShell dai questo comando e premi invio:

```
(Get-FileHash install-tl-windows.exe -Algorithm SHA512).Hash -eq
(Get-Content install-tl-windows.exe.sha512).Split()[0]
```

Se copi/incolli il comando con il pulsante a destra dell'area del codice conferma che stai inserendo un comando su più linee.

Se lo digiti da tastiera scrivilo su una sola riga oppure per comodità vai a capo dopo `-eq`. Nella digitazione ti puoi aiutare con il tasto `Tab`: se i caratteri sono sufficienti i comandi e i percorsi dei file vengono completati in automatico.

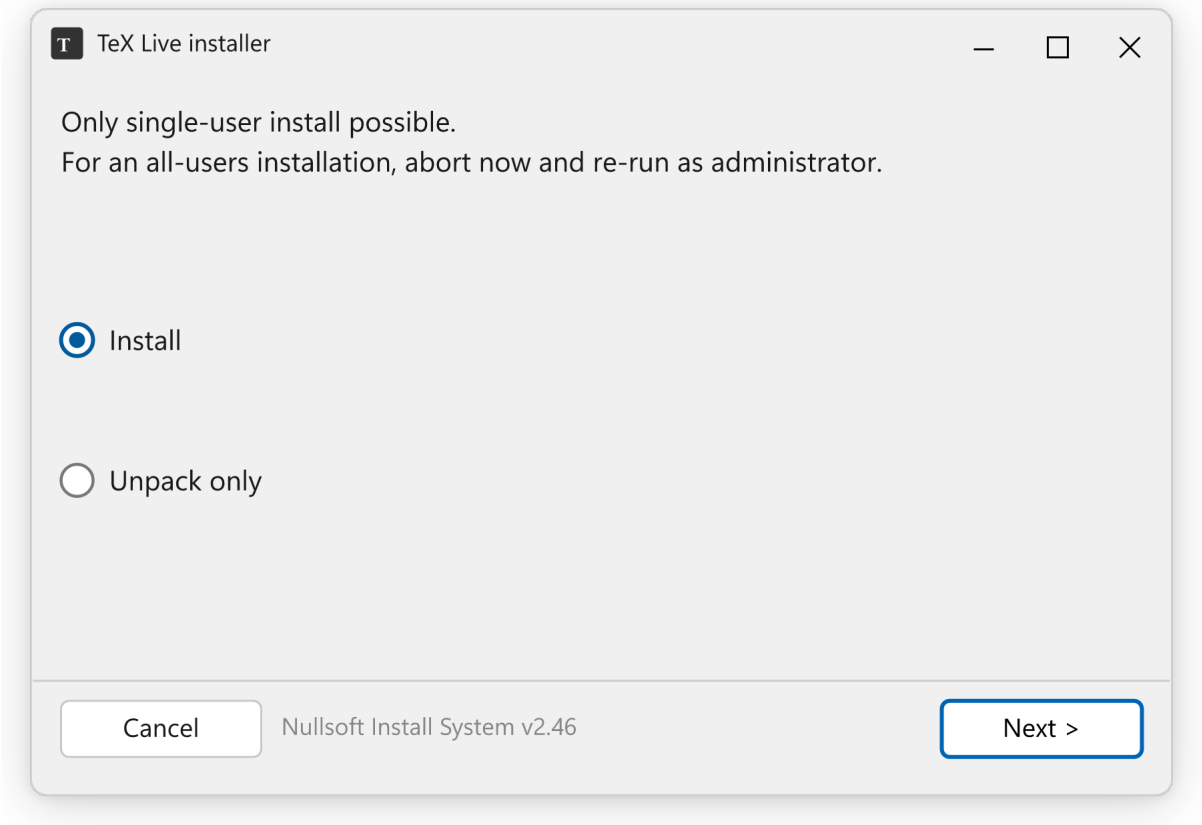
Il comando implementa il confronto tra i due checksum:

1. il checksum dell'archivio calcolato con l'utility `Get-FileHash`
2. il checksum atteso ricavato separando con il metodo `.Split()` la prima parte del contenuto del file `.sha512` letto con il comando `Get-Content`

Se il comando dà come risultato `True` il file è sicuro e puoi procedere, altrimenti ottieni `False` e allora elimina i file e scegli un altro mirror CTAN di download.

10.3. Passo 3: Avvio dell'installazione

Fai doppio click sull'installer. Ti troverai in questa schermata:



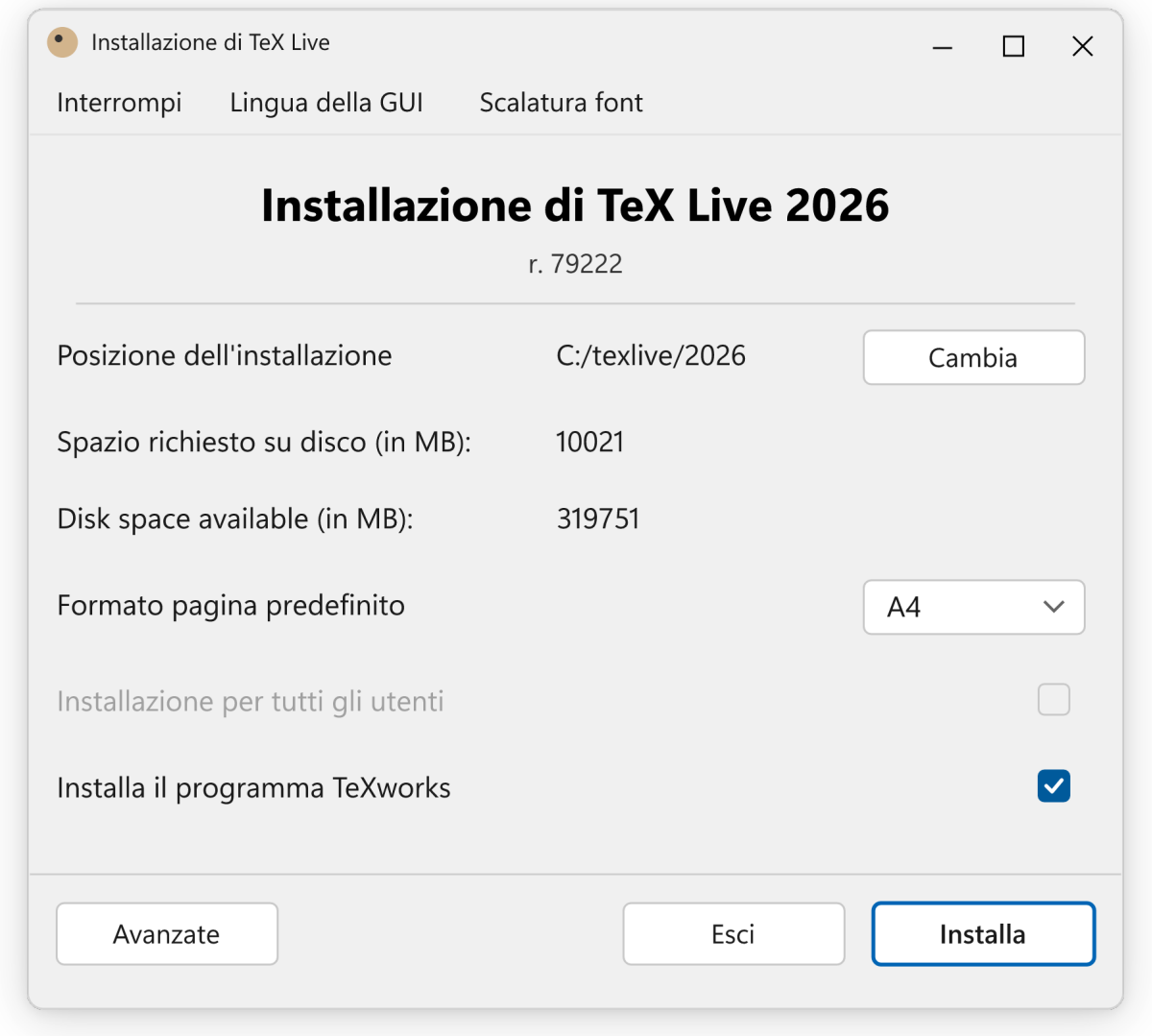
Premi Next e poi Install per decomprimere in una cartella temporanea i file e lanciare l'installatore vero e proprio.

> **Nota**
>
> Windows è probabile che blocchi l'esecuzione dell'installer poiché non essendo firmato è considerato potenzialmente pericoloso. In tal caso nel dialogo di blocco fai click su "Ulteriori Informazioni". Apparirà il motivo, ovvero che l'autore dell'eseguibile è sconosciuto, e il pulsante "Esegui comunque".

> **Suggerimento**
>
> Se desideri che TeX Live sia disponibile per tutti gli utenti accreditati sul computer Windows, esegui l'installatore come amministratore scegliendo la voce corrispondente del menù contestuale con un click destro sull'icona

10.4. Passo 4: installazione

A questo punto dovresti essere alla schermata principale dell'installer.

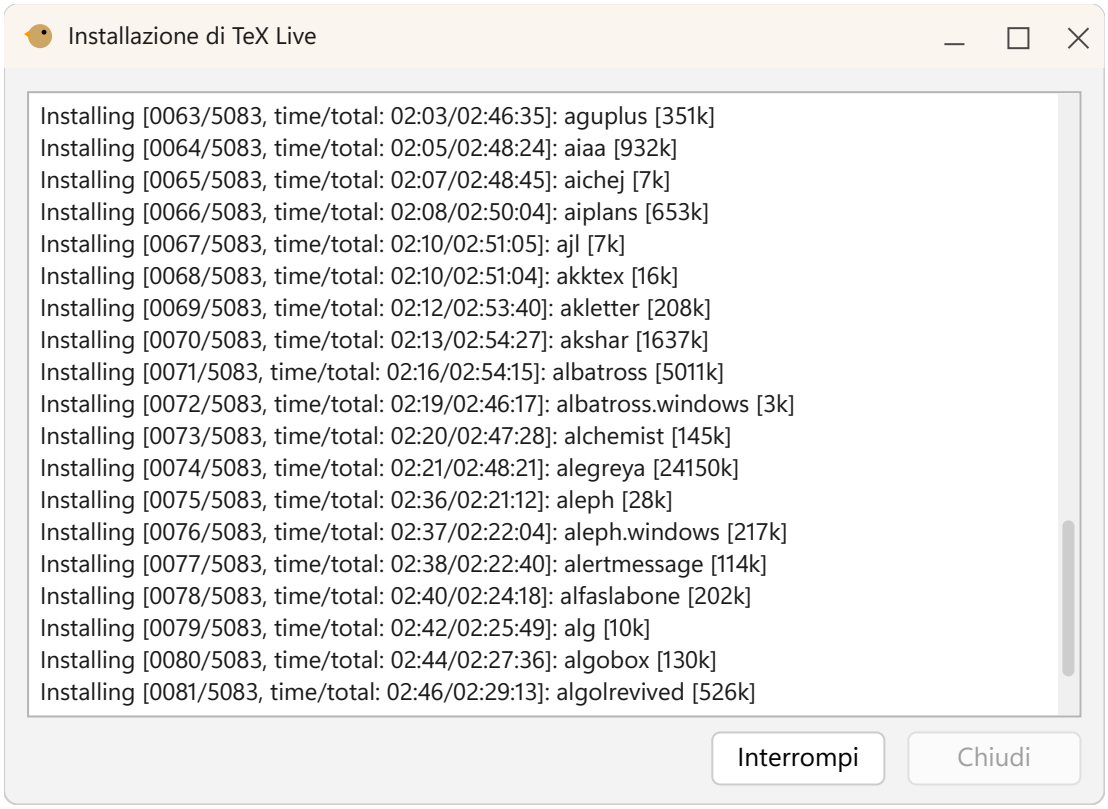


Non è necessario fare modifiche nel dialogo per il tipo di installazione che vogliamo fare tranne che per la scelta di installare o meno anche l'editor TEXWorks, comodissimo perché offre la doppia finestra affiancata codice/documento quindi consigliato.

Annota la directory "Posizione dell'installazione" che potrebbe essere utile in seguito, e premi Install.

10.5. Passo 5: Scaricamento pacchetti

Ottimo. Ultima fase: l'installer scarica e installa i pacchetti da uno dei server CTAN. È una procedura lenta poiché i pacchetti sono veramente tanti. Poco importa, attesa ripagata.



Al termine dell'installazione dei pacchetti, vengono eseguite le configurazioni finali e la generazione dei formati.

10.6. Passo 6: test dell'installazione

Ora tutto è pronto per fare un semplice test. Premi il tasto Windows e digita i caratteri `shell`. Fai invio sulla corrispondenza migliore per aprire una finestra di Windows PowerShell. Al prompt digita il seguente comando e fai un invio per confermarne l'esecuzione:

```
lualatex --version
```

Dovresti ottenere una serie di informazioni di versione che confermano che il sistema è pronto all'uso.

Per ulteriori informazioni sull'installazione di TeX Live per Windows fai riferimento alla pagina ufficiale windows.html.

10.7. Passo 7: aggiornamenti

Ogni giorno vengono rilasciati aggiornamenti conviene quindi periodicamente eseguire dal terminale il comando:

```
tlmgr update --all
```

10.8. Passo 8 (raccomandato): verifica dei pacchetti

Fare in modo che gli aggiornamenti di TeX Live siano verificati è ottima cosa. Vediamo in dettaglio.

Una volta in aggiornamento l'utility `tlmgr` stampa una prima riga che indica il repository di riferimento:

```
tlmgr.pl: package repository https://ctan.net/systems/texlive/tlnet/ (not
verified: gpg unavailable)
```

Dall'ultima parte del messaggio è chiaro che in questo caso i pacchetti in aggiornamento NON saranno verificati con `gpg`. Questo capita di norma su Windows o macOS ma non su Linux dove di solito `gnuPG` è già installato.

Risulta facilissimo installarlo anche in Windows (o macOS) e all'interno di TEX Live con quest'unico comando, che scarica il necessario dal repository personale di Norbert Preining, uno dei principali sviluppatori di TEX Live:

```
tlmgr --repository http://www.preining.info/tlgpg/ install tlgpg
```

Facile, rapido e assolutamente consigliato.

11. TeX Live su macOS

Per gli utenti Mac esiste una versione di TEX Live specifica denominata MacTEX adattamento al sistema di installazione per questo sistema operativo. La versione minima richiesta è la 11 Big Sur uscita nel 2020.

Gli eseguibili girano nativamente sia su piattaforma Intel sia su quella Apple Silicon con i processori della serie M.

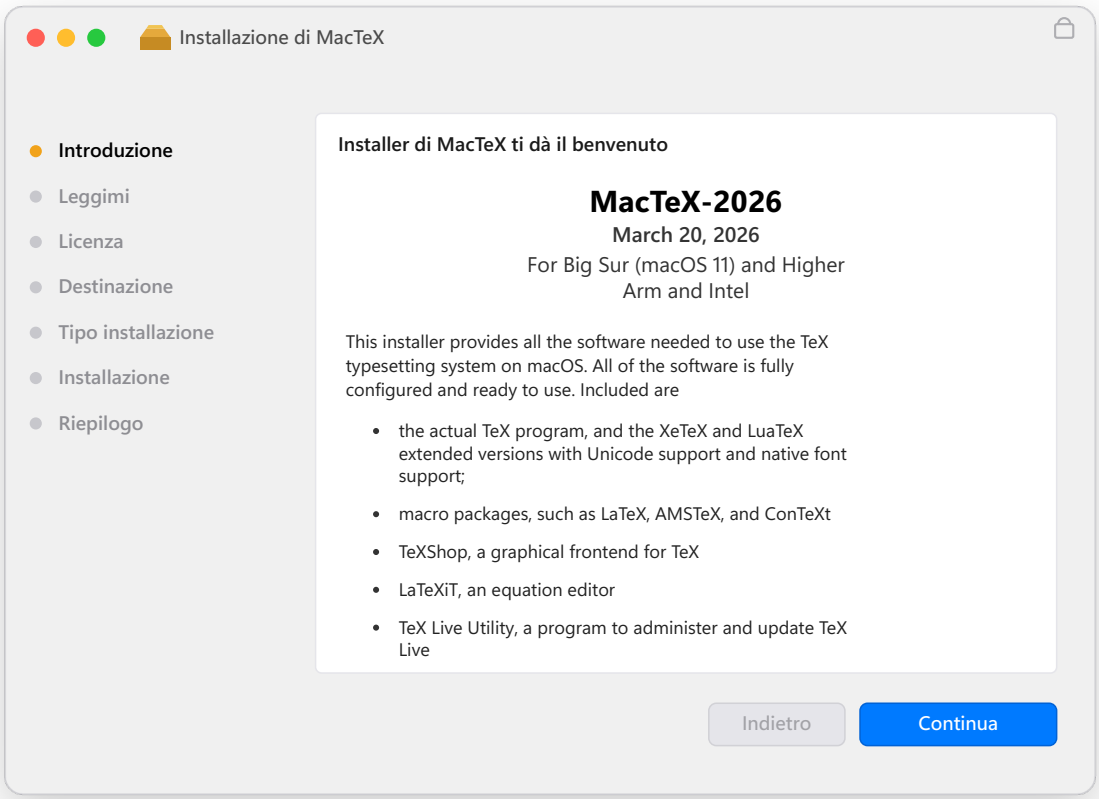
11.1. Passo 1: scarica MacTeX.pkg

Con un click alla pagina ufficiale di download oppure direttamente dal seguente link, scarica il pacchetto MacTEX.pkg per circa 6.4 GiB di peso.

11.2. Passo 2: installazione

Fai click sopra al file scaricato avviando l'installer. In alto a destra dovrebbe comparire l'icona con il lucchetto chiuso che indica che il file e la firma dell'Autore del programma sono verificati e si può procedere in sicurezza.

L'installer apre questo dialogo di benvenuto con informazioni generali e licenza d'uso. Per l'installazione completa non modificare nulla fino ad arrivare alla pagina finale premendo "Install".



11.3. Passo 3: test

Al termine dell'installazione apri un Terminale (premi Cmd + Spazio e digita "Terminale") ed esegui il comando

```
lualatex --version
```

Se leggi le informazioni di versione tutto è pronto per lavorare con LaTeX sul tuo Mac.

11.4. Passo 4: aggiornamenti

MacTeX installa anche l'applicazione TeX Live Utility in `Applications/TeX/`. Si tratta dell'interfaccia grafica di `tlmgr` che opera sulla TeX Live per aggiornamenti e altro. Eseguila periodicamente.

11.5. Altre utilità

In MacTeX ci sono anche due editor: TeXShop e TeXworks, entrambi molto comodi per lavorare con i file di LaTeX perché offrono le righe magiche e la ricerca diretta e inversa tra codice e PDF composto.

Menzioniamo anche il programma BibDesk che gestisce database bibliografici.

12. TeX Live su Linux

I passi qui descritti potrebbero dover essere adattati alla tua distribuzione Linux. Ci riferiremo a distro Debian e derivate, come Ubuntu.

Per le altre distro mandaci la tua guida all'installazione seguendo le indicazioni della sezione Contribuire alla guida.

> 🔥 **Suggerimento**
>
> Se non vuoi digitare i comandi copiali con l'apposito pulsante a destra del box del codice teli e incollali nel terminale con CTRL + Shift + V

12.1. Passo 1: archivio

Scarica l'archivio `install-tl-unx.tar.gz` contenente il necessario da CTAN e procedi con un click destro sulla sua icona a scompattarlo (voce "Estrai" o simile).

12.2. Passo 2 (raccomandato): verifica `SHA-512`

Accertiamoci che il file `install-tl-unx.tar.gz` sia stato scaricato correttamente e che non sia stato manipolato, ovvero che l'installer sia sicuro.

Scarica il file `install-tl-unx.tar.gz.sha512` che contiene la stringa di controllo `sha-512`.

Apri un terminale nella finestra dove l'archivio e il file di checksum sono stati salvati (con un click destro sullo sfondo libero da icone scegliendo "Apri Terminale qui" oppure con la combinazione di tasti `CTRL + Alt + T`) e digita il comando:

```
shasum -c install-tl-unx.tar.gz.sha512
```

Il responso è positivo se il comando stampa:

```
install-tl-unx.tar.gz.sha512: OK
```

Se negativo appare invece `FAILED` dunque elimina i file e riprova con un altro mirror CTAN.

12.3. Passo 3: avvio

Apri la cartella `install-tl-20*` appena creata e con un click destro sullo sfondo libero da icone scegli "Apri Terminale qui". Nella finestra del terminale digita il comando seguente e fai invio per eseguirlo:

```
sudo ./install-tl
```

Nell'interfaccia testuale, alla linea "Enter command:" premi la lettera `I` per *installation* e invio per confermare. Non è necessario modificare parametri.

I singoli pacchetti saranno scaricati con la protezione del protocollo di rete, dai server ufficiali CTAN. L'operazione può richiedere da 30 minuti a un'ora o più, in base alla tua connessione e alla velocità del computer.

Al termine apparirà il messaggio di benvenuto:

```
Benvenuto in TeX Live!
See /opt/texlive/2026/index.html for links to documentation.
```

```
The TeX Live web site (https://tug.org/texlive/) provides all updates
and corrections. TeX Live is a joint project of the TeX user groups
around the world; please consider supporting it by joining the group
best for you. The list of groups is available on the web
at https://tug.org/usergroups.html.

...
```

12.4. Passo 4: temp file

Seleziona sia l'archivio compresso che la directory temporanea non più necessaria e premi `Canc` per eliminarle.

12.5. Passo 5: configurazione

A questo punto TEX Live deve essere configurata manualmente non potendo contare sugli automatismi del sistema di pacchetti proprio della distribuzione Linux.

Istruiamo Debian/Ubuntu su dove trovare i programmi di TEX Live appena installati.

Apri il file `.profile` nel tuo editor di testo preferito. Si tratta di un file *nascosto* della `home` dell'utente. Premendo `CTRL + H` si attiva/disattiva la visualizzazione delle rispettive icone.

Vai in fondo al file e incolla le seguenti righe:

```
# addition for TeX Live 2026
export PATH=/usr/local/texlive/2026/bin/x86_64-linux:${PATH}
```

Esci salvando e ricarica senza riavviare le impostazioni d'ambiente con il comando:

```
source ~/.profile
```

12.6. Passo 6: Test di funzionamento

Verifichiamo che il sistema riconosca correttamente LATEX. Digita nel terminale:

```
lualatex --version
```

Se compare il testo che inizia con `This is LuaHBTeX, Version 1.24.0...` la procedura è terminata con successo. Il tuo ambiente TEX Live è installato, aggiornato e sicuro.

12.7. Passo 7: aggiornare TeX Live

L'utility di aggiornamento `tlmgr` ha necessità dei privilegi di amministratore dovendo scrivere in una directory di sistema. Se lanciamo il comando con `sudo` l'utility non verrà trovata perché non saranno riconosciti i percorsi nell'ambiente di esecuzione amministrativo. La soluzione più semplice consisterebbe nel digitare *tutto* il percorso ma non è comodo.

Basta creare un *alias* all'interno del file `.bash_aliases`. Si tratta di un nome che prende il posto di un comando lungo utilizzato spesso.

Apri il file `.bash_aliases` nella `home` dell'utente. Se non esiste crealo, e aggiungi le righe seguenti per creare l'alias `sutlmgr`:

```
# addition for TeX Live 2026
alias sutlmgr="sudo /usr/local/texlive/2026/bin/x86_64-linux/tlmgr"
```

Potrai usare `sutlmgr` da terminale per acquisire gli aggiornamenti dei pacchetti. I comandi più comuni sono:

```
# per controllare la presenza di aggiornamenti:
tlmgr update --list

# per aggiornare tutto:
sutlmgr update --all

# per controllare se un pacchetto è installato:
tlmgr show <nome_del_pacchetto>
```

13. TeX Live sicuro su Linux

I passi qui descritti potrebbero dover essere adattati alla tua distribuzione Linux. Ci riferiremo a distro Debian e derivate, come Ubuntu.

Per le altre distro mandaci la tua guida all'installazione seguendo le indicazioni della sezione Contribuire alla guida.

Tutta la procedura prevede solo comandi da terminale, come da tradizione Linux, e la massima sicurezza di livello crittografico per un'installazione configurata, pronta all'uso e sicura.

> 🔥 **Suggerimento**
>
> Se non vuoi digitare i comandi copiali e incollali nel terminale con CTRL + Shift + V

13.1. Passo 1: pulizia

Sul tuo sistema potresti avere la TeX Live installata tramite il gestore di pacchetti della distro. Nessun problema ma qui parliamo di una TeX Live fresca, installata direttamente da CTAN, indipendente dal gestore di pacchetti.

Se vuoi eliminarla apri un terminale e esegui i tre comandi:

```
sudo apt update && sudo apt upgrade
sudo apt remove --purge texlive*
sudo apt autoremove
```

13.2. Passo 2: firma GPG

Il team di TeX Live firma i file con una chiave crittografica come descritto in questa pagina. Per importare la chiave pubblica esegui il comando:

```
gpg --auto-key-locate=wkd --locate-keys "tex-live@tug.org"
```

L'output del comando è qualcosa di simile a:

```
gpg: chiave 0D5E5D9106BAB6BC: chiave pubblica "TeX Live Distribution <tex-
live@tug.org>" importata
gpg: Numero totale esaminato: 1
gpg:               importate: 1
pub   rsa2048 2016-03-19 [SC]
      C78B82D8C79512F79CC0D7C80D5E5D9106BAB6BC
uid           [ sconosciuto] TeX Live Distribution <tex-live@tug.org>
sub   rsa2048 2016-03-19 [E]
sub   rsa2048 2016-03-19 [S] [scadenza: 2027-07-13]
```

Una volta importata la chiave sarà possibile verificare i file firmati dal Team.

13.3. Passo 3: archivio

Scarichiamo il necessario in una directory temporanea, chiamiamola `temp-tl`, posizionata nella `home` dell'utente. I file da scaricare sono:

- l'installer ufficiale come archivio compresso
- il file con il suo checksum `sha-512`
- il file di firma `gpg` del checksum

Ecco i comandi:

```
cd ~
mkdir temp-tl
cd temp-tl
wget http://mirror.ctan.org/systems/texlive/tlnet/install-tl-unx.tar.gz
wget http://mirror.ctan.org/systems/texlive/tlnet/install-tl-unx.tar.gz.sha512
wget http://mirror.ctan.org/systems/texlive/tlnet/install-tl-unx.tar.gz.sha
512.asc
```

13.4. Passo 4: sicurezza

Effettueremo una doppia verifica:

1. controllo di autenticità del checksum per essere certi che nessuno abbia alterato i codici di controllo
2. verifica con esso dell'integrità del file compresso `install-tl-unx.tar.gz`.

13.4.1. Verifica del checksum

Dai il comando:

```
gpg --verify install-tl-unx.tar.gz.sha512.asc install-tl-unx.tar.gz.sha512
```

con l'output:

```
gpg: Firma effettuata mar 16 giu 2026, 01:49:32 CEST
gpg:                utilizzando la chiave RSA
D8F2F86057A857E42A88106A4CE1877E19438C70
gpg: Firma valida da "TeX Live Distribution <tex-live@tug.org>" [sconosciuto]
gpg: ATTENZIONE: questa chiave non è certificata con una firma fidata.
gpg:          Non ci sono indicazioni che la firma appartenga al proprietario.
Impronta digitale chiave primaria: C78B 82D8 C795 12F7 9CC0  D7C8 0D5E 5D91
06BA B6BC
```

```
     Impronta digitale della sottochiave: D8F2 F860 57A8 57E4 2A88  106A 4CE1
877E 1943 8C70
```

L'importante è leggere alla terza riga `Firma valida...`. Ottimo.

13.4.2. Verifica dell'archivio

Ora che il checksum è sicuro, verifichiamo con esso l'archivio:

```
sha512sum --check install-tl-unx.tar.gz.sha512
```

Il responso è:

```
install-tl-unx.tar.gz: OK
```

Se anche il vostro output mostra OK, il file è integro, sicuro al 100% e pronto per l'uso. Diversamente elimina il file e ripeti il download da un altro mirror CTAN.

13.5. Passo 5: estrazione e installazione

Ora che l'archivio ha superato i controlli di sicurezza, possiamo procedere all'estrazione e all'installazione vera e propria. L'archivio si estrae con l'utility `tar` che genera una cartella in cui ci sposteremo:

```
tar -xzf install-tl-unx.tar.gz
cd install-tl-20*
```

Volendo installare TEX Live in una cartella di sistema, per l'avvio dell'installer sono necessari i privilegi di amministratore che assumiamo tramite il comando `sudo`. Verrà richiesta la password:

```
sudo ./install-tl
```

Scegliamo di installare TEX Live nella directory di sistema `/opt/texlive/2026`. Tutti gli utenti potranno utilizzarla. Nell'interfaccia testuale, alla linea "Enter command:" premi la lettera D per *directories* e invio per confermare.

Nel sottomenù dai il numero 1 e digita la directory `/opt/texlive/2026` e conferma al solito con il tasto invio. Poi R per ritornare al menù principale e I per avviare l'installazione.

I singoli pacchetti saranno scaricati con la protezione del protocollo di rete, dai server ufficiali CTAN. L'operazione può richiedere da 30 minuti a un'ora o più, in base alla tua connessione e alla velocità del computer. Vengono poi generati i file di formato e di ricerca file.

Al termine apparirà il testo di benvenuto:

```
Benvenuto in TeX Live!
See /opt/texlive/2026/index.html for links to documentation.

The TeX Live web site (https://tug.org/texlive/) provides all updates
and corrections. TeX Live is a joint project of the TeX user groups
around the world; please consider supporting it by joining the group
best for you. The list of groups is available on the web
```

```
at https://tug.org/usergroups.html.

Add /opt/texlive/2026/texmf-dist/doc/man to MANPATH.
Add /opt/texlive/2026/texmf-dist/doc/info to INFOPATH.
Most importantly, add /opt/texlive/2026/bin/x86_64-linux
to your PATH for current and future sessions.

Logfile: /opt/texlive/2026/install-tl.log
```

13.6. Passo 6: eliminazione temp-tl

Eliminiamo la directory temporanea non più necessaria.

```
cd ~
rm -rf temp-tl
```

13.7. Passo 7: configurazione

A questo punto TeX Live deve essere configurata manualmente non potendo contare sugli automatismi del sistema di pacchetti proprio della distribuzione Linux.

Istruiamo Debian/Ubuntu su dove trovare i programmi di TeX Live appena installati.

Apri il file `.profile` nel tuo editor di testo preferito. Si tratta di un file *nascosto* della `home` dell'utente. Premendo `CTRL + H` si attiva/disattiva la visualizzazione delle rispettive icone.

Vai in fondo al file e incolla le seguenti righe:

```
# addition for TeX Live 2026
export PATH=/opt/texlive/2026/bin/x86_64-linux:${PATH}
```

La verifica del passo successivo può essere fatta anche senza riavviare la sessione. Prepara una finestra di terminale con cui provare la configurazione con il comando:

```
source ~/.profile
```

13.8. Passo 8: Test installazione

Verifichiamo che il sistema riconosca correttamente LaTeX. Digita nel terminale:

```
lualatex --version
```

Se compare il testo che inizia con `This is LuaHBTeX, Version 1.24.0...` la procedura è terminata con successo. Il tuo ambiente TeX Live è installato, aggiornato e sicuro.

13.9. Passo 9: aggiornare TeX Live

L'utility di aggiornamento `tlmgr` ha necessità dei privilegi di amministratore dovendo scrivere in una directory di sistema. Se lanciamo il comando con `sudo` l'utility non verrà trovata perché non saranno riconosciti i percorsi nell'ambiente di esecuzione amministrativo. La soluzione più semplice consisterebbe nel digitare *tutto* il percorso ma non è comodo.

Basta creare un *alias* all'interno del file `.bash_aliases`. Si tratta di un nome che prende il posto di un comando lungo utilizzato spesso.

Apri il file `.bash_aliases` nella `home` dell'utente. Se non esiste crealo, e aggiungi le righe seguenti per creare l'alias `sutlmgr`:

```
# addition for TeX Live 2026
alias sutlmgr="sudo /opt/texlive/2026/bin/x86_64-linux/tlmgr"
```

Potrai usare `sutlmgr` per acquisire gli aggiornamenti dei pacchetti. I comandi più comuni sono:

```
# per controllare la presenza di aggiornamenti:
tlmgr update --list

# per aggiornare tutto:
sutlmgr update --all

# per controllare se un pacchetto è installato:
tlmgr show <nome_del_pacchetto>
```

14. LaTeX online

Le tecnologie web si sono evolute davvero tanto fino a rendere possibile per l'utente utilizzare sofisticate applicazioni all'interno di un browser. Anche per LaTeX esistono servizi web che offrono:

- compilazione di sorgenti senza bisogno di installare e mantenere il sistema localmente sul proprio PC,
- e, cosa molto più importante, la possibilità di lavorare in più persone a distanza e contemporaneamente sullo stesso file sorgente.

Per LaTeX esistono diversi editor online con caratteristiche più o meno complete: Overleaf, Cocalc, TeXPage, Papeeria e molti altri.

I più diffusi Overleaf e Cocalc possono essere usati, sia pure con alcune limitazioni, del tutto gratuitamente.

14.1. Overleaf

Il sito più famoso, che ha assorbito altri fornitori di servizi analoghi, è certamente Overleaf che permette nella versione gratuita di avere un numero illimitato di progetti e un collaboratore.

Una volta fatto il *log-in* abbiamo la possibilità di creare "progetti" che possono essere condivisi con i nostri collaboratori (il numero di utenti con cui è possibile condividere dipende dal piano, gratuito o a pagamento). La condivisione può avvenire in due modi: il collaboratore può essere editor del progetto o solo visualizzatore.

L'editor di Overleaf è *user friendly* offrendo il completamento automatico (specificando se si tratta di un comando, un ambiente o altro) e la composizione automatica dei riferimenti incrociati.

In Overleaf troviamo tantissimo materiale: sia sull'editor che su LaTeX in generale, sia per utenti inesperti che per utenti con necessità più avanzate.

14.2. Cocalc

Cocalc non è solo un editor/compilatore per LaTeX online, ma mette a disposizione degli utenti moltissimi strumenti, tra cui un ambiente Linux (Ubuntu 24.04) abbastanza completo, il software statistico R, un compilatore Julia, l'ambiente Sagemath, il software matematico Octave, librerie Python, intelligenza artificiale, e molto altro.

Queste caratteristiche ne permettono un utilizzo molto avanzato che permette di lavorare in più collaboratori ai nostri progetti. Tuttavia se non si sottoscrive un abbonamento si hanno a disposizione risorse molto limitate in termini di memoria e CPU disponibile che ne rendono l'uso molto lento.

14.3. Per approfondire

Ottima fonte sugli strumenti web per l'editing dei sorgenti del sistema LaTeX e la loro compilazione è il recente articolo "Usare LaTeX online" sulla rivista del GuIT ArsTeXnica al numero 36 del 2025 a firma di Grazia Messineo e Salvatore Vassallo.

15. Primo documento

Proviamo a scrivere il semplice testo "Ciao mondo." su una pagina bianca. Per prima cosa, serve un editor di testi. In teoria se ne può scegliere uno qualsiasi. Bene, sceglietene uno e create un file di testo dal nome `primodocumento.tex`.

Adesso copiate nel file queste righe di codice:

```
\documentclass{article}
\begin{document}
Ciao mondo.
\end{document}
```

Queste quattro semplici righe di codice istruiscono il compositore a

- creare un documento di uno stile predefinito, in questo caso la classe article, ma ce ne sono tanti altri i cui dettagli possono essere approfonditi in un secondo momento
- iniziare un documento `\begin{document}`
- stampare la stringa `Ciao mondo.`
- terminare il documento `\end{document}`

Il linguaggio che si usa nei sorgenti LaTeX è un linguaggio di markup: le informazioni che servono a comporre i documenti sono "marcate" da appositi comandi.

Ecco il risultato della compilazione del vostro primo sorgente: primodocumento.pdf.

Facile!

> **Nota**
>
> la distribuzione TeX Live comprende l'editor TeXworks semplicissimo da usare

16. Matematica

LaTeX è famoso per la qualità con la quale si possono comporre formule matematiche. Ecco un esempio:

Ciao formula: $E = mc^2$. Ora, ancora testo.

ottenuto con il codice seguente:

```
Ciao formula: \( E = m c^2 \). Ora, ancora testo.
```

Nell'esempio precedente una breve formula matematica è stata inserita all'interno del testo corrente. In gergo si dice che quella è una formula *inline*. La matematica nei documenti può comparire anche in regioni a sé stanti: le cosiddette *formule in display*. Eccone un esempio:

$$f(n, \lambda) = \left(\sqrt{1 + \frac{1}{n^4}} - 1 \right)^{\lambda}$$

ottenuta con il codice seguente:

```
\[ f(n, \lambda) = \left( \sqrt{1 + \frac{1}{n^4}} - 1 \right)^{\lambda} \]
```

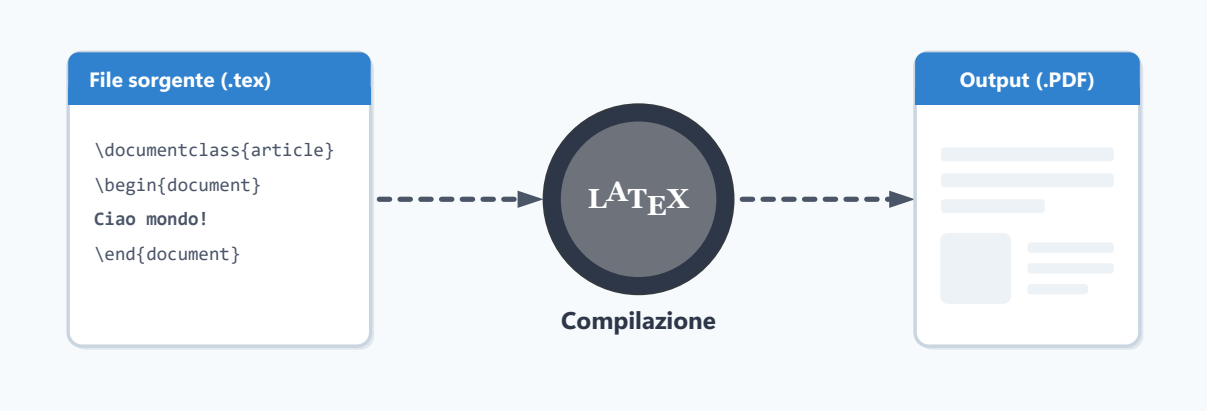
Per i due tipi di ambienti matematici ci sono i corrispondenti delimitatori, le coppie `\(` , `\)` e `\[` e `\]` , il primi per la matematica inline e i secondi per quella in display.

All'interno della coppia, ci sono i comandi di markup che servono a comporre le varie parti della formula: ad esempio `\sqrt{...}` serve a disegnare il simbolo di radice quadrata e a comporre sotto radice i simboli tra graffe.

Un altro esempio di markup è il comando che compone la frazione (fraction): nel codice `\frac{1}{n^4}` il primo argomento tra parentesi graffe è il numeratore e il secondo è il denominatore.

17. Compilare il sorgente

I programmi TeX sono a riga di comando. Compilare un sorgente significa eseguire il compositore in un terminale passandogli il nome del file.



Il file sorgente non viene mai modificato dal compositore. L'effetto della compilazione è quello di creare il file del documento composto in `PDF` e altri file secondari o ausiliari.

Il comando di compilazione va dato in un terminale scrivendo di seguito:

1. il nome del compositore, generalmente `lualatex`
2. uno spazio di separazione

3. il nome del file con o senza estensione
4. il tasto invio per eseguirlo

```
lualatex primodocumento.tex
```

Il processo di compilazione è accompagnato dalla stampa in console di numerosi messaggi informativi che si succedono rapidamente a schermo: pacchetti caricati, numero di pagine composte, font utilizzati, avvertimenti o messaggi di errore.

Al primo errore che incontra il compilatore si ferma e attende istruzioni dall'utente dal terminale. Per esempio premendo `H` e invio verranno stampate informazioni più dettagliate sull'errore. Per interrompere la compilazione e indagare su quanto è successo premere `CTRL + Z + Invio`.

17.1. Estensione e codifica

L'estensione dei file sorgente è `.tex` e la loro codifica Unicode `UTF-8`. In questo modo il sorgente viene compilato allo stesso modo su diversi sistemi operativi. Possiamo editare e compilare lo stesso sorgente in ufficio con Windows, a casa con Linux e da un amico con macOS.

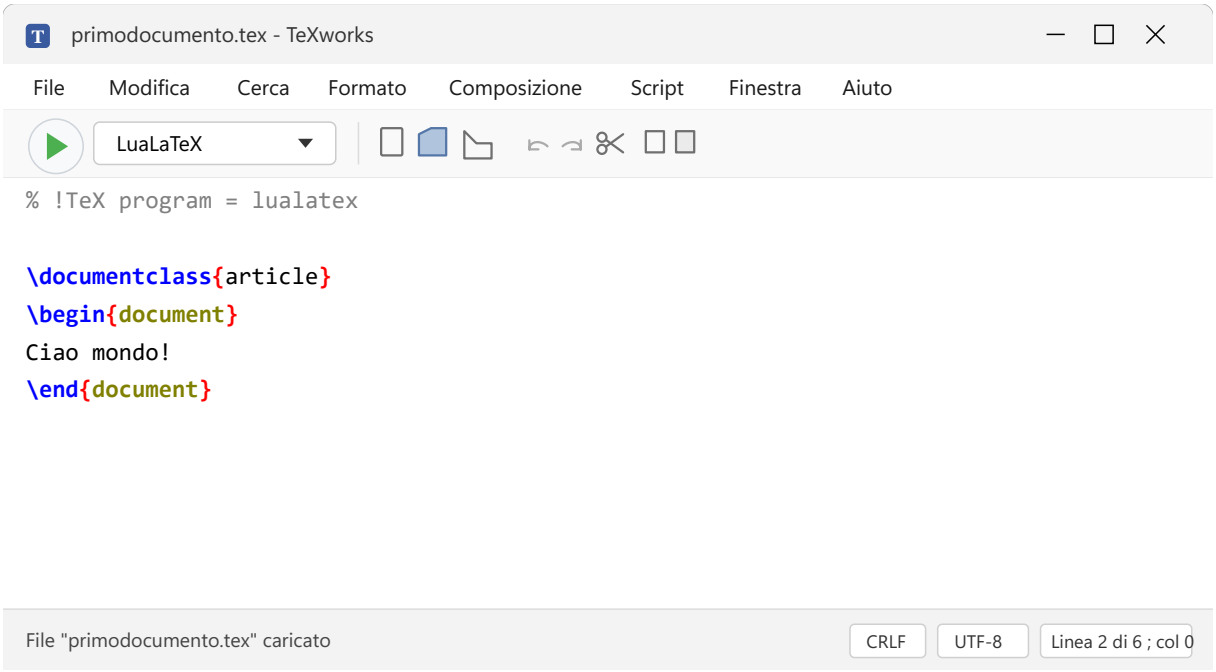
17.2. Compilare con TeXworks

Gli editor di testo dedicati a LaTeX hanno molte facilitazioni tra cui la compilazione con un click. Di più, spesso il PDF è mostrato a fianco del testo sorgente ed è possibile fare click in un punto per spostarsi nel corrispondente codice.

Uno degli editor più utilizzati è TeXworks. Per impostarlo a compilare con `lualatex` basta scrivere come prima riga del file sorgente questo commento:

```
% !TeX program = lualatex
```

Queste righe di commento vengono chiamate *righe magiche* e influenzano l'editor su diversi aspetti uno dei quali è appunto stabilire quale deve essere il compilatore al click sul bottone nella barra degli strumenti o alla pressione della combinazione di tasti `CTRL + T`.



Anche in TEXWorks dal menù `Finestra` > `Mostra il pannello di output` possono essere consultati i messaggi di output della compilazione.

18. Compositori

Assieme all'originale compositore `tex` ne esistono molti altri. `luatex` è uno degli ultimi arrivati e, come dice il nome, include un interprete Lua, un linguaggio di programmazione semplice, potente ed efficiente.

Per avvalersi di Lua e delle altre nuove funzionalità come quella della compatibilità con i font Open Type, è consigliabile utilizzare il compositore `luatex` con il formato LATEX ovvero il comando `lualatex`.

Il codice del sorgente dovrà essere adattato se si vuole cambiare compositore, e anche di molto se cambia anche il formato. Lo sviluppo del sistema tende invece a garantire la retrocompatibilità perciò vecchi file sorgente dovrebbero continuare a compilare.

19. Preambolo e corpo

Un sorgente LATEX si compone di due parti:

- il *preambolo* contiene
  - d'obbligo il caricamento della *classe*
  - se utili, il caricamento di pacchetti, le definizioni di macro personali e le impostazioni particolari
- il *corpo* contiene gli oggetti da comporre sulla pagina

Nel preambolo non è ammesso alcun materiale che sia possibile comporre sulla pagina, per esempio del testo. Nel corpo non è ammesso il caricamento di pacchetti.

Questa partizione è riconoscibile in questo sorgente:

```
% preambolo
\documentclass[a4paper]{article}
\usepackage[italian]{babel}
```

```
\usepackage{booktabs}

\usepackage{siunitx}
\sisetup{output-decimal-marker={,}}
\usepackage{graphicx}

\begin{document}
Corpo del documento.

Il testo del documento va quì.
\end{document}
```

Nel preambolo la prima cosa da fare è definire il tipo di documento caricando una *classe*. La classe stabilisce tutta una serie di impostazioni che caratterizzano l'aspetto tipografico del documento, a seconda di una specifica tipologia.

Fanno parte di LaTeX tre principali classi predefinite:

- `article`
- `book`
- `report`

Nel preambolo possiamo inserire righe bianche per separare il codice, e lasciare commenti per ricordarci l'utilità di un pacchetto.

20. Pacchetti

TeX prevede che si possano programmare nuove funzioni. LaTeX stesso ne è un esempio che a sua volta prevede la possibilità di scrivere *pacchetti d'estensione*. I pacchetti sono la ricchezza di LaTeX. Un patrimonio di migliaia di nuove possibilità compositive, funzioni e facilitazioni.

Possiamo classificarli a seconda dello scopo:

- **impostazioni**: font, gabbia del testo, testatine, …
- **localizzazione linguitica**: composizione in una lingua, regole di cesura a fin di riga, convenzioni tipografiche, documenti multilingua, …
- **potenziamento LaTeX**: sommari complessi, matematica avanzata, unità di misura, poesie in versi, schemi di partite a scacchi, …

I pacchetti si caricano esclusivamente nel preambolo con il comando `\usepackage{}`. La sintassi è la seguente:

```
\usepackage[opzioni]{nome-pacchetto}
```

20.1. Pacchetti di uso comune

Per diverse ragioni ci sono funzioni essenziali che non appartengono al nucleo di LaTeX. Per citarne una l'inclusione di immagini. A ciò rimediano i pacchetti di uso comune. Eccone un elenco:

20.2. `babel`

Compatibile con LuaLaTeX, `babel` è il pacchetto di eccellenza per consentire al compositore di applicare le regole tipografiche specifiche per una lingua.

Si utilizza nella modalità mono-lingua:

```
%
\usepackage[italian]{babel}
%
\begin{document}
Oltre che alle regole di sillabazione delle parole per la cesura a fin di riga
per l'italiano, ci sono altre specificità come i <<caporali>>.
\end{document}
```

Oppure in quella per più lingue. Nelle opzioni la lingua principale deve essere l'ultima:

```
%
\usepackage[english,italian]{babel}
%
\begin{document}
Questa è una parte dove sono in vigore le regole per l'italiano.

% brano nella lingua secondaria
\begin{otherlanguage}{english}
Considered this part to be compiled with the \emph{english} rules.
\end{otherlanguage}
\end{document}
```

20.3. `graphicx`

Questo pacchetto è in grado di inserire immagini nei formati `PDF`, `PNG`, `JPG` ed `EPS` eventualmente applicando un fattore di scala, una rotazione o altri effetti come il ritaglio sui bordi.

Il pacchetto non inserisce direttamente l'immagine nel file `PDF`, ma è capace di aggiungerla al catalogo degli oggetti interni e inserirla sulla pagina solo come riferimento. Così se includiamo più volte la stessa immagine il documento finale rimarrà sostanzialemnete della stessa dimensione.

Il comando fondamentale è `\includegraphics{}` che ha parecchie possibili opzioni.

```
%
\usepackage{graphicx}
%
\begin{document}
\includegraphics[scale=0.80,angle=90]{planimetria}
\end{document}
```

20.4. Gestione della pagina

- `geometry`: definizione intelligente della *gabbia del testo*
- `fancyhdr`: impostazione delle testatine anche differenziate per le pagine pari o per quelle dispari

20.5. `amsmath`

Un pacchetto dell'American Mathematical Society offre possibilità compositive avanzate per la matematica.

20.6. Grafica programmata

- `pict2e` : estensione della grafica vettoriale nativa di LaTeX
- `tikz` : grafica vettoriale
- `xcolor` : gestione avanzata dei colori

20.7. Oggetti flottanti

- `caption` : regola la composizione delle didascalie degli oggetti mobili
- `float` : introduce nuove tipologie di oggetti mobili con il relativo sommario
- `subfig` : gestisce oggetti mobili composti da più elementi

20.8. Liste

- `enumitem` : composizione di liste

20.9. Tabelle

- `booktabs` : filetti per tabelle tipograficamente corrette

20.10. Varie ed eventuali

- `hyperref` : URL a documenti Internet esterni
- `emptypage` : toglie testatine e numeri di pagina da pagine bianche
- `fancyvrb` : inclusione di testo verbatim. Si tratta di testo incluso nel documento così com'è nel file sorgente senza alcuna interpretazione
- `microtype` : attiva la micro-tipografia che offre il migliore riempimento della pagina
- `siunitx` : unità di misura e formattazione di numeri
- `pgfplots` : grafici di funzione e visualizzazione dati
- `pdfpages` : inserimento di documenti PDF all'interno di quello in composizione

21. Documentazione di sistema

La documentazione è essenziale per un programma come LaTeX. Tutto si basa sull'utilizzo di comandi che vanno usati secondo una precisa sintassi. Ci sono due differenti tipi di documentazione:

- quella di *sistema* che descrive i comandi di LaTeX o quelli dei pacchetti che vengono distribuiti da CTAN,
- quella per comprendere la filosofia del linguaggio di markup, l'arte della tipografia, le potenzialità del compositore

Del primo tipo, se la vostra installazione comprende un certo pacchetto, la sua documentazione può essere visualizzata a schermo attraverso la comoda utility `texdoc` con un comando da terminale:

```
texdoc babel
```

che aprirà a video la documentazione del pacchetto `babel` dedicato alla composizione del testo con le regole di una certa lingua.

In alternativa è possibile far ricorso al sito texdoc.org.

22. Organizzazione di un progetto

Per i documenti complessi è utile suddividere il documento in più file sorgente per migliorarne la gestione:

- file più piccoli facilitano l'editing dei contenuti e la concentrazione sulle singole unità,
- la struttura del documento si delinea con maggior chiarezza,
- in un gruppo di lavoro diverse persone lavorando su differenti sorgenti collaborano alla stesura di uno stesso documento,
- modificare l'ordine dei capitoli si riduce a spostare singoli comandi,
- i tempi di compilazione possono ridursi sensibilmente.

Con gli editor di testo moderni gli svantaggi sono invero ridotti al minimo. Si possono tenere aperti molti file allo stesso tempo o fare ricerche su tutti i file di una directory. I *plug-in* possono gestire nomi e riferimenti presenti in altri file.

22.1. Struttura dell'archivio

I sorgenti assumono una struttura nidificata che rispecchia quella del documento: con un livello la suddivisione è in capitoli o in sezioni. Con due livelli la suddivisione è in parti e sotto-parti.

Ogni unità può essere un singolo file sorgente oppure più file sorgente in una directory assieme a risorse locali come immagini, grafici, o bibliografie.

L'importante è che il progetto resti ordinato mano a mano che evolve e che si possa riconoscere il ruolo di un dato file semplicemente leggendone nome e posizione.

22.2. Assemblare file sorgente

Ogni documento ha un unico sorgente principale detto `main file`. All'interno del `main` possono essere inclusi altri file tramite i comandi `\input{}` o `\include{}`.

Quando il compositore compila il `main file` processerà i file esterni inclusi in un dato punto come se fossero stati digitati in quella posizione.

Questo ha una conseguenza importante perché nei file secondari i percorsi dei file di immagini devono riferirsi alla posizione dove si trova il file principale e non a quella del file secondario.

La cartella dove risiede il file principale è detta `root` o `home` del progetto.

22.3. Il comando `\input{}`

Questo comando come argomento accetta il percorso del file da inserire.

Nel percorso i nomi dei file possono essere separati da caratteri `/` (slash) indipendentemente dal sistema operativo, partendo dalla cartella di `root` del progetto. L'estensione `.tex` può essere omessa.

Qui un esempio di main file per una ipotetica biografia di un poeta (i sei file dei capitoli si trovano in una sotto cartella `chapter`):

```
% main file
\documentclass{book}

\begin{document}
\input{chapter/00-introduzione}
\input{chapter/01-infanzia}
\input{chapter/02-primo-amore}
\input{chapter/03-il-capolavoro}
\input{chapter/04-anni-guerra}
\input{chapter/99-conclusioni}
\end{document}
```

e questa la corrispondente struttura dei file:

```
home-progetto/
├── main.tex                  # Sorgente principale
├── chapter/
│   ├── 00-introduzione.tex   # sorgente di capitolo
│   ├── 01-infanzia.tex       # sorgente di capitolo
│   ├── 02-primo-amore.tex    # sorgente di capitolo
│   ├── 03-il-capolavoro.tex  # sorgente di capitolo
│   ├── 04-anni-guerra.tex    # sorgente di capitolo
│   └── 99-conclusioni.tex    # sorgente di capitolo
├── main.pdf                  # documento finale compilato
├── main.aux                  # file ausiliario della compilazione
├── main.log                  # file di log della compilazione
├── main.toc                  # dati ausiliari del sommario
└── README.md                 # eventuale file di testo per note varie
```

> **Nota**
>
> Dare buoni nomi ai file e alle cartelle è un'utile arte. Evita di usare caratteri spazio o accentati. Se iniziano con una numerazione saranno mostrati nell'ordine numerico assegnato e non con quello alfabetico dei semplici nomi.

22.4. Il comando `\include{}`

In LaTeX esiste anche il comando `\include{}` pensato per compilare documenti complessi suddivisi in capitoli. Con questo comando è possibile compilare in modo selettivo i capitoli così da limitare il tempo di compilazione mantenendo riferimenti corretti alle parti escluse.

Lo scenario è quindi quello di un documento complesso, come una tesi per esempio, che richiede compilazioni frequenti ogni volta che si desidera controllare il risultato. A mano a mano che si inserisce contenuto i tempi si allungano.

Dunque è utile compilare solamente una selezione di capitoli, magari anche uno solo. Nel preambolo `\includeonly{}` imposta `\include{}` a inserire solo i file specificati. Ecco un esempio di codice:

```
% main file
\documentclass{book}

\includeonly{chapter/02-primo-amore,chapter/03-il-capolavoro}

\begin{document}
\include{chapter/00-introduzione}
\include{chapter/01-infanzia}

% unici due capitoli effettivamente inseriti
\include{chapter/02-primo-amore}
\include{chapter/03-il-capolavoro}

\include{chapter/04-anni-guerra}
\include{chapter/99-conclusioni}
\end{document}
```

A differenza di `\input{}` e a conferma che `\include{}` è stato pensato per includere file di capitolo, quest'ultimo inserisce sia prima che dopo il contenuto del file un'interruzione di pagina, non può essere usato nel preambolo e non può essere annidato, ovvero non è possibile includere file che a sua volta contengono chiamate `\include{}`.

23. Il linguaggio di markup

Immagina di dover scrivere un programma che componga un testo tipografico elaborando semplici sequenze di caratteri e che tra questi ci sia il *titolo* del documento.

Il titolo dovrà essere messo in risalto per esempio con un font più grande a centro pagina. Invece di dire al programma *come* comporre il titolo potremo semplicemente dirgli che quei carratteri *sono* un titolo.

Potendo inserire informazioni solo attraverso caratteri, viene naturale *marcare* i caratteri del titolo con altri che ne dichiarino la funzione. Tra tante vengono in mente queste tre soluzioni:

```
1 - sintassi "chiave: valore":
title: 1984

2 - sintassi "chiave(valore)":
title(1984)

3 - sintassi "chiave valore endchiave":
title 1984 endtitle
```

La marcatura crea una regola che fa emergere un *linguaggio*. I caratteri diventano *codice* conforme a prestabilite regole grammaticali.

23.1. Il carattere di escape

Stando a questa idea, alcuni caratteri rappresentano un *testo* mentre altri una *marcatura*. I caratteri svolgono due ruoli tra loro molto diversi: il testo è parte del contenuto del documento mentre la marcatura è parte della presentazione tipografica.

Fino a ora tuttavia non è escluso che marcatura e testo non si possano confondere: nella seconda riga di questo brano i caratteri `titolo: 1984` per l'Autore sono evidentemente testo semplice ma per il programma che lo elabora si tratta del titolo `1984` da comporre come tale.

```
Sto scrivendo un libro ma non sono sicuro
del titolo: 1984 è l'anno di questo futuro
distopico in cui si svolge il romanzo...
```

Ogni ambiguità può essere esclusa introducendo un carattere speciale detto *carattere di escape*, un simbolo riservato alla definizione di una marcatura preferibilmente usato molto raramente.

La barra rovescia o backslash `\` è una scelta conveniente anche perché sulla tastiera corrisponde a uno dei pulsanti in alto a destra.

Il titolo del romanzo di George Orwell preso a esempio si definirebbe così:

```
1. sintassi \chiave: valore
\title: 1984

2. sintassi \chiave(valore)
\title(1984)

3. sintassi \chiave valore \endchiave
\title 1984 \endtitle
```

Delle tre marcature la due è interessante per questi motivi:

- non ha il difetto della uno dove sulla stessa riga è ammesso solo il valore
- è più compatta o meno prolissa della tre
- da l'idea di funzione con il suo argomento

> **Nota**
>
> In LaTeX esiste il comando `\title{}` definito nelle classi standard, ma non ha lo scopo descritto in questa sezione. La macro ha solo il compito di scrivere in memoria il titolo che poi verrà effettivamente composto assieme ad altre informazioni dalla macro `\maketitle` che non accetta argomenti

23.2. Le macro di LaTeX

In LaTeX il backslash `\` è il carattere di escape e la sintassi più comune è proprio la due con l'argomento delimitato da parentesi graffe `{` e `}`. Anche questi due caratteri di delimitazione sono poco usati e non sono mai interpretati come testo.

> Emerge così il primo elemento del linguaggio di markup di LaTeX: il *comando*. La sequenza dei caratteri di un comando prevede `\` + nome marcatura + eventuale argomento. Il comando è chiamato anche *macro*.

Alcune considerazioni:

- una macro e il corrispondente effetto sono concetti separati: un titolo può essere composto in molti modi differenti ma il codice dichiarativo rimane identico
- l'Autore definisce oggetti tipografici esprimendo *astrazioni* rispetto ai caratteri digitati
- le proprietà tipografiche di una macro possono dipendere da parametri o da elaborazioni di parametri consentendo dinamiche compositive arbitrariamente complesse

La diversità rispetto a un Word Processor dovuta all'uso di un linguaggio per esprimere i contenuti del documento è netta.

Un sistema di composizione tipografica basato sul codice non è per niente desueto come dimostra il numeroso gruppo di linguaggi di markup disponibile oggi a cominciare da HTML, XML, Markdown, AsciiDoc, il recente Typst e i formati per la rappresentazione dei dati strutturati come JSON, YAML, o TOML.

23.3. Tutto così semplice?

Il linguaggio di markup di LaTeX è qui presentato in modo semplificato per gli scopi della guida. Molti dettagli importanti sono stati taciuti come i codici di categoria, la costruzione dei gruppi, i token e molto altro.

È un po' come ammettere che un *utente* LaTeX lo utilizza in modo proficuo anche senza conoscere i dettagli mentre un *esperto* è capace di qualsiasi costruzione, ma questo è proprio lo scopo di LaTeX stesso rispetto al vero motore di composizione sottostante TeX.

24. Macro

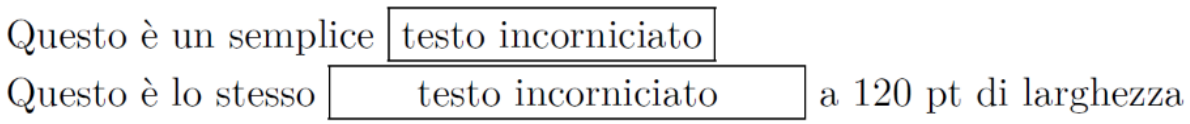
In LaTeX la sintassi di una macro si compone di backslash `\` seguito da un nome e da eventuali argomenti tra parentesi graffe.

Le macro possono accettare opzioni facoltative che ne modificano il comportamento. Le opzioni devono essere inserite prima dell'argomento e tra parentesi quadre `[]`. Come esempio consideriamo la macro `\framebox[][]{}` che incornicia un testo:

```
Questo è un semplice \framebox{testo incorniciato}

Questo è lo stesso \framebox[120pt]{testo incorniciato}
a 120 pt di larghezza
```

Che produce:



24.1. Ambienti

In LaTeX oltre alla macro esiste un secondo elemento molto espressivo del linguaggio: l'*ambiente*. Un ambiente inizia con `\begin{nome-ambiente}` e termina con `\end{nome-ambiente}`. All'interno il codice si riferirà alla definizione di un specifico oggetto.

Per esempio, per costruire un elenco numerato si fa ricorso all'ambiente `enumerate`:

```
\begin{enumerate}
\item prima riga
```

```
\item seconda riga
\item terza riga
\end{enumerate}
```

Anche gli ambienti possono avere argomenti e/o opzioni. Queste informazioni vanno scritte con graffe per i primi o con quadre per le seconde, al seguito della macro di apertura. Come esempio inseriamo una tabella tramite l'ambiente `tabular`. L'argomento obbligatorio definisce gli allineamenti per ciascuna colonna della tabella con i codici: `l` per "a sinistra" (left), `r` per "a destra" (right), e `c` per "al centro":

```
\begin{tabular}{lcr}
prima cella a sinistra & seconda cella centrata & terza cella a destra \\
1 & 2 & 3\\
4 & 5 & 6
\end{tabular}
```

Macro e ambienti possono anche comprendere una variante chiamata versione *asteriscata* poiché al nome si aggiunge un asterisco per produrne la versione alternativa.

Un esempio è l'ambiente `equation` che compone in display il codice matematico numerandolo in automatico . Il pacchetto `amsmath` aggiunge la variante asteriscata che per così dire spegne proprio la numerazione:

```
% ambiente standard: numerazione si
\begin{equation}
f(x) = \frac{x^2 + 1}{x^2 - 1}
\end{equation}

% ambiente asteriscato: numerazione no
\begin{equation*}
f(x) = \frac{x^2 + 1}{x^2 - 1}
\end{equation*}
```

24.2. Spazi e ritorni a capo

Uno spazio è solo un simbolo di separazione tra parole. Non decide l'effettiva distanza tra queste nel testo composto che sarà invece assegnata dal compositore secondo i criteri di leggibilità propri della tipografia.

Nel sorgente gli spazi e i ritorni a capo hanno una particolare interpretazione da parte del compositore secondo queste regole:

1. gli spazi a inizio riga sono ignorati
2. più di un carattere spazio è interpretato come un solo carattere spazio
3. lo spazio dopo una macro senza argomento o senza altro separatore è ignorato
4. un ritorno a capo singolo è interpretato come uno spazio
5. più di un ritorno a capo è interpretato come separazione tra due capoversi

Questi capoversi differenti nel codice danno identico risultato in composizione:

```
Ecco un testo molto simpatico per capire come
sono interpretati in \LaTeX{} sia gli spazi che
i ritorni a capo.
```

```
Ecco un testo                    molto simpatico
    per capire come sono interpretati in \LaTeX{}
      sia   gli   spazi
        che
          i ritorni a capo.
```

Anche i caratteri di tabulazione possono essere utilizzati con lo stesso significato di uno spazio ma normalmente non si usano.

Capita spesso di digitare per errore due spazi tra due parole anziché uno. Mentre per i Word Processor essi saranno tutti significativi, per LaTeX non farà differenza. E ancora, se nei Word Processor siamo abituati (forse male) a spostare il testo sulla riga aggiungendo caratteri spazio, in LaTeX ciò non funzionerà e dovremo usare altri mezzi.

In LaTeX capita invece di fare l'errore contrario: eliminare invece che aggiungere spazio. Ciò avviene quando scriviamo una macro senza argomenti come `\LaTeX` che produce il logo caratteristico.

Lo spazio dopo la macro verrà ignorato come da regola 3 così tra il logo e la parola successiva non ci sarà separazione. Ecco la soluzione:

```
% lo spazio dopo la macro senza argomenti non ha effetto:
Usiamo \LaTeX per migliorare i nostri documenti.

% rimedio 1: coppia di graffe:
Usiamo \LaTeX{} per migliorare...

% rimedio 2: backslash + spazio:
Usiamo \LaTeX\ per migliorare...
```

Anche i ritorni a capo — se singoli — sono semplici separatori di parole, l'ultima della riga precedente con la prima di quella successiva, perciò si è liberi di formattare il testo di un capoverso su un'unica riga oppure di spezzarlo su più righe consecutive.

Per le righe molto lunghe solitamente gli editor mostrano il testo come se fosse su più righe per evitare di dover scorrere in orizzontale la finestra di visualizzazione, una modalità detta di *word wrapping*.

Se un sorgente è sottoposto al controllo di versione di `git` tutte le modifiche sono tracciate e visibili con una colorazione rosso/verde. Questo confronto tra versioni funziona bene se le righe sono corte, diciamo di una lunghezza non superiore a 80 caratteri. Così conviene spezzare a questa lunghezza il capoverso con dei ritorni a capo singoli, operazione che certi editor fanno in automatico.

Se si inseriscono più di due ritorni a capo allora il testo successivo diventa un nuovo capoverso. L'aspetto del testo in LaTeX è perciò una sequenza di righe di testo consecutive separate da righe vuote.

24.3. Commenti

I *commenti* sono una particolare categoria sintattica e sono gestiti con un carattere dedicato: il simbolo di percentuale `%`. Quando il compositore legge un carattere di percento lo ignora assieme a tutti i caratteri che lo seguono fino alla fine della riga.

I commenti sono importanti per fare annotazioni nel sorgente e generalmente per:

- inserire le *righe magiche* a inizio file che istruiscono in vari modi gli editor compatibili
- descrivere idee su come proseguire o migliorare il testo di un capitolo
- inserire un segnaposto a indicare il punto il cui si è arrivati nella revisione del testo
- inserire informazioni di data e ora per poter valutare il tempo impiegato nella redazione del documento
- eliminare dal documento finale una parte di contenuto senza cancellarla dal sorgente
- evitare che a fine riga venga inserito uno spazio spurio

Non esiste il commento multiriga ma se lo si vuole ad ogni inizio riga va inserito un `%`. Poco male, gli editor più evoluti lo fanno in automatico oppure rendono immediata l'operazione tramite funzionalità come il *cursore multiplo* così che è sufficiente una sola battuta del carattere `%` per inserirlo a capo di un numero qualsiasi di righe.

In alternativa esiste il pacchetto `comment` di Victor Eijkhout che esclude/include una porzione di testo con l'omonimo ambiente.

25. Formattazione del testo

Oltre che scrittori se siamo noi a comporre il testo ne siamo anche un po' i tipografi. Documento dopo documento si può acquisire competenza in questo campo specialmente se lo strumento software che usiamo ci aiuta per esempio scoraggiando abitudini sbagliate.

LaTeX può renderci più consapevoli dei valori della tipografia sopratutto perché gli esperti sanno costruire macro che sfruttano correttamente le innumerevoli capacità compositive del sistema.

Così ci approcciamo alle prossime pagine sulla formattazione del testo in LaTeX ricordando che applicarla correttamente è anche una questione di coerenza tipografica. Di conseguenza faremo cenno a considerazioni di stile già prodotte dai *maestri*.

25.1. Evidenziare testo con `\emph{}`

L'occasione più comune per evidenziare un testo è l'introduzione nel brano di un nuovo concetto. In LaTeX lo si fa con una l'apposita macro `\emph{}` che accetta come argomento il testo che interessa evidenziare.

Per esempio:

```
Lo scrittore decide \emph{se} e \emph{cosa} evidenziare
della propria prosa. A ben vedere evidenziare è al tempo
stesso sia un significato semantico sia uno stile tipografico.

Lo \emph{stile tipografico} comunica al lettore informazioni
di contenuto che fanno parte della comprensione del testo.
```

Interessante è notare l'effetto di annidare la macro per evidenziare testo all'interno di uno già in evidenza. La macro `\emph{}` tiene conto del contesto:

```
Con \emph{totale fiducia della \emph{qualità} tipografica}.
```

25.2. Le macro per il testo

Il gruppo di macro per il testo iniziano tutte con `\text` a cui segue una coppia di caratteri. Da usare con cognizione di causa, in elenco sono:

- `\texttt{}`: testo monospaziato, `tt` sta per *teletype* e ricorda la telescrivente. Si utilizza per comporre un codice come un nome di una funzione di un linguaggio di programmazione
- `\textit{}`: testo in corsivo `it` sta per *italics*
- `\textsc{}`: testo in maiscoletto, `sc` sta per *small caps*, si utilizza per comporre i nomi nei frontespizi ed è molto elegante
- `\textbf{}`: testo in grassetto, `bf` sta per *bold face*
- `\textsl{}`: testo inclinato, `sl` sta per *slanted*
- `\textrm{}`: testo in tondo, `rm` sta per *roman*, è lo stile principale che si usa per comporre un testo
- `\textsf{}`: testo senza grazie, `sf` sta per *sans serif*

> **Nota**
>
> Nel testo è bene non mescolare troppi stili differenti.

25.3. Incisi

Un *inciso* aggiunge un'informazione secondaria, un chiarimento o il punto di vista di chi parla. Deve avere automia sintattica e essere necessariamente non troppo lungo. L'inciso può essere messo tra parentesi tonde oppure per dargli maggiore enfasi tra trattini lunghi che in LaTeX si formano con tre trattini `---`.

```
In Lua --- il linguaggio di programmazione disponibile
in Lua\LaTeX{} --- è possibile svolgere calcoli con
alta precisione grazie alla libreria \texttt{math}.
```

25.4. Sezionamento del testo

La suddivisione di un testo è annidata: un capitolo può essere suddiviso in sezioni e a sua volta una sezione può essere suddivisa in unità più piccole. Ancora, una sezione può appartenere solamente a una unità del livello superiore.

Non tutti i testi sono strutturati in capitoli o in sezioni: il sezionamento dipende dal tipo di documento:

- un libro è diviso in parti o solamente in capitoli con prologo e unità conclusive
- un articolo è diviso a partire da sezioni
- un report è diviso in capitoli senza distinzioni

In LaTeX il sezionamento non può che essere demandato alla classe che deve definire apposite macro. Le classi standard `book`, `article` e `report` hanno macro a cui passare come argomento il titolo dell'unità e come opzione il corrispondente titolo breve usato per comporre le testatine e il sommario, così da evitare mancanza di spazio sulla riga.

Per esempio, la sintassi per l'introduzione di un nuovo capitolo è

```
\chapter[Titolo breve]{Titolo completo}
```

Questo l'elenco sovrapposto delle macro di sezionamento delle classi standard:

- livelli per documenti molto strutturati come i libri:
  - `\part{}` : inizia una parte del testo
  - `\chapter{}` : inizia un capitolo
  - livelli per documenti mediamente strutturati come gli articoli
    - `\section{}` : inizia una sezione
    - `\subsection{}` : inizia una sotto sezione
    - `\subsubsection{}` : inizia una sotto-sotto sezione
    - `\paragraph{}` : inizia un paragrafo
    - `\subparagraph{}` : inizia un sotto paragrafo, l'unità più piccola della suddivisione

> **Nota**
>
> se ti interessa la documentazione delle classi standard di LaTeX apri una finestra di terminale e digita il comando `texdoc classes`. Nel documento visualizzato a schermo dopo pochi attimi trovi anche tutto il codice che implementa le classi

25.5. Riferimenti incrociati

Dopo aver definito una qualsiasi unità del documento puoi etichettarla con un codice per poi poterla richiamare ovunque nel documento. Il codice di etichetta è libero ma conviene dividerlo in due parti: l'indicazione del livello e una sigla inerente al titolo. Per esempio:

```
% inizio un nuovo capitolo
\chapter{La pedagogia secondo Maria Montessori}
\label{chapter:montessori}
```

Gli editor per LaTeX leggono tutte le etichette dei riferimenti dai sorgenti aiutando l'utente con il loro inserimento automatico.

La macro `\label{}` deve essere inserita subito dopo l'oggetto che si intende riferire, non solo per capitoli o sezioni ma anche per tabelle, figure, formule matematiche, teoremi, lemmi, eccetera.

Per stampare il riferimento a un oggetto si usano due macro principali a cui va passata l'etichetta dell'oggetto: `\ref{}` stampa il numero dell'oggetto e `\pageref{}` stampa il numero della pagina in cui si trova.

Per citare il capitolo precendente si può usare il codice:

```
Nel capitolo~\ref{chapter:montessori} che inizia nella
pagina~\pageref{chapter:montessori} abbiamo delineato...
```

Nel codice noterai i simboli `~` tra parola e macro. Questo particolare carattere inserisce uno spazio insecabile. Ciò vuol dire che LaTeX non potrà considerare quello spazio come possibile punto di cesura a fine riga e ciò evita quelle situazioni dove il numero dell'oggetto o della pagina va alla riga successiva.

Se nel tuo documento PDF al posto dei riferimenti compare un doppio punto interrogativo `??` significa che il riferimento non è stato trovato: possono esserci due casi:

- un errore di digitazione del riferimento o la non corretta definizione dell'etichetta. Correggi i codici del riferimento o inserisci `\label{}` *dopo* la macro di sezionamento
- non sono ancora stati generati i file ausiliari con i dati necessari. Compila una seconda volta il sorgente

I riferimenti incrociati funzionano infatti con iterazioni successive: immagina che in una sezione ci sia un riferimento a una sezione successiva. Alla prima compilazione non è possibile conoscere né il numero né la pagina effettiva di quella specifica sezione che sarà composta più tardi nel processo. Nelle compilazioni successive LaTeX giunto ai riferimenti può leggere i file ausiliari frutto della compilazione precedente dove sono riportati i dati.

Se sono attivi i link, i riferimenti nel documento PDF finale diventano aree su cui si può fare clic con il mouse per spostarsi su un dato capitolo o sezione e funziona anche con il sommario. Per il manoscritto stampato su carta i link si perderanno ma per il documento digitali saranno utilissimi per il lettore.

Elenco delle macro per i riferimenti incrociati:

- `\label{tipo-oggetto:sigla}` : genera un riferimento all'oggetto che sia un capitolo una sezione, una tabella o una figura
- `\ref{tipo-oggetto:sigla}` : stampa il numero dell'oggetto, per esempio 1.2.3 per la sotto sezione 3 della sezione 2 del capitolo 1
- `\pageref{tipo-oggetto:sigla}` : stampa il numero di pagina dell'oggetto

25.6. Oggetti mobili

I contenuti vengono letti in sequenza dal compositore, testi, titoli, figure, tabelle, eccetera, e sistemati uno dopo l'altro a riempire la pagina. Consideriamo tuttavia le figure *fuori testo*.

Questi oggetti sono chiamati *oggetti mobili* perchè LaTeX li sposta rispetto al loro punto d'inserimento nel sorgente relativamente al resto dei contenuti, in una posizione sulla pagina in alto, a tutta pagina al centro o in basso.

Se il testo non potesse "girare" attorno agli oggetti mobili, poiché questi spesso hanno dimensioni notevoli, si creerebbero spazi vuoti a fondo pagina.

Un buon posizionamento degli oggetti fuori testo offre loro importanza nei confronti della lettura e porta equilibrio visivo sulla pagina.

Spesso all'utente non piace come LaTeX sistema gli oggetti mobili. A ben vedere si tratta di equilibrio tra quantità di testo e oggetti fuori testo. Con un po' di esercizio in sede di revisione finale del documento, spostando il punto nel codice dell'oggetto o agendo sui parametri di posizionamento, è possibile imparare a bilanciare le parti.

Le opzioni di posizionamento da usare con parsimonia, semplificando un po', si danno tra quadre con le lettere: `t` per la posizione *top* sulla pagina, `b` *bottom* per il fondo, `p` *page* per il centro pagina tutta dedicata all'oggetto mobile e `h` *here* a indicare una posizione li dove è inserito il codice dell'oggetto compatibilmente con lo spazio disponibile.

Per ciascun tipo, ma se ne possono creare altri, abbiamo un ambiente: `table` per le taelle e `figure` per le figure. Così è possibile creare sommari separati e coerenza tipografica.

25.7. Oggetto mobile `table`

All'interno dell'ambiente `table` ci sono queste macro in ordine (la *didascalia* va posta sopra la tabella):

1. `\centering`: centra la tabella nella pagina senza aggiungere spazi verticali come invece farebbe l'ambiente `center`
2. `\caption{}`: compone la didascalia completa di numerazione
3. `\label{}` raccoglie la numerazione appena generata per i riferimenti all'oggetto
4. la tabella che si può costruire con l'ambiente `tabular` con i filetti orizzontali di `booktabs` o includere un oggetto tabella da altra fonte

Qui di seguito un esempio di una tabella dedotta dal libro del maestro calligrafo Ernesto Casciato dal titolo "Calligrafia: evoluzione e futuro della bellezza scritta" edito nel 2021:

```
% nel preambolo:
% filetti orizzontali per tabular con la giusta spaziatura verticale
% macro: \toprule, \midrule, \bottomrule
\usepackage{booktabs}
```

```
% nel corpo del documento
\begin{table}
\centering
\caption{Alcuni antichi stili scrittori dell'area mediterranea}
\label{tabCalligraphyAncientStyles}
\begin{tabular}{lcl}
\toprule
Stile                  & Età                   & Testimonianza \\
\midrule
Capitalis Monumentalis & \textsc{i} secolo     & Monumenti romani\\
Onciale                & \textsc{iv} secolo    & Testi monastici\\
Textura gotica         & \textsc{xi} secolo    & Libri universitari \\
Cancelleresca          & \textsc{xvi} secolo   & Primo trattato di
calligrafia\\
Corsiva inglese        & \textsc{xviii} secolo & Testi coloniali\\
\bottomrule
\end{tabular}
\end{table}
```

25.8. Oggetto mobile `figure`

Per l'ambiente flottante `figure` la didascalia va posta sotto la figura. Al suo interno abbiamo questa sequenza di macro:

1. `\centering`: centra l'immagine nella pagina senza aggiungere spazi verticali come invece farebbe l'ambiente `center`
2. `\includegraphics[]{}` include l'immagine preferibilmente nel formato vettoriale come il PDF
3. `\caption{}`: compone la didascalia completa di numerazione
4. `\label{}` raccoglie la numerazione appena generata per i riferimenti all'oggetto

Anche qui un esempio di codice da un documento reale — una relazione tecnica — suddiviso tra caricamento dei pacchetti necessari e ambiente `figure`.

```
% carica il pacchetto per importare immagini
\usepackage{graphicx}
```

```
% nel corpo del documento
\begin{figure}
\centering
\includegraphics[width=0.62\textwidth]{image/sezmodel}
\caption{Rendering solido del modello strutturale
dell'edificio visto in sezione verticale trasversale.}
\label{figSezModel}
\end{figure}
```

25.9. Indici (o sommari)

Gli indici (o sommari) si generano in automatico con queste macro:

- `\tableofcontents`: genera l'indice delle unità che compongono il documento così come definite dalle macro di sezionamento
- `\listoffigures`: genera l'indice delle figure fuori testo che fanno capo agli ambienti `figure` se dotati di comandi `\caption`
- `\listoftables`: genera l'indice delle tabelle fuori testo che fanno capo agli ambienti `table` se dotati di comandi `\caption`

25.10. Indice analitico

Normalmente posizionato alla fine del documento l'*indice analitico* è l'elenco delle parole già marcate nel testo con i numeri di pagina a fianco.

L'indice può essere specifico e allora si parla di indice dei nomi, indice dei luoghi, eccetera.

In LaTeX la composizione di un indice analitico può essere automatizzata da pacchetti come `imakeidx`.

25.11. Elenchi

Queste strutture sono grosso modo di due tipi: non numerati e numerati. Corrispondentemente in LaTeX ci sono gli ambienti `itemize` e `enumerate`. All'interno di questi ambienti si inseriscono i vari punti con la macro `\item`.

In questo esempio è mostrata anche la possibiità di creare sottoelenchi combinando i due ambienti:

```
% nel preambolo:
% \GuIT stampa il logo GuIT
\usepackage{guit}
```

```
% nel corpo del documento
\begin{itemize}
\item studiare la Guida \GuIT;
\item iscriversi al gruppo;
\item studiare i pacchetti \LaTeX{} di uso comune
\item studiare \textsf{enumitem} che consente di
```

```
personalizzare gli elenchi.
\end{itemize}
```

26. Esempi di codice

In progress.

Qui di seguito un esempio di codice preso da una relazione tecnica reale:

```
% nel preambolo:
% filetti orizzontali per tabular con la giusta spaziatura verticale
% macro \toprule, \midrule, \bottomrule
\usepackage{booktabs}

% composizione unità di misura con impostazione del separatore decimale
% macro \si{} solo unità di misura, \num{} solo numero \SI{}{} numero con
unità
\usepackage{siunitx}
\sisetup{output-decimal-marker={,}}
```

```
% nel corpo del documento
\begin{table}
\centering
\caption{Carichi per tipo di progetto per la verifica dei solai}
\label{tabCarSol}
\begin{tabular}{clccc}
\toprule
\textsc{id} & Descrizione & Tipo & \(\gamma_\mathrm{slu}\) & Azione in \si{kN/
m^2}\\
\midrule
\(g_1\) & Permanenti strutturali     & \(G_1\)            & \num{1.3} &
\num{3.20}\\
\(g_2\) & Permanenti non strutturali & \(G_2\)            & \num{1.5} &
\num{3.90}\\
\(q\)   & Carico di superficie       & \(Q_\mathrm{k,A}\) & \num{1.5} &
\num{2.00}\\
\bottomrule
\end{tabular}
\end{table}
```

Qui di seguito un esempio di una tabella sui comandi di sezionamento in relazione alle classi standard, adattata da un corso tenuto da Claudio Beccari:

```
% nel preambolo:
% filetti orizzontali per tabular con la giusta spaziatura verticale
% macro: \toprule, \midrule, \bottomrule
\usepackage{booktabs}

% comando per comporre macro
\newcommand*\cmd[1]{{\normalfont\texttt{\string#1}}}
```

```
% nel corpo del documento
\begin{table}
\centering
\caption{Macro di sezionamento delle classi standard di \LaTeX}
\label{tabSectioning}
\begin{tabular}{llcccc}
\toprule
Unità & Comando di & Livello & \multicolumn{3}{c}{Classe \LaTeX{} standard}\\
      & sezionamento &       & \texttt{book} & \texttt{report} &
\texttt{article} \\
\midrule
Parte              & \cmd{\part}`         & -1 & si & si & si \\
Capitolo           & \cmd{\chapter}`      &  0 & si & si & no \\
Sezione            & \cmd{\section}`      &  1 & si & si & si \\
Sottosezione       & \cmd{\subsection}`   &  2 & si & si & si \\
Sotto sottosezione & \cmd{\subsubsection}`&  3 & si & si & si \\
Paragrafo          & \cmd{\paragraph}`    &  4 & si & si & si \\
Sotto paragrafo    & \cmd{\subparagraph}` &  5 & si & si & si \\
\bottomrule
\end{tabular}
\end{table}
```

27. Informazioni sulla guida

- Titolo: Guida a LaTeX
- Sottotitolo: Tutto il necessario per cominciare con LaTeX
- Codice progetto: `guidalatex`
- Online dal: giugno 2026
- Link: https://guitex.org/guide/guidalatex
- Versione: 0.1.13
- Data di rilascio: 06 agosto 2026

Della Guida a LaTeX quì troverai:

- le informazioni per contribuirvi e il gruppo dei contributori
- il record bibliografico `biblatex` per citarla già pronto
- alcune notizie su come è prodotta
- la roadmap per i prossimi sviluppi
- la licenza d'uso e distribuzione

27.1. Contribuire alla guida

Denominato `guidalatex` questo progetto dell'associazione GuIT è libero e Open Source.

Opinioni, suggerimenti, segnalazioni, proposte o contributi sono i *benvenuti*. Se sei un utente che l'ha utilizzata per i primi passi con LaTeX condividi con noi la tua opinione sulla guida: l'innovazione nasce dalle opportunità.

Per contribuire sono previste queste modalità:

- via email all'indirizzo webteam at GuITeX.

- via Pull Request verso il repository ufficiale della guida che si trova al link repository `guidalatex`.

Per condividere testo considera che il formato nativo dei sorgenti della guida è il Markdown.

27.2. Contributori

Questa è la lista delle persone che hanno contribuito o che contribuiscono a questa guida su LaTeX, con materiale, osservazioni o nuove idee.

Un grande ringraziamento a loro!

- Claudio Beccari
- Giovanni Nuccetelli
- Agostino De Marco
- Grazia Messineo
- Salvatore Vassallo
- Roberto Giacomelli robitex

27.3. Citare la guida: entry biblatex

```
@manual{guit:it-guidalatex,
    title        = {Guida a \LaTeX},
    subtitle     = {Tutto il necessario per cominciare con \LaTeX},
    author       = {GuIT},
    date         = {2026-08-06},
    year         = {2026},
    version      = {0.1.13},
    langid       = {italian},
    url          = {https://guitex.org/guide/guidalatex/},
    urldate      = {2026-08-06},
    organization = {GuIT, Gruppo Utilizzatori Italiani di \TeX},
    series       = {Guide del GuIT},
}
```

27.4. Produzione della guida

Questa guida è costruita con `mdbook` a partire da file testuali in formato Markdown.

Questi file sono tenuti sotto *controllo di versione* tramite `git` e resi pubblici in questo repository Github.com.

Un particolare sorgente Typst converte tutti i sorgenti Markdown per produrre una versione a stampa sperimentale della guida.

Esiste anche una versione eBook dell guida scaricabile da questo link, sperimentale ovviamente.

27.5. Roadmap

- [x] Parti introduttive
- [x] Procedure d'installazione di TeX Live per i tre sistemi operativi Desktop
  - [x] screenshot dell'installazione macOS
- [x] Primo documento
- [x] Paragrafo sui vantaggi di LaTeX rispetto ai Word Processor

In lavorazione:

- [x] Capitolo Pacchetti
  - [ ] da completare con le descrizioni e qualche esempio di codice
- [ ] Capitolo linguaggio:
  - [x] linguaggio
  - [x] formattazione di base del testo: costrutti fondamentali

In programma:

- [ ] Inserimento di esempi di codice
- [ ] Sezione Kit, ovvero sorgenti pronti per vari tipi di documento

In futuro:

- [ ] Creazione di un logo della guida
- [ ] Collegamenti verso le guide GuIT di approfondimento
- [ ] Brevi note sull'uso di Lua
- [ ] Affinamento prosa

27.6. Licenza d'uso