

Scrivere con Typst

Guida all'uso del linguaggio Typst

Andrea Stefanelli

Scrivere con Typst

Manuale per scrivere testi utilizzando Typst in Italiano

Questo documento è stato scritto *interamente* nel linguaggio Typst, senza alcun ausilio di altri strumenti, utilizzando il programma per linux [Typesetter](#) che potete trovare nel formato flatpack su [Flathub](#).

Parte di questo manuale è stato estratto e parzialmente modificato dalla guida [Typst: from ZERO to HERO](#) di [Stefano Coelati Rama](#), che oltre ad essere la più completa esistente e in italiano, è decisamente più completa di questa, quindi merita decisamente la consultazione.

Tutti gli esempi presenti in questo manuale sono direttamente eseguibili nell'editor online ufficiale: [typst.app](#)

Il contenuto di questo manuale è rilasciato per uso didattico. In caso questo manuale diventi vecchio o non si trovino alcune informazioni, vi invito ad andare nella [community](#), o nei [docs](#) per cercare informazioni. Questo testo non è per fini di lucro, Typst è un progetto open-source! <https://github.com/typst/typst>

I miei contatti li trovate [qui](#)

Indice

1. Scrivere con Typst	8.
1.1. Perché dovrei utilizzare Typst	8.
1.2. Come funziona Typst: il flusso di lavoro	9.
1.3. Scrivere con Typst online	10.
1.4. Scrivere con Typst con il computer	10.
1.5. Scrivere con Typst con i dispositivi mobili	10.
2. Scrivere un documento con Typst	11.
2.1. Creare un documento con Typst	11.
3. Scrivere un libro con Typst	15.
4. Scrivere un manuale con Typst	20.
5. Sezionamento del documento	29.
5.1. Titolo di primo livello	29.
5.2. Titolo di secondo livello	29.
5.3. Titolo di terzo livello	29.
5.4. Titolo di quarto livello	29.
5.5. Titolo fuori dall'indice	30.
6. Attributi del testo	31.
6.1. Scrivere in corsivo (italico)	31.
6.2. Scrivere in grassetto	31.
6.3. Scrivere in corsivo grassetto	31.
6.4. Scrivere testo come su macchina da scrivere	32.
6.5. Scrivere testo tutto maiuscolo	32.
6.6. Scrivere testo in maiuscoletto con maiuscole	32.
6.7. Testo sottolineato	32.
6.8. Barrare il testo	33.
6.9. Scrivere testo tra virgolette	33.
6.10. Scrivere testo con linea sopra	33.
6.11. Scrivere pedici e apici	33.
6.12. Evidenziare il testo	34.
6.13. Caratteri speciali e come usarli	34.
7. Evidenziare e colorare il testo	35.
7.1. I codici per specificare i colori	35.
8. Inserire citazioni e blocchi di testo	38.
8.1. Inserire citazioni	38.
8.2. Inserire blocchi di testo	38.
8.3. Inserire blocchi di testo con sfondo colorato	39.

8.4. Inserire un riquadro colorato:	40.
8.5. Creare un template a schede riutilizzabile	40.
9. Scrivere elenchi e liste	42.
9.1. Scrivere una lista non numerata	42.
9.2. Scrivere una lista numerata	42.
9.3. Annidare una lista	42.
9.4. Scrivere una lista di definizioni	43.
9.5. Voci su più righe	43.
9.6. Spaziatura tra le voci	44.
10. Allineamento e dimensioni del testo	45.
11. Link web e link interni al documento	47.
11.1. Link con URL nel documento	47.
11.2. Link con etichetta nel collegamento	47.
11.3. Link per mandare una email	47.
11.4. Link interni al documento	48.
11.5. Link a un determinato numero di pagina	48.
12. Inserire immagini	49.
12.1. Inserire immagini allineate a sinistra	49.
12.2. Inserire immagini centrate	50.
12.3. Inserire immagini centrate con didascalia	50.
12.4. Inserire immagini allineate a destra	51.
12.5. Inserire immagini con la larghezza dei margini della pagina	52.
12.6. Inserire immagini affiancate	53.
13. Rappresentare il codice	55.
13.1. Rappresentare il codice per blocchi	55.
13.2. Rappresentare una porzione di riga come codice:	56.
14. Inserire una tabella	57.
14.1. Inserire una tabella semplice	57.
14.2. Inserire una tabella con colori alternati	57.
14.3. Inserire una tabella con intestazioni stilizzate	58.
14.4. Inserire una tabella con lunghezza colonne definita	59.
14.5. Bordi (stroke)	60.
15. Inserire spazi nel documento	61.
15.1. Spazio orizzontale	61.
15.2. Spazio verticale	61.
15.3. Spazio infinito	61.
15.4. Spazio non divisibile	61.
15.5. Inserire riga vuota nel documento	62.
15.6. Inserire una interruzione di pagina	62.
15.7. Inserire una linea orizzontale	62.
16. Matematica	63.

16.1. Modalità matematica	63.
16.1.1. Matematica inline	63.
16.1.2. Matematica a blocco (display)	63.
16.2. Operatori di base	64.
16.3. Potenze e pedici	64.
16.4. Frazioni	65.
16.5. Radici	65.
16.6. Simboli speciali comuni	66.
16.6.1. Lettere greche	66.
16.6.2. Altri simboli frequenti	66.
16.7. Sommatorie, produttorie, limiti	67.
16.8. Vettori e matrici	67.
16.9. Testo all'interno delle formule	68.
16.10. Formule numerate	68.
16.11. Spazi in matematica	68.
17. Inserire simboli	70.
18. Inserire caselle di testo nel documento	71.
19. Inserire una nota a piè di pagina	72.
20. Le regole set	73.
20.1. Cos'è una regola set?	73.
20.2. set text — il testo	73.
20.3. set par — i paragrafi	74.
20.4. set heading — i titoli	74.
20.5. set page — la pagina	74.
20.6. set list e set enum	75.
20.7. set table — le tabelle	75.
20.8. set figure — le figure	76.
20.9. set math.equation — le equazioni	76.
20.10. Lo scope delle regole set	76.
21. Le regole show	77.
21.1. La differenza tra set e show	77.
21.2. Trasformare il grassetto	77.
21.3. Trasformare i link	77.
21.4. Selettori: .where()	78.
21.5. Trasformare i titoli	79.
21.6. Trasformare il documento intero (show globale)	79.
22. Box e blocchi	81.
22.1. box vs block: la differenza	81.
22.2. Parametri di box e block	81.
22.2.1. Angoli arrotondati personalizzati	82.
22.2.2. Inset dettagliato	82.

22.3. Impedire le interruzioni di pagina	82.
22.4. Stack: impilare elementi	83.
22.5. Allineamento	83.
23. Impaginazione avanzata	85.
23.1. Intestazioni e piè di pagina	85.
23.1.1. Intestazione semplice	85.
23.1.2. Intestazione con numero di pagina	85.
23.1.3. Il contesto (context) e la consapevolezza del documento	85.
23.1.4. Intestazione con titolo sezione corrente	86.
23.2. Numerazione pagine	86.
23.2.1. Cambiare numerazione nel documento	87.
23.3. Colonne	87.
23.3.1. Saltare una colonna	87.
23.4. Pagine speciali	88.
23.4.1. Pagina con margini diversi	88.
23.4.2. Pagina in landscape (orizzontale)	89.
23.4.3. Copertina libro	89.
24. Layout a griglia con grid()	92.
24.1. Sintassi di base	92.
24.2. Definire le colonne	92.
24.3. Esempi pratici di layout	92.
24.3.1. Layout copertina (icona + testo)	92.
24.3.2. Layout a tre colonne informative	93.
24.4. La funzione layout()	94.
25. File multipli e moduli	95.
25.1. <code>#include</code> — includere contenuto	95.
25.1.1. Vantaggi di <code>#include</code>	95.
25.2. <code>#import</code> — importare funzioni e variabili	96.
25.2.1. Importare tutto con <code>*</code>	96.
25.2.2. Importare con alias	96.
25.2.3. Differenza tra import e include	96.
25.3. Organizzare un progetto grande	97.
26. Pacchetti	98.
26.1. Dove trovare i pacchetti	98.
26.2. Come importare un pacchetto	98.
26.3. Pacchetti principali	98.
26.3.1. showybox — Box stilizzate	98.
26.3.2. codly — Codice con numeri di riga	99.
26.3.3. cetz — Disegno vettoriale	99.
26.3.4. fletcher — Diagrammi e frecce	99.
26.3.5. physica — Notazione fisica	99.

26.3.6. polylux — Presentazioni	100.
26.3.7. tablex — Tabelle avanzate	100.
26.3.8. lovelace — Pseudocodice e algoritmi	100.
26.3.9. diagraph — Grafi con GraphViz	101.
26.4. Come scegliere la versione giusta	101.

1. Scrivere con Typst

Typst è un nuovo sistema di composizione tipografica open-source e linguaggio di markup corrispondente. Il compilatore Typst è un software gratuito ed è distribuito sotto la licenza [Apache License 2.0](#).

Il sistema è progettato per scrivere e formattare testi scientifici e formule matematiche. Typst supporta la formattazione semplice per applicazioni di formattazione comuni, funzioni personalizzabili, un linguaggio di scripting integrato e tipi di composizione matematica. È progettato per essere un'alternativa a LaTeX, potente come LaTeX pur essendo molto più facile da imparare e da usare.

A differenza di LaTeX, non ci sono requisiti minimi per la struttura di un documento. Non è necessario un preambolo e le proprietà dei documenti non devono essere impostate se non modificate dalle loro impostazioni predefinite. Inoltre, molte funzionalità che necessitano di pacchetti esterni in LaTeX (come il supporto bibliografico) sono implementate nativamente in Typst. Typst supporta pacchetti e modelli forniti dalla comunità, tuttavia rispetto a LaTeX, Typst ha relativamente meno pacchetti specializzati disponibili.

Il compilatore è sviluppato da Typst GmbH, che mantiene e supporta lo sviluppo del software, e gestisce un editor cloud-based collaborativo su typst.app.

Typst lo potete utilizzare per scrivere articoli, libri, presentazioni e tanto altro con uno stile tipografico elegante ed automatizzato.

Typst ha:

- Markup integrato per le attività di formattazione più comuni
- Funzioni flessibili per tutto il resto
- Un sistema di scripting strettamente integrato
- Tipografia matematica, gestione bibliografia e altro ancora
- Tempi di compilazione rapidi grazie alla compilazione incrementale
- Messaggi di errore amichevoli nel caso in cui qualcosa vada storto

1.1. Perché dovrei utilizzare Typst

Typst è molto adatto per la produzione di testi scientifici e per scrivere libri o documenti molto lunghi e con strutture complesse.

E' uno dei programmi utile agli studenti all'università per scrivere le tesi. Questo perché oltre a permettere di fare lavori di impaginazione all'altezza di lavori professionali consente di risparmiare molto tempo.

Immaginate di poter scrivere un documento come un libro o una lunga tesi divisa in capitoli, sezioni e sottosezioni. Poi si dice al programma di creare l'indice e lui lo crea. Da solo crea l'indice indicando i numeri di pagina e se in seguito noi andiamo a modificare il nostro documento lui aggiorna l'indice automaticamente. Potete immaginare che bel risparmio di tempo non dover creare l'indice manualmente e non doverlo aggiornare manualmente per ogni modifica che noi facciamo al documento.

La maggior parte delle persone sono abituate a usare solo programmi come Word o Writer per scrivere testi. Ma scrivere un libro o gestire un documento molto lungo con tali programmi non è per niente comodo. Per imparare a usare Typst dovrete inizialmente fare un piccolo sforzo per entrare in un'altra modalità di gestire il testo. Ma se lo farete vi accorgerete che ne vale la pena e non vorrete mai più tornare a scrivere con un programma tradizionale come Word o Writer.

Questa guida non vuol essere una guida completa su Typst, ma un manuale pratico per poter scrivere subito usando tutte le principali funzioni di questo splendido linguaggio di markup.

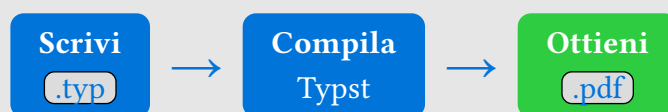
Per scelta ho deciso di non inserire in questa guida approfondimenti completi su come scrivere la matematica, ma ho deciso di concentrarmi nella spiegazione su come scrivere testi più o meno lunghi e come scrivere libri di qualità.

Questa guida è volutamente sintetica per essere utilizzata anche dai neofili che non hanno voglia di confrontarsi con una voluminosa documentazione in inglese.

Per ulteriori informazioni dettagliate al seguente link trovate la guida completa in inglese di Typst:

- [Typst Documentation](#)

1.2. Come funziona Typst: il flusso di lavoro



Il processo è sempre lo stesso. Nell'editor online (typst.app), oppure nell'editor che avete scelto di utilizzare (nel mio caso Typesetter) la compilazione avviene automaticamente mentre scrivi: ogni volta che aggiungi testo o codice typst o fai una modifica nel documento, l'anteprima del PDF si aggiorna in tempo reale nel pannello di destra. È sempre istantaneo (è istantaneo anche se avete un file enorme). Quando hai finito il tuo documento compilalo con Typst esportandolo in formato PDF.

Typst lo potete utilizzare per scrivere articoli, libri, presentazioni e tanto altro con uno stile tipografico elegante ed automatizzato. Scrivete il vostro documento con il codice Typst e poi compilate in formato PDF il vostro documento.

1.3. Scrivere con Typst online

Potete scrivere con Typst online con l'editor cloud-based collaborativo gestito da Typst GmbH, che consente agli utenti di visualizzare in anteprima il loro lavoro mentre scrivono. Il servizio di editor online lo troverete su typst.app. Vengono offerti sia servizi gratuiti che a pagamento. Vengono offerti gratuitamente anche una lista di templates pronti su [Typst Universe](https://typst.universe), che potete scaricare ed utilizzare.

1.4. Scrivere con Typst con il computer

Se usate linux potete scrivere utilizzando Typst offline sul vostro PC utilizzando il programma [Typesetter](https://github.com/typst/typst) che troverete in formato flatpack su [Flathub](https://flathub.org/apps/details/com.typst.Typesetter).

Se utilizzate Windows potete utilizzare il software [Typst Desktop](https://typst.com/desktop).

Se utilizzate un Mac potete invece utilizzare [TypstEdit](https://typst.com/edit).

1.5. Scrivere con Typst con i dispositivi mobili

Potete utilizzare delle app che utilizzano Typst anche su tablet con [Typst Editor](https://typst.com/editor) per Ipad e [Typtic Typst Editor](https://typst.com/editor) per i tablet e gli smartphone android. Con entrambe le app la compilazione avviene online.

2. Scrivere un documento con Typst

Quando noi creiamo un progetto con Typst per prima cosa dobbiamo creare una cartella in cui inserire il nostro progetto. Il file.typ del nostro progetto lo salveremo in questa cartella. All'interno di questa cartella dobbiamo creare una cartella con il nome «immagini» in cui posizioneremo le immagini del nostro documento. Quando apriamo il file.typ del progetto, se il progetto ha anche delle immagini, se utilizziamo **Typesetter** ci verrà chiesto ogni volta di selezionare la cartella del progetto.

Se il nostro progetto è online dobbiamo caricare comunque le immagini nella cartella del nostro progetto.

Ora apriamo un nuovo file .typst vuoto e andiamo a vedere come impostarlo per poter iniziare a scrivere con Typst.

2.1. Creare un documento con Typst

Quando abbiamo aperto un nuovo documento di typst, cioè un file con l'estensione .typ dobbiamo inserire il codice che definisce le istruzioni tipografiche del nostro documento.

Qui vi mostro come setto i miei documenti con Typst, potete decidere di conservare queste impostazioni tipografiche così come sono o quando avete preso confidenza con il programma modificarle secondo i vostri gusti personali.

Quindi ora scriviamo nel documento il seguente codice per impostare tipograficamente il documento.

Un file Typst ha estensione .typ ed è un semplice file di testo. Non c'è una struttura obbligatoria: puoi iniziare a scrivere direttamente.

Per ogni pezzo di codice inseriremo un titolo con il commento di cosa fa quel codice.

Le righe che iniziano con `//` sono commenti: Typst le ignora completamente. Sono utili per lasciare note a te stesso.

L'inserimento dei commenti nel codice è importante perché permette di trovare rapidamente il codice di tutte le impostazioni del documento per modificarlo se successivamente desideriamo.

Quindi vediamo ora il codice che dobbiamo inserire per impostare tipograficamente il documento. Inseriamo un gruppo di impostazioni tipografiche alla volta:

Inseriamo le impostazioni tipografiche della pagina:

```
// _____  
// IMPOSTAZIONI PAGINA  
// _____  
  
#set page(  
  paper: "a4",  
  margin: (x: 2.5cm, y: 2.5cm),  
  columns: 1,  
)
```

Inseriamo le impostazioni tipografiche del testo:

```
// _____  
// IMPOSTAZIONI TESTO  
// _____  
  
#set text(font: "Libertinus Serif", lang: "it")  
#set par(justify: true)  
#set text(size: 12pt)
```

Inseriamo le impostazioni tipografiche per lo stile dei titoli:

```
// _____  
// STILE TITOLI  
// _____  
  
// Livello 1 — Capitolo  
#show heading.where(level: 1): set align(center)  
#show heading.where(level: 1): set text(size: 25pt)  
#show heading.where(level: 1): it => [  
  #[] <new-chapter>  
  #pagebreak(weak: true)  
  #it.body  
  #set align(center)  
  #v(0.8em)  
]  
  
// Livello 2 — Sezione  
#show heading.where(level: 2): set text(size: 20pt)  
#show heading.where(level: 2): it => [  
  #it.body  
  #v(0.5em)  
]
```

```
// Livello 3 — Sottosezione
#show heading.where(level: 3): set text(size: 16pt)
#show heading.where(level: 3): it => [
  #it.body
  #v(0.3em)
]
```

Inseriamo le impostazioni tipografiche per lo stile dei link:

```
// _____
// LINK
// _____
#show link: set text(blue)
```

Ora che abbiamo inserito le impostazioni tipografiche possiamo iniziare a scrivere il nostro documento.

Se invece vogliamo scrivere un libro o comunque una dispensa con frontespizio e indice generale bisogna inserire ulteriore codice.

Per scrivere un semplice frontespizio con il titolo del documento e il sottotitolo del documento inserire il seguente codice:

```
// =====
// FRONTESPIZIO
// =====
#show title: set text(size: 40pt)
#show title: set align(center)
#title[#v(9cm)
Titolo libro
]
#grid(
  columns: (1fr),
  align(center)[
    #text(size: 25pt)[Sottotitolo]
  ],
)
```

Ovviamente sostituiremo la scritta «Titolo libro» con il titolo del nostro documento e la scritta «Sottotitolo» con il sottotitolo del nostro documento.

Per inserire l'indice generale nel documento e la numerazione delle pagine inserire il seguente codice:

```
// _____  
// NUMERAZIONE PAGINE E INDICE  
// _____  
#set page(numbering: "1")  
#outline(  
  title: [Indice #v(0.7em)],  
  depth: 3,  
  indent: auto,  
)
```

Ora possiamo iniziare a scrivere il nostro libro.

E' buona abitudine mettere un commento nel documento che identifica l'inizio del documento, prima di scrivere il nostro primo capitolo. Quindi inserire il seguente codice:

```
// =====  
// INIZIO DOCUMENTO  
// =====
```

3. Scrivere un libro con Typst

Se vogliamo scrivere un libro più elaborato con intestazioni, copertina e indice generale. Quindi dobbiamo impostare il documento diversamente. Vediamo come farlo.

Quindi ora scriviamo nel documento il seguente codice per impostare tipograficamente il nostro libro. Inseriamo un gruppo di impostazioni tipografiche alla volta:

Inseriamo le impostazioni tipografiche della pagina:

```
// _____  
// IMPOSTAZIONI PAGINA  
// _____  
  
#set page(  
  paper: "a4",  
  margin: (top: 3.6cm, bottom: 2.6cm, left: 2.6cm, right: 2.6cm),  
  numbering: "1",  
  number-align: center,  
  header: context {  
    let pn = counter(page).get().first()  
    if pn > 2 {  
      set text(size: 10pt, fill: luma(20))  
      grid(  
        columns: (1fr, auto),  
        align(left)[_ Titolo intestazione pagina_],  
        align(right)[Pag. #pn],  
      )  
      v(-0.4em)  
      line(length: 100%, stroke: 0.4pt + luma(20))  
    }  
  },  
  footer: context {  
    let pn = counter(page).get().first()  
    if pn > 2 {  
      line(length: 100%, stroke: 0.4pt + luma(20))  
      v(-0.3em)  
    }  
  },  
)
```

Inseriamo le impostazioni tipografiche del testo:

```
// _____  
// IMPOSTAZIONI TESTO  
// _____  
  
#set text(  
  font: ("Libertinus Serif"),  
  size: 12pt,  
  lang: "it",  
  region: "IT",  
  hyphenate: true,  
)  
  
#set par(  
  justify: true,  
  leading: 0.75em,  
  spacing: 1.15em,  
)
```

Inseriamo le impostazioni tipografiche per lo stile dei titoli:

```
// _____  
// STILE TITOLI  
// _____  
  
// Livello 1 — Capitolo  
#show heading.where(level: 1): set align(center)  
#show heading.where(level: 1): set text(size: 25pt)  
#show heading.where(level: 1): it => [  
  #[] <new-chapter>  
  #pagebreak(weak: true)  
  #it.body  
  #set align(center)  
  #v(0.8em)  
]  
  
// Livello 2 — Sezione  
#show heading.where(level: 2): set text(size: 20pt)  
#show heading.where(level: 2): it => [  
  #it.body  
  #v(0.5em)  
]  
  
// Livello 3 — Sottosezione
```

```
#show heading.where(level: 3): set text(size: 16pt)
#show heading.where(level: 3): it => [
  #it.body
  #v(0.3em)
]
```

Inseriamo le impostazioni tipografiche per lo stile dei link:

```
// -----
// LINK
// -----
#show link: it => {
  set text(fill: blue)
  underline(it)
}
```

Per scrivere un frontespizio con il titolo del documento, il sottotitolo del documento e l'autore inserire il seguente codice:

```
// =====
// FRONTESPIZIO
// =====
#page(
  margin: 0pt,
  header: none,
  footer: none,
  numbering: none,
)[
  // Sfondo a gradiente
  #place(
    top + left,
    block(
      width: 100%,
      height: 100%,
      fill: gradient.linear(
        rgb("#0d2137"),
        rgb("#1a3a5c"),
        angle: 150deg,
      ),
    ),
  ),
)
```

```
// Logo / titolo
#place(
  center + horizon,
  block(width: 88%)[
    #align(center)[
      #v(1.2fr)

      // Titolo principale
      #text(
        fill: white,
        size: 56pt,
        weight: "bold",
        font: ("Libertinus Serif"),
      )[Titolo libro]

      #v(0.4em)

      // Sottotitolo
      #text(fill: rgb("#aed6f1"), size: 22pt,
        font: ("Libertinus Serif"),)[
        Sottotitolo del libro
      ]
      #v(3em)

      // Autore del libro
      #text(fill: rgb("#d8e2f2"), size: 13pt,
        font: ("Libertinus Serif"),)[
      #text(fill: white)[Autore del libro]
      ]

      #v(1.8em)

      // Descrizione
      #text(fill: rgb("#d6eaf8"), size: 12.5pt, style: "italic")[]

      #v(2.5em)

      #h(0.5em)

      #h(0.5em)
```

```

#h(0.5em)

#h(0.5em)

#h(0.5em)

#h(0.5em)

#v(1fr)
]
],
)
]

```

Ovviamente sostituiremo la scritta «Titolo libro» con il titolo del nostro documento, la scritta «Sottotitolo» con il sottotitolo del nostro documento e metteremo il nome dell'autore del libro.

Per inserire l'indice generale nel documento e la numerazione delle pagine inserire il seguente codice:

```

// -----
// NUMERAZIONE PAGINE E INDICE
// -----
#set page(numbering: "1")
#outline(
  title: [Indice #v(0.7em)],
  depth: 3,
  indent: auto,
)

```

Ora possiamo iniziare a scrivere il nostro libro.

E' buona abitudine mettere un commento nel documento che identifica l'inizio del documento, prima di scrivere il nostro primo capitolo. Quindi inserire il seguente codice:

```

// =====
// INIZIO DOCUMENTO
// =====

```

4. Scrivere un manuale con Typst

Se vogliamo scrivere un manuale più tecnico, sarà più elaborato con intestazioni numerate, copertina e indice generale. Quindi dobbiamo impostare il documento diversamente. Vediamo come farlo.

Quindi ora scriviamo nel documento il seguente codice per impostare tipograficamente il nostro libro. Inseriamo un gruppo di impostazioni tipografiche alla volta:

Inseriamo le impostazioni tipografiche della pagina:

```
// _____  
// IMPOSTAZIONI PAGINA  
// _____  
  
#set page(  
  paper: "a4",  
  margin: (top: 2.6cm, bottom: 2.6cm, left: 2.6cm, right: 2.6cm),  
  numbering: "1",  
  number-align: center,  
  header: context {  
    let pn = counter(page).get().first()  
    if pn > 5 {  
      set text(size: 10pt, fill: luma(20))  
      grid(  
        columns: (1fr, auto),  
        align(left)[Titolo manuale],  
        align(right)[Pag. #pn],  
      )  
      v(-0.4em)  
      line(length: 100%, stroke: 0.4pt + luma(20))  
    }  
  },  
  footer: context {  
    let pn = counter(page).get().first()  
    if pn > 5 {  
      line(length: 100%, stroke: 0.4pt + luma(20))  
      v(-0.3em)  
    }  
  },  
)
```

Inseriamo le impostazioni tipografiche del testo:

```
// _____  
// IMPOSTAZIONI TESTO  
// _____  
  
#set text(  
  font: ("Libertinus Serif"),  
  size: 12pt,  
  lang: "it",  
  region: "IT",  
  hyphenate: true,  
)  
  
#set par(  
  justify: true,  
  leading: 0.75em,  
  spacing: 1.15em,  
)
```

Inseriamo le impostazioni tipografiche per lo stile dei titoli:

```
// _____  
// STILE TITOLI  
// _____  
  
// 1. Questa regola si occupa SOLO di mandare a capo i titoli di livello 1 in modo  
// pulito  
  
#show heading.where(level: 1): it => [  
  #[] <new-chapter>  
  #pagebreak(weak: true)  
  #it.body  
]  
  
#show heading.where(level: 1): it => {  
  pagebreak(weak: true)  
  it  
}  
  
// 2. Questa regola si occupa SOLO della grafica verde scuro del capitolo  
#show heading.where(level: 1): it => {  
  v(0.5em)
```

```
block(
  fill: color.green.darken(65%),
  width: 100%,
  inset: (x: 1.3em, y: 1em),
  radius: 7pt,
)[
  #set text(fill: white, size: 20pt, weight: "bold", font: ("Libertinus Serif"))
  #set par(justify: false)
  #context {
    let n = counter(heading).get()
    if it.numbering != none {
      numbering(it.numbering, ..n)
      h(0.5em)
    }
  }
  #it.body
]
v(1em)
}
```

```
// Livello 2 — Sezione
#show heading.where(level: 2): it => {
  v(1.2em)
  block[
    #set text(size: 18pt, weight: "bold", fill: color.green.darken(65%))
    #set par(justify: false)
    #context {
      let n = counter(heading).get()
      if it.numbering != none {
        numbering(it.numbering, ..n)
        h(0.4em)
      }
    }
    #it.body
    #v(-0.15em)
    #line(length: 100%, stroke: 1.8pt + color.green.darken(65%))
  ]
  v(0.5em)
}
```

```
// Livello 3 — Sottosezione
```

```
#show heading.where(level: 3): it => {
  v(0.9em)
  block[
    #set text(size: 16pt, weight: "bold", fill: color.green.darken(65%))
    #set par(justify: false)
    #context {
      let n = counter(heading).get()
      if it.numbering != none {
        numbering(it.numbering, ..n)
        h(0.4em)
      }
    }
    #it.body
  ]
  v(0.6em)
}

// Livello 4 – Paragrafo titolato
#show heading.where(level: 4): it => {
  v(0.6em)
  block[
    #set text(size: 11pt, style: "italic", fill: C.grigio)
    #set par(justify: false)
    #it.body
  ]
  v(0.1em)
}
```

Inseriamo le impostazioni tipografiche per lo stile dei link:

```
// _____
// LINK
// _____
#show link: it => {
  set text(fill: blue)
  underline(it)
}
```

Inseriamo le impostazioni tipografiche per lo stile dei blocchi di codice:

```
// _____
// STILE BLOCCHI DI CODICE
```

```
// -----
#show raw.where(block: true): it => {
  block(
    fill: luma(220),
    width: 100%,
    inset: (x: 1.3em, y: 1em),
    radius: 6pt,
    stroke: 0.6pt + black,
  )[
    #set text( font: ("Libertinus Serif"), size: 12pt)
    #set par(justify: false, leading: 0.7em)
    #it
  ]
}

#show raw.where(block: false): it => {
  box(
    fill: luma(220),
    inset: (x: 4pt),
    outset: (y: 2pt),
    radius: 4pt,
    stroke: 0.5pt + black,
  )[
    #set text( font: ("Libertinus Serif"), size: 12pt)
    #it
  ]
}
```

Inseriamo le impostazioni tipografiche per lo stile dei blocchi di testo aggiunto:

```
// -----
// STILE BLOCCHI DI TESTO AGGIUNTI
// -----

#let line-block = rect.with(fill: luma(240), inset: 1em, stroke: (left: 0.25em))

#let blocco-giallo(corpo) = {
  set text(size: 12pt)
  block(
    fill: rgb("#fef9e7"),
    inset: 0.6em,
    radius: 6pt,
```

```

stroke: 0.5pt + luma(210),
)[#corpo]
}

#let schede(titolo: "Titolo", colore: blue, corpo) = block(
  width: 100%,
  radius: 8pt,
  stroke: 1pt + luma(150),
  clip: true,
  below: 1em,
)[
  #block(fill: colore, width: 100%, inset: (x: 1em, y: 0.6em))[
    #text(fill: white, weight: "bold")[#titolo]
  ]
  #block(fill: white, width: 100%, inset: (x: 1em, y: 0.9em))[#corpo]
]

#let riquadro-colorato(colore, corpo) = block(
  fill: colore.lighten(70%),
  stroke: (left: 4pt + colore),
  inset: (x: 1em, y: 0.8em),
  radius: 4pt,
  width: 100%,
)[#corpo]

```

Per scrivere un frontespizio con il titolo del documento, il sottotitolo del documento e l'autore inserire il seguente codice:

```

// =====
// FRONTESPIZIO
// =====

#page(
  margin: 0pt,
  header: none,
  footer: none,
  numbering: none,
)[
  // Sfondo a gradiente
  #place(
    top + left,
    block(
      width: 100%,

```

```
height: 100%,
fill: color.green.darken(75%),
),
),
)

// Logo / titolo
#place(
  center + horizon,
  block(width: 88%)[
    #align(center)[
      #v(1fr)

      #v(1.2em)

      // Titolo principale
      #text(
        fill: white,
        size: 58pt,
        weight: "bold",
        font: ("Libertinus Serif"),
      )[Titolo manuale]

      #v(0.4em)

      // Sottotitolo
      #text(fill: rgb("#aed6f1"), size: 24pt,
        font: ("Libertinus Serif"),)[
        Sottotitolo
      ]

      #v(1.8em)

      // Autore
      #v(1.5em)
      #text(fill: rgb("#d8e2f2"), size: 14pt,
        font: ("Libertinus Serif"),)[
      #text(fill: white)[Autore]
    ]
  ]
)
```

```
#v(1.8em)

// Descrizione
#text(fill: rgb("#d6eaf8"), size: 12.5pt, style: "italic")[]

#v(2.5em)

// Badge argomenti
#let badge(t) = box(
  fill: white.transparentize(82%),
  stroke: 0.7pt + white.transparentize(60%),
  inset: (x: 0.9em, y: 0.4em),
  radius: 4pt,
)[#text(fill: rgb("#d6eaf8"), size: 8.5pt)[#t]]

#h(0.5em)

#h(0.5em)

#h(0.5em)

#h(0.5em)

#v(1fr)
]
],
)

// Footer frontespizio
#place(
  bottom + center,
  dy: -1.8cm,
  text(fill: rgb("#aed6f1"), size: 11pt)[
    Scritto ed impaginato interamente utilizzando Typst con il software Typesetter],
)
]
```

Ovviamente sostituiremo la scritta «Titolo manuale» con il titolo del nostro documento, la scritta «Sottotitolo» con il sottotitolo del nostro documento e metteremo il nome dell'autore del libro.

Per inserire l'indice generale nel documento e la numerazione delle pagine inserire il seguente codice:

```
// _____  
// NUMERAZIONE PAGINE E INDICE  
// _____  
#set page(numbering: "1.1.1.")  
#outline(  
  title: [Indice],  
  depth: 3,  
  indent: 2em,  
)
```

Per inserire la numerazione dei titoli inserire il seguente codice:

```
// _____  
// NUMERAZIONE TITOLI  
// _____  
#set heading(numbering: "1.1.1.")
```

Ora possiamo iniziare a scrivere il nostro manuale.

E' buona abitudine mettere un commento nel documento che identifica l'inizio del documento, prima di scrivere il nostro primo capitolo. Quindi inserire il seguente codice:

```
// =====  
// INIZIO DOCUMENTO  
// =====
```

5. Sezionamento del documento

5.1. Titolo di primo livello

Scrivere una volta il carattere (=) seguito dal titolo di primo livello preceduto da uno spazio:

= Titolo di 1° livello

5.2. Titolo di secondo livello

Scrivere due volte il carattere (=) seguito dal titolo di secondo livello preceduto da uno spazio:

= = Titolo di 2° livello

5.3. Titolo di terzo livello

Scrivere tre volte il carattere (=) seguito dal titolo di terzo livello preceduto da uno spazio:

= = = Titolo di 3° livello

5.4. Titolo di quarto livello

Scrivere quattro volte il carattere (=) seguito dal titolo di quarto livello preceduto da uno spazio:

= = = = Titolo di 4° livello

5.5. Titolo fuori dall'indice

Se non vogliamo che un titolo non compaia nell'indice (per esempio nella prefazione o nell'appendice) dobbiamo scrivere un apposito codice.

Immaginiamo che vogliamo scrivere un titolo di primo livello che non compaia nell'indice con il titolo «Prefazione» dovremo scrivere il seguente codice:

```
#heading(level: 1, numbering: none, outlined: false)[Prefazione]
```

6. Attributi del testo

6.1. Scrivere in corsivo (italico)

Scrivere una parola o frase tra una linea di sottolineatura è in corsivo (italico). Il codice:

```
_corsivo_  
_frase in corsivo_
```

Produce:

corsivo
frase in corsivo

6.2. Scrivere in grassetto

Scrivere una parola o frase tra asterischi è in grassetto. Il codice:

```
*grassetto*  
*frase in grassetto*
```

Produce:

grassetto
frase in grassetto

6.3. Scrivere in corsivo grassetto

Scrivere una parola o frase tra un asterisco e una linea di sottolineatura è in corsivograssetto. Il codice:

```
*corsivo grassetto*  
*frase in corsivo grassetto*
```

Produce:

corsivo grassetto

frase in corsivo grassetto

6.4. Scrivere testo come su macchina da scrivere

Per scrivere testo come su macchina da scrivere (testo monospazio) inserire il seguente codice:

```
`Testo monospazio come su macchina da scrivere`
```

Produce:

```
Testo monospazio come su macchina da scrivere
```

6.5. Scrivere testo tutto maiuscolo

Per scrivere testo tutto maiuscolo inserire il seguente codice:

```
#upper("testo in maiuscolo")
```

Produce:

TESTO IN MAIUSCOLO

6.6. Scrivere testo in maiuscoletto con maiuscole

Per scrivere testo in maiuscoletto con maiuscole inserire il seguente codice:

```
#smallcaps("Testo in maiuscoletto con Maiuscole")
```

Produce:

TESTO IN MAIUSCOLETTO CON MAIUSCOLE

6.7. Testo sottolineato

Per scrivere testo sottolineato inserire il seguente codice:

```
#underline("testo sottolineato")
```

Produce:

testo sottolineato

6.8. Barrare il testo

Per scrivere testo barrato inserire il seguente codice:

```
#strike("testo barrato")
```

Produce:

~~testo barrato~~

6.9. Scrivere testo tra virgolette

Per scrivere testo tra virgolette inserire il seguente codice:

```
#smartquote()testo tra virgolette#smartquote()
```

Produce:

«testo tra virgolette»

6.10. Scrivere testo con linea sopra

Per scrivere testo con linea sopra inserire il seguente codice:

```
#overline("testo con linea sopra")
```

Produce:

testo con linea sopra

6.11. Scrivere pedici e apici

Per scrivere un pedice in piccolo sotto il testo X scrivere il seguente codice:

```
X#sub("pedice")
```

E produrremo:

X_{pedice}

Per scrivere una apice in piccolo sopra il testo X scrivere il seguente codice:

```
X#super("apice")
```

E produrremo:

X^{apice}

6.12. Evidenziare il testo

Potete evidenziare il testo scrivendo il seguente codice:

```
#highlight[testo evidenziato]
```

E otterremo:

testo evidenziato

6.13. Caratteri speciali e come usarli

Alcuni caratteri hanno un significato speciale in Typst. Se vuoi inserirli come testo normale, devi farli precedere dal tasto $\backslash$:

Carattere	Come scriverlo	Significato speciale
$\textcircled{*}$	$\backslash*$	Grassetto
$\textcircled{\text{c}}$	$\backslash\text{c}$	Corsivo
$\textcircled{\sim}$	$\backslash\sim$	Spazio insecabile
$\textcircled{\text{`}}$	$\backslash\text{...`}$	Codice inline
$\textcircled{\$}$	$\backslash\$$	Matematica
$\textcircled{\#}$	$\backslash\#$	Istruzione codice
$\textcircled{@}$	$\backslash@$	Riferimento
$\textcircled{<}$	$\backslash<$	Etichetta
$\textcircled{\backslash}$	$\backslash\backslash$	Backslash letterale

7. Evidenziare e colorare il testo

Ci sono due metodi per colorare il testo del documento.

Con il primo metodo per colorare il testo scrivere il seguente codice:

```
#text(fill:red)[testo in rosso]
```

E otterremo:

testo in rosso

Con il secondo metodo per colorare il testo scrivere il seguente codice:

```
#text(red)[Testo in rosso]
```

E otterremo:

Testo in rosso

7.1. I codici per specificare i colori

Typst supporta diversi modi per specificare i colori. Vediamo quali sono.

RGB

Il più comune: tre valori (Rosso, Verde, Blu) da 0 a 255:

```
// RGB con valori 0-255
#rgb(255, 0, 0) // Rosso puro
#rgb(0, 0, 255) // Blu puro
#rgb(128, 128, 128) // Grigio medio
// Con stringa esadecimale (come in CSS):
#rgb("#ff0000") // Rosso
#rgb("#1a3a5c") // Blu scuro
#rgb("#27ae60") // Verde
```

Quindi se scriviamo il seguente codice:

```
#text(rgb(0, 0, 255))[Testo in blu puro]
```

Otterremo:

Testo in blu puro

Scala di grigi

```
#luma(0) // Nero
#luma(128) // Grigio 50%
#luma(255) // Bianco
```

Quindi se scriviamo il seguente codice:

```
#text(luma(128))[Testo in Grigio 50%]
```

Otterremo:

Testo in Grigio 50%

OKLCH (moderno, perceptually uniform)

```
// oklch(luminosità, croma, tonalità)
#oklch(60%, 0.2, 240deg) // Blu medio
#oklch(80%, 0.15, 30deg) // Arancio chiaro
```

Quindi se scriviamo il seguente codice:

```
#text(oklch(80%, 0.15, 30deg))[Testo in arancio chiaro]
```

Otterremo:

Testo in arancio chiaro

Colori predefiniti

black	white	gray	silver
red	maroon	orange	yellow
olive	green	teal	navy
blue	aqua	purple	fuchsia
lime	lime	...	

Usali con il prefisso `color.`:

```
color.red color.blue color.green
// Oppure direttamente (senza prefisso nei contesti colore):
red blue green
```

Quindi se scrivo il codice:

```
#text(red)[Testo in rosso]
```

Oppure il codice:

```
#text(color.red)[Testo in rosso]
```

In entrambi i casi otterremo:

Testo in rosso

I colori si possono modificare:

```
// Modificare un colore  
#color.red.darken(30%) // Rosso più scuro  
#color.blue.lighten(50%) // Blu più chiaro  
#color.green.transparentize(70%) // Verde trasparente
```

Quindi se scrivo il codice:

```
#text(color.red.darken(30%))[Testo in rosso più scuro]
```

Otterremo:

Testo in rosso più scuro

8. Inserire citazioni e blocchi di testo

8.1. Inserire citazioni

Per rappresentare una citazione con il relativo autore che risulti separato dal resto scrivere il seguente codice:

```
#quote(block: true, attribution: "Autore citazione")[  
  Testo della citazione  
]
```

Quindi se scriviamo il seguente codice:

```
#quote(block: true, attribution: "Mahatma Gandhi")[  
  Vivi come se dovessi morire domani, impara come se dovessi vivere per sempre.  
]
```

Produrremo:

Vivi come se dovessi morire domani, impara come se dovessi vivere per sempre.
— Mahatma Gandhi

8.2. Inserire blocchi di testo

Ora creiamo un blocco di testo personalizzato chiamato `line-block` inserendo nelle impostazioni tipografiche del documento il seguente codice:

```
#let line-block = rect.with(fill: luma(240), inset: 1em, stroke: (left: 0.25em))
```

Ora per rappresentare il blocco di testo `line-block` nel documento scrivere per esempio il seguente codice:

```
#line-block[  
  Dalla favola di Pinocchio è stato fatto un libro diffuso in tutto il mondo. Poi a Collodi
```

è stato creato il parco di Pinocchio che attira turisti da tutta l'Italia.
]

Otterremo:

Dalla favola di Pinocchio è stato fatto un libro diffuso in tutto il mondo. Poi a Collodi è stato creato il parco di Pinocchio che attira turisti da tutta l'Italia.

8.3. Inserire blocchi di testo con sfondo colorato

Ora creiamo un blocco di testo personalizzato chiamato `blocco-giallo` inserendo nelle impostazioni tipografiche del documento il seguente codice:

```
#let blocco-giallo(corpo) = {  
  set text(size: 12pt)  
  block(  
    fill: rgb("#fef9e7"),  
    inset: 0.6em,  
    radius: 6pt,  
    stroke: 0.5pt + luma(210),  
  )[#corpo]  
}
```

Ora per rappresentare il blocco di testo `blocco-giallo` nel documento scrivere per esempio il seguente codice:

```
#blocco-giallo[  
  Questo blocco di testo ha uno stile  
  personalizzato interno.  
  Abbiamo scelto il colore dello sfondo, per il resto è un normale paragrafo.  
]
```

Otterremo:

Questo blocco di testo ha uno stile personalizzato interno. Abbiamo scelto il colore dello sfondo, per il resto è un normale paragrafo.

8.4. Inserire un riquadro colorato:

Vediamo come inserire un riquadro colorato.

Prima dobbiamo scrivere nelle impostazioni tipografiche del documento il seguente codice:

```
#let riquadro-colorato(colore, corpo) = block(  
  fill: colore.lighten(70%),  
  stroke: (left: 4pt + colore),  
  inset: (x: 1em, y: 0.8em),  
  radius: 4pt,  
  width: 100%,  
)[#corpo]
```

Poi nel documento dove vogliamo inserire un riquadro colorato scriveremo il seguente codice:

```
#riquadro-colorato(blue)[  
  Questo è un riquadro blu con _corsivo_ e *grassetto*.  
]  
  
#riquadro-colorato(red)[  
  Questo è un riquadro rosso.  
]
```

Otterremo:

Questo è un riquadro blu con *corsivo* e **grassetto**.

Questo è un riquadro rosso.

8.5. Creare un template a schede riutilizzabile

Vediamo come creare un template riutilizzabile a schede.

Prima dobbiamo scrivere nelle impostazioni tipografiche del documento il seguente codice:

```
#let schede(titolo: "Titolo", colore: blue, corpo) = block(  
  width: 100%,  
  radius: 8pt,
```

```
stroke: 1pt + luma(150),  
clip: true,  
below: 1em,  
)[  
#block(fill: colore, width: 100%, inset: (x: 1em, y: 0.6em))[  
#text(fill: white, weight: "bold")[#titolo]  
]  
#block(fill: white, width: 100%, inset: (x: 1em, y: 0.9em))[#corpo]  
]
```

Poi nel documento dove vogliamo inserire le schede scriveremo il seguente codice:

```
#schede(titolo: " Nota Importante")[  
Ricordati di salvare il file prima di chiudere.  
]  
#schede(titolo: " Completato", colore: rgb("#1e8449"))[  
Il documento è stato consegnato con successo.  
]
```

Otterremo:

Nota Importante

Ricordati di salvare il file prima di chiudere.

Completato

Il documento è stato consegnato con successo.

9. Scrivere elenchi e liste

9.1. Scrivere una lista non numerata

Scrivere una linea a inizio riga per ogni elemento della lista. Per esempio il codice:

```
- Primo punto della lista  
- Secondo punto della lista  
- Terzo punto della lista
```

Produce:

- Primo punto della lista
- Secondo punto della lista
- Terzo punto della lista

9.2. Scrivere una lista numerata

Scrivere il segno `+` a inizio riga per ogni elemento della lista. Per esempio il codice:

```
+ Primo punto della lista  
+ Secondo punto della lista  
+ Terzo punto della lista
```

Produce:

1. Primo punto della lista
2. Secondo punto della lista
3. Terzo punto della lista

9.3. Annidare una lista

Sia le liste non numerate che numerate possono essere annidate. Cioè si possono creare liste all'interno di liste.

Facciamo un esempio per capire come funziona. Il seguente codice:

```
+ Primo punto della lista
#set enum(numbering: "a")
+ Primo
+ Secondo
+ Secondo punto della lista
+ Terzo punto della lista
+ Quarto punto della lista
```

Produce:

1. Primo punto della lista
 - a) Primo
 - b) Secondo
2. Secondo punto della lista
3. Terzo punto della lista
4. Quarto punto della lista

9.4. Scrivere una lista di definizioni

Per creare un glossario o elenco di termini con definizioni, usa il codice `/ Termine: definizione` per ogni elemento della lista. Per esempio il codice:

```
/ Markup: Un linguaggio che usa marcatori speciali nel testo per descriverne la
struttura e la formattazione.
/ Compilatore~Typst: Programma che trasforma il codice sorgente in un file di
output (nel caso di Typst, in un PDF).
/ PDF: Portable Document Format. Formato di file universalmente leggibile.
```

Produce:

Markup Un linguaggio che usa marcatori speciali nel testo per descriverne la struttura e la formattazione.

Compilatore Typst Programma che trasforma il codice sorgente in un file di output (nel caso di Typst, in un PDF).

PDF Portable Document Format. Formato di file universalmente leggibile.

9.5. Voci su più righe

Le voci di una lista possono contenere più paragrafi. Basta far rientrare il testo successivo allo stesso livello.

Se scriviamo il codice:

```
- *Prima voce:*  
Questo è il paragrafo di spiegazione per la  
prima voce. Può essere lungo quanto vuoi.  
Puoi anche avere più paragrafi per la stessa voce.  
- *Seconda voce:*  
Un'altra spiegazione.
```

Otteniamo:

- **Prima voce:** Questo è il paragrafo di spiegazione per la prima voce. Può essere lungo quanto vuoi. Puoi anche avere più paragrafi per la stessa voce.
- **Seconda voce:** Un'altra spiegazione.

9.6. Spaziatura tra le voci

Di default le voci di lista sono compatte. Puoi aumentare la spaziatura tra le voci:

```
#set list(spacing: 1.5em)  
- Prima voce  
- Seconda voce (ora c'è più spazio tra le voci)
```

10. Allineamento e dimensioni del testo

Ci potrebbero essere ragioni per cui nel documento potreste aver bisogno di cambiare l'allineamento del testo o la sua dimensione. Vediamo con degli esempi come fare.

Per scrivere del testo della grandezza di 1 cm scrivere il seguente codice:

```
#text(size: 1cm)[testo di 1 cm]
```

Otterremo:

testo di 1 cm

Per scrivere del testo della grandezza di 0,7 cm di colore rosso scrivere il seguente codice:

```
#text(red, size: 0.7cm)[Testo in rosso di 0,7 cm]
```

Otterremo:

Testo in rosso di 0,7 cm

Per scrivere del testo della grandezza di 0,7 cm di colore rosso centrato scrivere il seguente codice:

```
#set align(center)  
#text(red, size: 0.7cm)[Testo in rosso di 0,7 cm]  
#set align(left)
```

Otterremo:

Testo in rosso di 0,7 cm

Per scrivere del testo allineato al margine destro scrivere il seguente codice:

```
#align(right)[testo allineato a destra]
```

Otterremo:

testo allineato a destra

Per scrivere del testo centrato scrivere il seguente codice:

```
#align(center)[testo centrato]
```

Otterremo:

testo centrato

11. Link web e link interni al documento

11.1. Link con URL nel documento

Per inserire nel documento un indirizzo web scrivere il seguente codice:

```
#link("https://www.puntoinformaticofree.it")
```

E otterremo:

<https://www.puntoinformaticofree.it>

11.2. Link con etichetta nel collegamento

Per inserire nel documento un indirizzo web con l'etichetta scrivere il seguente codice:

```
#link("https://www.puntoinformaticofree.it")[Punto Informatico Free]
```

E otterremo:

[Punto Informatico Free](https://www.puntoinformaticofree.it)

11.3. Link per mandare una email

Per mandare una email si può mettere un link che apre automaticamente il programma di posta elettronica. Per farlo digitare il seguente codice:

```
#link("mailto:whynotlogic@gmail.com")[mandami una email]
```

E otterremo:

[mandami una email](mailto:whynotlogic@gmail.com)

11.4. Link interni al documento

Puoi assegnare un'etichetta a qualsiasi titolo (o figura, o equazione) e poi creare un link da un'altra parte del documento. Questo riferimento creerà un link che ci porterà all'etichetta selezionata.

Quindi prima andare nella pagina del documento in cui vogliamo assegnare l'etichetta e dopo il titolo, scrivi il codice `<nome-etichetta>` tra parentesi angolari:

= **Inserire una tabella** `<tabelle1>`

Andare nel documento dove vogliamo mettere il riferimento all'etichetta inserendo il codice del link all'etichetta. Quindi nel mio caso se scrivo il codice:

Vedi il capitolo `#link(<tabelle1>)[Inserire una tabella]`.

Ottingo:

Vedi il capitolo [inserire una tabella](#).

11.5. Link a un determinato numero di pagina

Per inserire nel documento un link a un determinato numero di pagina scrivere il seguente codice:

Vedere a `#link((page: 20, x: 2.5pt, y: 2.5pt))[pagina 20]`

E otterremo:

Vedere a [pagina 20](#)

12. Inserire immagini

Possiamo inserire nel documento una immagine presente in una cartella nel PC. L'immagine deve essere inserita nella stessa cartella del documento. Oppure all'interno di una cartella posizionata all'interno della stessa cartella del documento. Le immagini possono essere in formato JPG, PNG o SVG.

12.1. Inserire immagini allineate a sinistra

Immaginiamo di aver inserito nella cartella «immagini» un file col nome «gatto.png» con l'immagine di un gatto. E vogliamo che questa immagine abbia la larghezza di 6,5 cm.

Se non specifichiamo l'allineamento dell'immagine per default viene allineata a sinistra.

Digitiamo il seguente codice:

```
#image("/gatto.png",  
width: 6.5cm)
```

Produrremo:



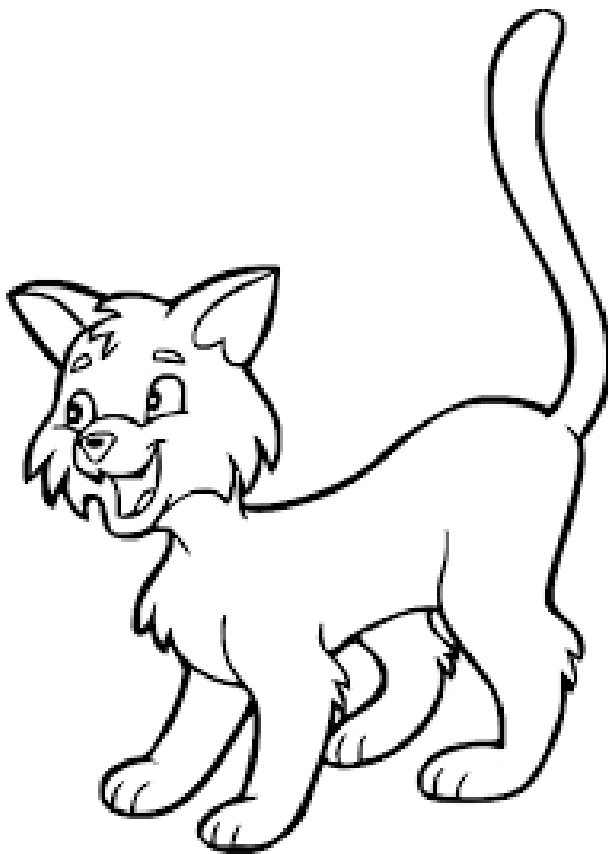
12.2. Inserire immagini centrate

Immaginiamo di aver inserito nella cartella «immagini» un file col nome «gatto.png» con l'immagine di un gatto. E vogliamo che questa immagine abbia la larghezza di 8 cm.

Se vogliamo centrare questa immagine dobbiamo digitare il seguente codice:

```
#align(center,  
image("/immagini/gatto.jpg",  
width: 8cm),  
)
```

Produrremo:



12.3. Inserire immagini centrate con didascalia

Immaginiamo di aver inserito nella cartella «immagini» un file col nome «gatto.png» con l'immagine di un gatto. E vogliamo che questa immagine abbia la larghezza di 8 cm e la didascalia «Immagine del gatto».

Per le immagini più professionali (con didascalia, numerazione automatica, e inclusione nell'indice delle figure) usa il codice `#figure()`, quindi dobbiamo digitare il seguente codice:

```
#figure(center,  
image("gatto.jpg",  
width: 8cm),  
)
```

Produrremo:

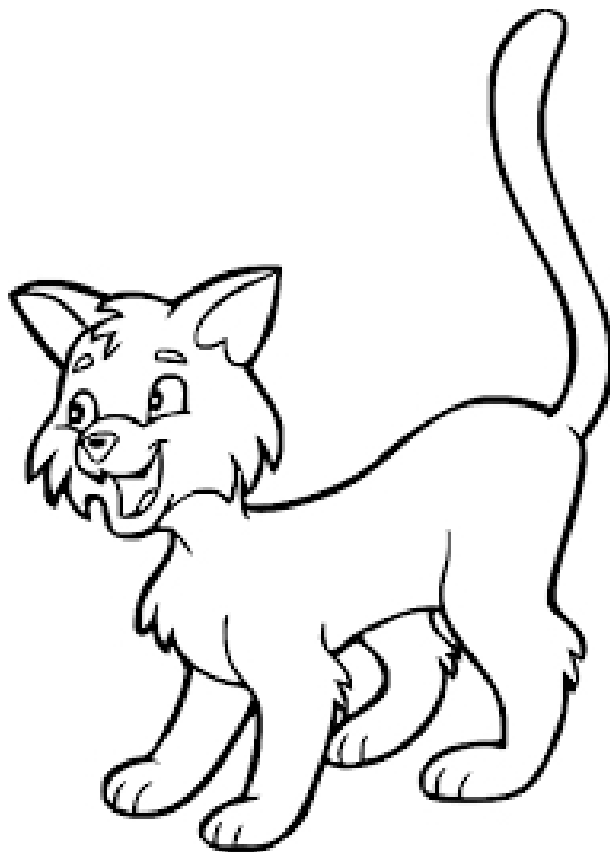


Figura 1: Immagine del gatto

12.4. Inserire immagini allineate a destra

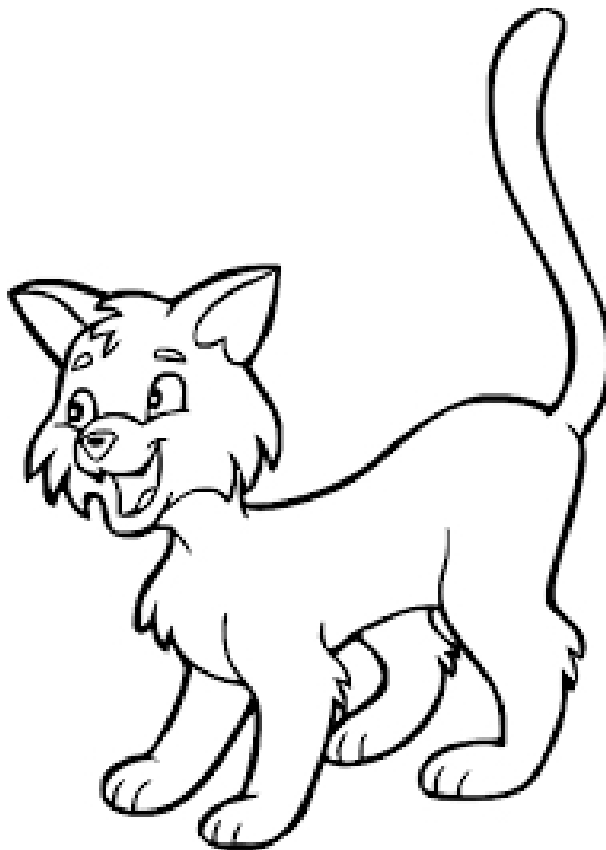
Immaginiamo di aver inserito nella cartella «immagini» un file col nome «gatto.png» con l'immagine di un gatto. E vogliamo che questa immagine abbia la larghezza di 8 cm.

Se vogliamo allineare a destra questa immagine dobbiamo digitare il seguente codice:

```
#align(right,  
image("gatto.jpg",
```

```
width: 8cm),  
)
```

Produrremo:



12.5. Inserire immagini con la larghezza dei margini della pagina

Immaginiamo invece che vogliamo inserire nel documento l'immagine del desktop di linux. Il file è posizionato nella cartella «immagini» col nome di «desktop.png». Questa immagine vogliamo che abbia la larghezza del 100% e la didascalia «desktop di Omarchy». Dovremo scrivere il seguente codice:

```
#figure(  
  image("desktop.png",  
  width: 100%),  
  caption: [desktop di Omarchy]  
)
```

Produrremo:

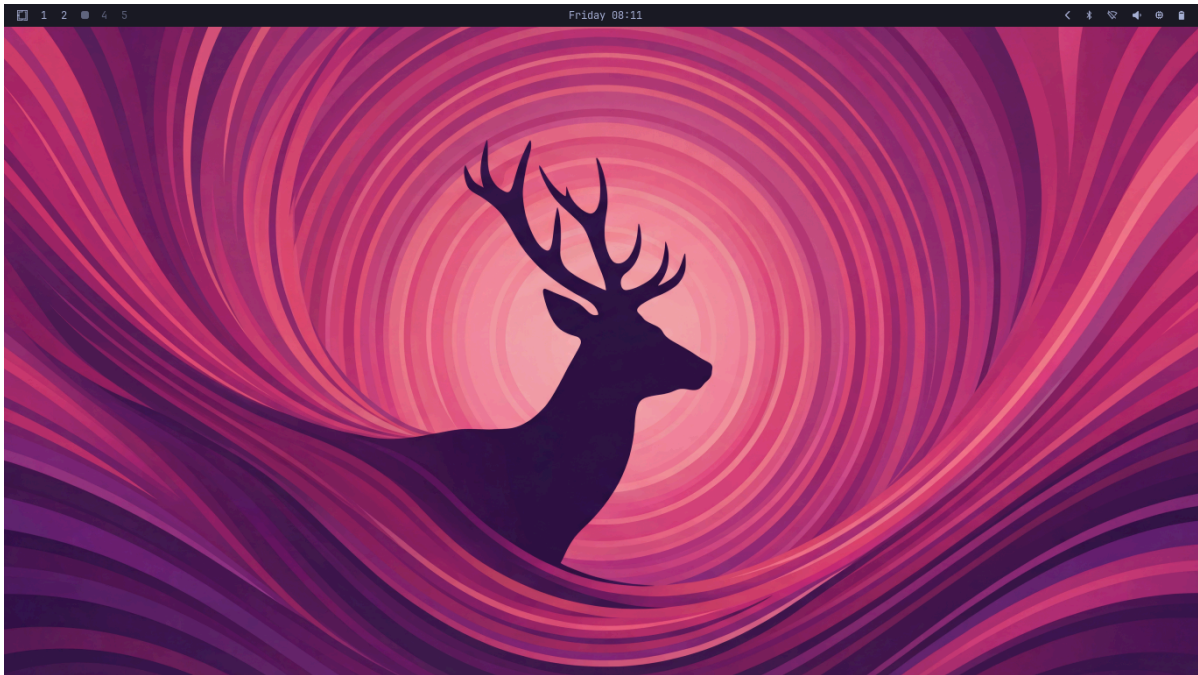


Figura 2: desktop di Omarchy

12.6. Inserire immagini affiancate

Immaginiamo invece che vogliamo inserire nel documento l'immagine di un cane affiancata all'immagine di un gatto. Entrambi i files sono posizionati nella cartella «immagini» con i nomi di «cane.png» e «gatto2.png». Questa immagini vogliamo che abbiano le larghezza del 50% e le didascalie «Immagine cane» e «Immagine gatto». Dovremo scrivere il seguente codice:

```
#grid(  
  columns: (1fr, 1fr),  
  gutter: 1.5em,  
  figure(  
    image("/immagini/cane.png"),  
    caption: [Disegno cane.],  
  ),  
  figure(  
    image("/immagini/gatto2.png"),  
    caption: [Disegno gatto.],  
  ),  
)
```

Produrremo:



Figura 3: Disegno cane.



Figura 4: Disegno gatto.

13. Rappresentare il codice

13.1. Rappresentare il codice per blocchi

Prima inserire nelle impostazioni del documento il seguente codice:

```
#show raw.where(block: true): it => {  
  block(  
    fill: luma(220),  
    width: 100%,  
    inset: (x: 1.3em, y: 1em),  
    radius: 6pt,  
    stroke: 0.6pt + black,  
  ) [  
    #set text( font: ("Libertinus Serif"), size: 12pt)  
    #set par(justify: false, leading: 0.7em)  
    #it  
  ]  
}
```

Typst rappresenta l'evidenziazione per diversi tipi di codice, quindi in base il codice da rappresentare cambia il codice da inserire.

Per rappresentare una o più righe come codice nel documento scrivere il seguente codice:

```
```Identificatore  
codice da rappresentare
```
```

Quindi se voglio scrivere rappresentare il codice Typst scrivere il seguente codice:

```
```typ  
codice da rappresentare
```
```

Typst supporta l'evidenziazione per moltissimi linguaggi. Vediamo nella seguente tabella per ogni linguaggio l'identificatore da inserire nel codice:

| Linguaggio | Identificatore | Linguaggio | Identificatore |
|------------|-----------------------|------------|---|
| Python | <code>python</code> | JavaScript | <code>javascript</code> o <code>js</code> |
| Java | <code>java</code> | C/C++ | <code>c</code> , <code>cpp</code> |
| Rust | <code>rust</code> | Go | <code>go</code> |
| HTML | <code>html</code> | CSS | <code>css</code> |
| SQL | <code>sql</code> | Bash | <code>bash</code> , <code>sh</code> |
| LaTeX | <code>latex</code> | Typst | <code>typ</code> |
| JSON | <code>json</code> | YAML | <code>yaml</code> |
| Markdown | <code>markdown</code> | R | <code>r</code> |
| MATLAB | <code>matlab</code> | Julia | <code>julia</code> |

13.2. Rappresentare una porzione di riga come codice:

Prima inserire nelle impostazioni del documento il seguente codice:

```
#show raw.where(block: false): it => {
  box(
    fill: luma(220),
    inset: (x: 4pt),
    outset: (y: 2pt),
    radius: 4pt,
    stroke: 0.5pt + black,
  )[
    #set text( font: ("Libertinus Serif"), size: 12pt)
    #it
  ]
}
```

Per rappresentare una o più parole come codice digitare:

```
`codice da rappresentare`
```

Produrremo:

```
codice da rappresentare
```

14. Inserire una tabella

E' possibile inserire una tabella nel documento. Per capirlo mostriamolo con degli esempi che ci permettono di vedere tutte le possibilità tipografiche più importanti per inserire una tabella.

14.1. Inserire una tabella semplice

Inseriamo una tabella semplice scrivendo il seguente codice:

```
#table(  
  columns: 2,  
  inset: 0.5em,  
  fill: (x, y) => if y == 0 { highlight.fill },  
  [Colonna 1], [Colonna 2],  
  [A], [B],  
  [C], [D],  
  [E], [F],  
  [G], [H],  
)
```

Otterremo il seguente risultato:

| Colonna 1 | Colonna 2 |
|-----------|-----------|
| A | B |
| C | D |
| E | F |
| G | H |

14.2. Inserire una tabella con colori alternati

Vediamo con un esempio come inserire una tabella con colori alternati. Inseriamo una tabella scrivendo il seguente codice:

```
#table(  
  columns: 3,
```

```
inset: 0.5em,
align: (left, center, right),
fill: (col, row) => if calc.odd(row) { luma(200) } else { white },
[Colonna 1], [Colonna 2], [Colonna 3],
[A], [B], [C],
[D], [E], [F],
[G], [H], [I],
[L], [M], [N],
)
```

Otterremo il seguente risultato:

| Colonna 1 | Colonna 2 | Colonna 3 |
|-----------|-----------|-----------|
| A | B | C |
| D | E | F |
| G | H | I |
| L | M | N |

14.3. Inserire una tabella con intestazioni stilizzate

Vediamo con un esempio come inserire una tabella con intestazioni stilizzate. Inseriamo una tabella scrivendo il seguente codice:

```
#table(
  columns: (1fr, 1fr, 1fr),
  inset: 0.5em,
  stroke: 0.6pt + luma(180),
  fill: (col, row) => if row == 0
  { rgb("#1a3a5c") }
  else if calc.odd(row) { white }
  else { rgb("#f4f6f7") },
  table.cell(fill: color.green.darken(65%))[#set text(fill: white, weight: "bold"); Colonna
  1],
  table.cell(fill: color.green.darken(65%))[#set text(fill: white, weight: "bold"); Colonna
  2],
  table.cell(fill: color.green.darken(65%))[#set text(fill: white, weight: "bold"); Colonna
  3],
  [A], [B], [C],
  [D], [E], [F],
)
```

```
[G], [H], [I],
[L], [M], [N],
)
```

Otterremo il seguente risultato:

| Colonna 1 | Colonna 2 | Colonna 3 |
|-----------|-----------|-----------|
| A | B | C |
| D | E | F |
| G | H | I |
| L | M | N |

14.4. Inserire una tabella con lunghezza colonne definita

Possiamo definire la lunghezza delle colonne scrivendo i seguenti codici nella riga `columns` della tabella:

```
// 3 colonne uguali
columns: 3
// Larghezze specifiche
columns: (3cm, 2cm, 4cm)
// Proporzioni: 1fr = "il resto dello spazio"
columns: (1fr, 2fr, 1fr)
// Prima e terza colonna uguali, seconda larga il doppio
// Mix di fisso e proporzionale
columns: (auto, 1fr, 2cm)
// Prima colonna: si adatta al contenuto
// Seconda: prende lo spazio rimanente
// Terza: fissa 2cm
```

Quindi se scriviamo il seguente codice:

```
#table(
  columns: (1fr, 2fr, 1fr),
  inset: 0.5em,
  align: (left, center, center),
  fill: (col, row) => if calc.odd(row) { luma(200) } else { white },
  [Colonna 1], [Colonna 2], [Colonna 3],
```

```
[A], [B], [C],  
[D], [E], [F],  
[G], [H], [I],  
[L], [M], [N],  
)
```

Otterremo il seguente risultato:

| Colonna 1 | Colonna 2 | Colonna 3 |
|-----------|-----------|-----------|
| A | B | C |
| D | E | F |
| G | H | I |
| L | M | N |

14.5. Bordi (stroke)

```
// Bordo uniforme  
stroke: 0.5pt + black  
// Nessun bordo  
stroke: none  
// Bordo solo orizzontale  
stroke: (top: 0.5pt + black, bottom: 0.5pt + black)  
// Bordo personalizzato per cella  
// (nella funzione fill puoi tornare stroke)
```

15. Inserire spazi nel documento

15.1. Spazio orizzontale

Per inserire uno spazio orizzontale per esempio di due centimetri, scrivere il seguente codice:

```
#h(2cm)
```

15.2. Spazio verticale

Per inserire uno spazio verticale per esempio di due centimetri, scrivere il seguente codice:

```
#v(2cm)
```

15.3. Spazio infinito

Per inserire uno spazio orizzontale infinito che spinge il testo ai lati della riga, scrivere il seguente codice:

```
#h(1fr)
```

15.4. Spazio non divisibile

Per inserire uno spazio non divisibile tra due parole inserire il carattere `~`, come nel seguente codice:

```
Dott.~Rossi
```

15.5. Inserire riga vuota nel documento

Per inserire una riga vuota nel documento scrivere ad inizio riga il seguente codice:

```
\
```

15.6. Inserire una interruzione di pagina

Per inserire una interruzione di pagina scrivere il seguente codice:

```
#pagebreak()
```

15.7. Inserire una linea orizzontale

Per inserire una linea orizzontale scrivere il seguente codice:

```
#line(length: 100%)
```

Oppure il seguente codice:

```
#line(length: 100%, stroke: 0.5pt)
```

16. Matematica

Uno dei punti di forza di Typst è la sua sintassi matematica: più pulita di LaTeX, facile da imparare, e con risultati tipograficamente eccellenti.

16.1. Modalità matematica

Come avevamo accennato in precedenza, ci sono (essenzialmente) due modi per scrivere matematica in Typst:

16.1.1. Matematica inline

La matematica **inline** appare nel corpo del testo, circondata da `$...$` (dollaro singolo, senza spazi).

Se scriviamo il codice:

La formula di Eulero è `$e^{i \pi} + 1 = 0$`,
 considerata la più bella della matematica.
 Il teorema di Pitagora: `$a^2 + b^2 = c^2$`.

Otteniamo:

La formula di Eulero è $e^{i\pi} + 1 = 0$, considerata la più bella della matematica. Il teorema di Pitagora: $a^2 + b^2 = c^2$.

16.1.2. Matematica a blocco (display)

La matematica **a blocco** appare su una riga propria, centrata e con simboli più grandi. Si ottiene con `$(...)$` su una riga separata, oppure aggiungendo uno spazio dopo il `$` iniziale.

Se scriviamo il codice:

L'equazione di Einstein:
`$ E = m c^2 $`
 Oppure l'identità di Eulero in grande:
`$ e^{i \pi} + 1 = 0 $`

Otteniamo: L'equazione di Einstein:

$$E = mc^2$$

Oppure l'identità di Eulero in grande:

$$e^{i\pi} + 1 = 0$$

La differenza tra inline e display è lo spazio dopo `$`:

- `$formula$` = inline (nel flusso del testo)
- `$ formula $` = display (su riga propria, centrata)

16.2. Operatori di base

| Operazione | Sintassi Typst | Risultato |
|---------------------|-------------------------|---------------|
| Addizione | <code>a + b</code> | $a + b$ |
| Sottrazione | <code>a - b</code> | $a - b$ |
| Moltiplicazione | <code>a dot b</code> | $a \cdot b$ |
| Prodotto vettoriale | <code>a times b</code> | $a \times b$ |
| Divisione | <code>a / b</code> | $\frac{a}{b}$ |
| Uguale | <code>a = b</code> | $a = b$ |
| Diverso | <code>a != b</code> | $a \neq b$ |
| Minore | <code>a < b</code> | $a < b$ |
| Maggiore | <code>a > b</code> | $a > b$ |
| Minore uguale | <code>a <= b</code> | $a \leq b$ |
| Maggiore uguale | <code>a >= b</code> | $a \geq b$ |
| Approssimato | <code>a approx b</code> | $a \approx b$ |
| Proporzionale | <code>a prop b</code> | $a \propto b$ |

16.3. Potenze e pedici

```
// Potenza semplice
$x^2$ //  $x^2$ 
$x^{(n+1)}$ //  $x^{(n+1)}$  — serve la parentesi!
// Pedice semplice
$x_i$ //  $x_i$ 
$x_{(i,j)} //  $x_{(i,j)}$  con pedice composto
// Insieme
```

`$x_i^2$` // x_i^2
`$x_{(i+1)}^{(2n)}$` // con espressioni composte

Se scriviamo il codice:

`$x^2 + y^2 = r^2$` è la circonferenza.
`$a_n = a_{(n-1)} + a_{(n-2)}$` è Fibonacci.
 La serie `$\sum_{(n=0)}^{(+infinity)} x^n = 1/(1-x)$`
 converge per `$|x| < 1$`.

Otteniamo:

$x^2 + y^2 = r^2$ è la circonferenza.

$a_n = a_{n-1} + a_{n-2}$ è Fibonacci.

La serie $\sum_{n=0}^{+\infty} x^n = \frac{1}{1-x}$
 converge per $|x| < 1$.

16.4. Frazioni

`$a/b$` // Frazione a/b (forma compatta in inline)
`$\frac{a}{b}$` // Frazione esplicita (sempre con sbarra)
`$(a+b)/(c+d)$` // Frazione con espressioni

Se scriviamo il codice:

La derivata: `$\frac{d f}{d x} = \lim_{(h \rightarrow 0)} \frac{f(x+h)-f(x)}{h}$`
 L'integrale: `$\int_a^b f(x) d x = F(b) - F(a)$`

Otteniamo:

La derivata: $\frac{df}{dx} = \lim_{h \rightarrow 0} \frac{f(x+h)-f(x)}{h}$ L'integrale:

$$\int_a^b f(x) dx = F(b) - F(a)$$

16.5. Radici

`$\sqrt{x}$` // Radice quadrata
`$\sqrt{x + y}$` // Radice di un'espressione

`$root(3, x)$` // Radice cubica
`$root(n, x^n)$` // Radice n-esima

Se scriviamo il codice:

La soluzione è `$x = frac(-b plus.minus sqrt(b^2 - 4a c), 2a)$`.

Otteniamo:

La soluzione è $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$.

16.6. Simboli speciali comuni

16.6.1. Lettere greche

| Simbolo | Codice | Simbolo | Codice | Simbolo | Codice |
|----------|--------------------|---------------|----------------------|-----------|---------------------|
| α | <code>alpha</code> | β | <code>beta</code> | γ | <code>gamma</code> |
| δ | <code>delta</code> | ε | <code>epsilon</code> | ζ | <code>zeta</code> |
| η | <code>eta</code> | θ | <code>theta</code> | λ | <code>lambda</code> |
| μ | <code>mu</code> | ν | <code>nu</code> | ξ | <code>xi</code> |
| π | <code>pi</code> | ρ | <code>rho</code> | σ | <code>sigma</code> |
| τ | <code>tau</code> | φ | <code>phi</code> | ψ | <code>psi</code> |
| ω | <code>omega</code> | Δ | <code>Delta</code> | Σ | <code>Sigma</code> |
| Ω | <code>Omega</code> | Λ | <code>Lambda</code> | Π | <code>Pi</code> |

16.6.2. Altri simboli frequenti

| Simbolo | Codice | Simbolo | Codice |
|-------------------|-------------------------|---------------|-----------------------------|
| ∞ | <code>infinity</code> | ∂ | <code>partial</code> |
| ∇ | <code>nabla</code> | $\forall$ | <code>forall</code> |
| $\exists$ | <code>exists</code> | $\in$ | <code>in</code> |
| $\subset$ | <code>subset</code> | $\cup$ | <code>union</code> |
| $\cap$ | <code>inter</code> | $\rightarrow$ | <code>arrow.r</code> |
| $\leftrightarrow$ | <code>arrow.l.r</code> | $\Rightarrow$ | <code>arrow.r.double</code> |
| $\pm$ | <code>plus.minus</code> | $\mp$ | <code>minus.plus</code> |

16.7. Sommatorie, produttorie, limiti

```
// Sommatoria
$sum_(i=0)^n a_i$
// Produttoria
$product_(k=1)^n k = n!$
// Limite
$lim_(x -> 0) frac(sin x, x) = 1$
// In modalità display (più grande)
$ sum_(i=0)^n i^2 = frac(n(n+1)(2n+1), 6) $
```

Se scriviamo il codice:

```
$ sum_(i=1)^n i = frac(n(n+1), 2) $
$ lim_(n->infinity) (1 + 1/n)^n = e $
```

Otteniamo:

$$\sum_{i=1}^n i = \frac{n(n+1)}{2}$$

$$\lim_{n \rightarrow \infty} \left(1 + \frac{1}{n}\right)^n = e$$

16.8. Vettori e matrici

```
// Vettore colonna
$vec(a, b, c)$
// Matrice
$mat(a, b; c, d)$ // Il punto e virgola separa le righe
// Matrice con bordi
$mat(delim: "[", a, b; c, d)$
$mat(delim: "|", a, b; c, d)$ // Determinante
```

Se scriviamo il codice:

```
La matrice di rotazione è:
$ R(theta) = mat(cos theta, -sin theta; sin theta, cos theta) $
Un sistema lineare:
$ mat(a, b; c, d) dot vec(x, y) = vec(e, f) $
```

Otteniamo:

La matrice di rotazione è:

$$R(\theta) = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$$

Un sistema lineare:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} e \\ f \end{pmatrix}$$

16.9. Testo all'interno delle formule

Per inserire testo normale dentro una formula usa le virgolette:

```
$f(x) = x^2 \text{quad "per"} x > 0$
```

Se scriviamo il codice:

```
$f(x) = \text{cases}(x^2 \ \&"se" \ x \geq 0, -x^2 \ \&"se" \ x < 0)$
```

Otteniamo:

$$f(x) = \begin{cases} x^2 & \text{se } x \geq 0 \\ -x^2 & \text{se } x < 0 \end{cases}$$

16.10. Formule numerate

Per le formule numerate in un documento scientifico:

```
#set math.equation(numbering: "(1)")
$ E = m c^2 $ <eq-einstein>
Come dimostrato in @eq-einstein...
```

16.11. Spazi in matematica

In modalità matematica gli spazi normali vengono ignorati. Per aggiungere spaziatura usa:

```
$a \text{quad } b$ // spazio grande
$a \text{wide } b$ // spazio molto grande
```

`$a thin b$` // spazio sottile

`$a med b$` // spazio medio

`$a space b$` // spazio normale

17. Inserire simboli

Per scrivere un simbolo nel documento bisogna scrivere il suo codice corrispondente tra 2 segni del dollaro (\$)

Se voglio scrivere il carattere greco della `Lambda` scrivo il seguente codice:

```
$Lambda$
```

Λ

Se voglio inserire una freccia squadrata scrivo il seguente codice:

```
$arrow.quad$
```

⇒

Al seguente link troverete a lista dei simboli utilizzati in Typst:

- <https://typst.app/docs/reference/symbols/sym/>

18. Inserire caselle di testo nel documento

Vediamo ora più codici per inserire caselle di testo sul documento.

Se scrivo il codice:

```
#block(stroke: red, inset: 1em, width: 100%)[
  testo all'interno del blocco di testo
]
```

Otengo:

testo all'interno del blocco di testo

Se scrivo il codice:

```
#rect(width: 5cm, text(red)[Ciao *ragazzi*])
```

Otengo:

ciao **ragazzi**

Se scrivo il codice:

```
#block(stroke: red, inset: 1em)[Blocco di testo]
```

Otengo:

Blocco di testo

Se scrivo il codice:

```
#rect[Blocco di testo]
```

Otengo:

Blocco di testo

19. Inserire una nota a piè di pagina

Per inserire una nota a piè di pagina inserire il seguente codice:

```
Testo del documento.  
#footnote[nota a piè di pagina]
```

E otterremo il risultato mostrato in questa pagina della guida:

Testo del documento.¹

Le note a piè di pagina che aggiungeremo saranno numerate progressivamente in automatico.

¹nota a piè di pagina

20. Le regole set

Le regole `(set)` sono uno dei meccanismi più potenti e eleganti di Typst. Ti permettono di modificare i **parametri predefiniti** di qualsiasi elemento, una volta per tutte.

20.1. Cos'è una regola set?

Immagina di volere che tutto il testo del tuo documento sia di dimensione 12pt con font «Libertinus Serif». Potresti scrivere `#text(size: 12pt, font: "Libertinus Serif") [...]` attorno a ogni singolo paragrafo. Potresti farlo ma prederesti tanto tempo, sarebbe una follia, usiamo typst e con `(set)`, dici una sola volta:

```
#set text(size: 12pt, font: "Libertinus Serif")
```

E da quel punto in poi, **tutto** il testo usa quelle impostazioni.

Una regola `(set)` modifica i parametri di default di una funzione (o elemento) per tutto ciò che viene dopo.

20.2. set text — il testo

```
#set text(  
  font: "Libertinus Serif", // Nome del font  
  size: 12pt, // Dimensione  
  fill: luma(30), // Colore (quasi nero)  
  weight: "regular", // "bold", "light", "regular", ...  
  style: "normal", // "italic", "oblique"  
  tracking: 0pt, // Spaziatura tra le lettere  
  spacing: 100%, // Spaziatura tra le parole  
  lang: "it", // Lingua (per trattini, ecc.)  
  hyphenate: true, // Sillabazione automatica  
)
```

20.3. set par — i paragrafi

```
#set par(  
  justify: true, // Giustificazione (allineamento sx e dx)  
  leading: 0.75em, // Interlinea (spazio tra le righe)  
  spacing: 1.2em, // Spazio tra i paragrafi  
  first-line-indent: 1.5em, // Rientro prima riga  
  hanging-indent: 0pt, // Rientro sospeso  
)
```

20.4. set heading — i titoli

```
#set heading(  
  numbering: "1.1", // Schema di numerazione  
  outlined: true, // Appare nell'indice?  
)
```

20.5. set page — la pagina

```
#set page(  
  paper: "a4", // Formato carta  
  margin: ( // Margini  
    top: 2.5cm,  
    bottom: 2.5cm,  
    left: 3cm,  
    right: 2.5cm,  
  ),  
  numbering: "1", // Formato numero pagina  
  number-align: center, // Posizione numero: center, right, left  
  columns: 1, // Numero di colonne  
  fill: white, // Colore sfondo pagina  
  flipped: false, // Orientamento (true = landscape)  
)
```

Formati carta disponibili:

| Formato | Codice | Formato | Codice |
|---------|--------|---------------|---------------------|
| A0 | "a0" | A5 | "a5" |
| A1 | "a1" | A6 | "a6" |
| A2 | "a2" | US Letter | "us-letter" |
| A3 | "a3" | US Legal | "us-legal" |
| A4 | "a4" | Presentazione | "presentation-16-9" |

20.6. set list e set enum

```
// Lista puntata
#set list(
  marker: "→", // Marcatore personalizzato
  indent: 1.5em, // Rientro
  spacing: 0.6em, // Spazio tra le voci
)
// Lista numerata
#set enum(
  numbering: "1.", // Schema numerazione
  indent: 1.5em,
  spacing: 0.6em,
)
```

20.7. set table — le tabelle

```
#set table(
  stroke: 0.5pt + luma(200), // Bordo di default
  inset: 0.5em, // Padding celle
  align: left, // Allineamento
  fill: (col, row) => // Funzione colore
    if calc.odd(row) { luma(245) } else { white },
)
```

20.8. set figure — le figure

```
#set figure(  
  supplement: "Figura", // Testo prima del numero  
  gap: 0.8em, // Spazio tra contenuto e didascalia  
  numbering: "1", // Schema numerazione  
  placement: none, // Posizionamento  
)
```

20.9. set math.equation — le equazioni

```
#set math.equation(  
  numbering: "(1)", // Le equazioni vengono numerate  
  supplement: "Eq.", // Testo nel riferimento  
)
```

20.10. Lo scope delle regole set

Le regole `set` hanno uno **scope**: si applicano da dove sono definite in poi, oppure solo dentro il blocco `{ }` o `[ ]` in cui si trovano.

Per esempio se scrivo il codice:

```
Testo normale di 12pt.  
#{  
  set text(size: 14pt, fill: red)  
  [Questo testo è grande e rosso.]  
}  
Di nuovo testo normale di 12pt.
```

Ottengo:

Testo normale di 12pt. **Questo testo è grande e rosso.** Di nuovo testo normale di 12pt.

21. Le regole show

Le regole `show` sono più potenti delle `set`: non cambiano solo i **parametri** di un elemento, ma ne **trasformano completamente l'aspetto**, intercettando l'elemento prima che venga reso nel PDF.

Una regola `show` intercetta un elemento (o tutti gli elementi di un certo tipo) e lo trasforma in qualcosa di diverso. Sintassi: `#show selettore: trasformazione`.

21.1. La differenza tra set e show

```
// set: modifica i parametri → cambia proprietà
#set text(size: 14pt) // Il testo diventa più grande
// show: trasforma l'elemento → può cambiare struttura
#show strong: it => text(fill: red, it.body) // Il grassetto diventa rosso
```

21.2. Trasformare il grassetto

```
// Rendi il grassetto di colore blu
#show strong: it => text(fill: rgb("#1a3a5c"), it.body)
Questo è testo normale.
Questo è testo in grassetto, ora appare blu.
Anche questa frase è in grassetto blu.
```

21.3. Trasformare i link

```
#show link: it => {
  set text(fill: rgb("#2471a3"))
  underline(stroke: (paint: rgb("#2471a3"), dash: "dashed"), it)
}
Visita #link("https://typst.app")[Typst] o
#link("https://github.com/typst/typst")[GitHub].
```

21.4. Selettori: `.where()`

Puoi applicare la regola `show` solo a un sottoinsieme di elementi usando `.where()`:

```
// Solo i titoli di livello 1
#show heading.where(level: 1): it => ...
// Solo il codice a blocco (non inline)
#show raw.where(block: true): it => ...
// Solo le figure che contengono immagini
#show figure.where(kind: image): it => ...
// Solo le figure che contengono tabelle
#show figure.where(kind: table): it => ...
```

Altre impostazioni che possiamo fare applicando la regola `show` solo a un sottoinsieme di elementi usando `.where()` sono per esempio di decidere che i titoli di 1° livello siano di colore rosso. Per farlo aggiungere il seguente codice:

```
#show heading.where(level: 1): set text(red)
```

Oppure che i titoli di secondo livello siano centrati aggiungendo il codice:

```
#show heading.where(level: 2): set align(center)
```

Applicando la regola `show` solo a un sottoinsieme di elementi usando `.where()` possiamo per esempio modificare le impostazioni per lo stile per i blocchi di codice:

```
// _____
// STILE BLOCCHI DI CODICE
// _____
#show raw.where(block: true): it => {
  block(
    fill: luma(220),
    width: 100%,
    inset: (x: 1.3em, y: 1em),
    radius: 6pt,
    stroke: 0.6pt + black,
  )[
    #set text( font: ("Libertinus Serif"), size: 12pt)
    #set par(justify: false, leading: 0.7em)
    #it
  ]
}
```

```
#show raw.where(block: false): it => {  
  box(  
    fill: luma(220),  
    inset: (x: 4pt),  
    outset: (y: 2pt),  
    radius: 4pt,  
    stroke: 0.5pt + black,  
  ) [  
    #set text( font: ("Libertinus Serif"), size: 12pt)  
    #it  
  ]  
}
```

21.5. Trasformare i titoli

Per esempio per trasformare i titoli di colore viola:

```
#show heading.where(level: 2): it => {  
  v(1em)  
  block [  
    #set text(fill: rgb("#7d3c98"), size: 14pt)  
    #it.body  
    #v(-0.2em)  
    #line(length: 100%, stroke: 2pt + rgb("#7d3c98"))  
  ]  
  v(0.4em)  
}  
  
== Prima sezione viola  
Testo sotto la prima sezione.  
  
== Seconda sezione viola  
Testo sotto la seconda sezione.
```

21.6. Trasformare il documento intero (show globale)

La forma più potente: trasforma il contenuto di tutto il documento wrappandolo in qualcosa:

```
// Metti tutto il documento in una colonna con max-width
#show: body => block(
  width: 80%,
  inset: 2cm,
)(#body)
// Oppure applica impostazioni globali
#show: body => {
  set text(size: 12pt, font: "Libertinus Serif")
  set par(justify: true)
  body
}
```

22. Box e blocchi

`box` e `block` sono i due contenitori fondamentali in Typst. Capire la differenza è essenziale per controllare il layout.

22.1. box vs block: la differenza

| | box | block |
|------------|-------------------------------|-----------------------------------|
| Tipo | Inline (nel flusso del testo) | Block (occupa tutta la larghezza) |
| Uso | Dentro una riga di testo | Elemento separato |
| Dimensione | Si adatta al contenuto | Larghezza 100% di default |
| Va a capo? | No (sta in riga) | Sì (nuovo rigo) |

Se per esempio scriviamo il seguente codice:

```
#block(fill: rgb("#eaf4fb"), inset: 1em, radius: 5pt)[
  Questo è un blocco separato che occupa
  tutta la larghezza disponibile nella riga del documento
]
```

Otteniamo:

Questo è un blocco separato che occupa tutta la larghezza disponibile nella riga del documento

22.2. Parametri di box e block

Entrambi condividono questi parametri principali:

```
#block(
  // Dimensioni
  width: 80%, // Larghezza (length, ratio, o auto)
  height: auto, // Altezza

  // Spaziatura interna
```

```

inset: 1em, // Uguale su tutti i lati
inset: (x: 1em, y: 0.5em), // Orizzontale e verticale
inset: (top: 0.5em, bottom: 0.5em, left: 1em, right: 1em),

// Spaziatura esterna (solo block)
above: 1em, // Spazio sopra
below: 1em, // Spazio sotto

// Aspetto
fill: white, // Colore sfondo
stroke: 1pt + black, // Bordo
radius: 5pt, // Angoli arrotondati

// Comportamento
clip: false, // Taglia il contenuto che eccede?
breakable: true, // Si spezza tra le pagine? (solo block)
)

```

22.2.1. Angoli arrotondati personalizzati

```

#block(
  radius: (top-left: 10pt, top-right: 10pt,
    bottom-left: 0pt, bottom-right: 0pt),
)

```

22.2.2. Inset dettagliato

```

inset: (top: 0.5em, bottom: 0.5em, left: 1.5em, right: 0.5em)
inset: (x: 1em, y: 0.5em) // Scorciatoia

```

22.3. Impedire le interruzioni di pagina

Se non vuoi che un blocco venga spezzato tra due pagine:

```

#block(breakable: false)[
  Questa tabella importante non deve essere spezzata
  a metà tra due pagine.
]

```

```
#table(...)  
]
```

22.4. Stack: impilare elementi

`#stack()` impila elementi uno sopra l'altro o affiancati:

```
// Impila verticalmente (default)  
#stack(dir: ttb, spacing: 1em,  
  block(fill: red, width: 100%, height: 1cm)[],  
  block(fill: green, width: 100%, height: 1cm)[],  
)  
// Impila orizzontalmente  
#stack(dir: ltr, spacing: 0.5em,  
  rect(width: 2cm, fill: red),  
  rect(width: 3cm, fill: blue),  
)
```

22.5. Allineamento

`#align()` allinea il contenuto dentro il suo contenitore:

```
#align(center)[Centrato]  
#align(right)[A destra]  
#align(left)[A sinistra]  
#align(center + top)[Centro in alto]  
#align(right + bottom)[Destra in basso]
```

Se per esempio scriviamo il codice:

```
#block(width: 100%, height: 2cm, fill: luma(245),  
  stroke: 0.5pt + luma(200), radius: 5pt)[  
  #align(center + horizon)[  
    Questo testo è centrato orizzontalmente e verticalmente.  
  ]  
]
```

Otteniamo:

Questo testo è centrato orizzontalmente e verticalmente.

23. Impaginazione avanzata

23.1. Intestazioni e piè di pagina

Le intestazioni (`header`) e i piè di pagina (`footer`) si definiscono dentro `#set page()`.

23.1.1. Intestazione semplice

```
#set page(  
  header: [  
    Il Mio Documento  
    #h(1fr)  
    Capitolo 1  
  ],  
)
```

23.1.2. Intestazione con numero di pagina

```
#set page(  
  header: context {  
    [Il Mio Libro]  
    h(1fr)  
    counter(page).display()  
  },  
  footer: none,  
)
```

23.1.3. Il contesto (context) e la consapevolezza del documento

Per accedere a informazioni dinamiche del documento (numero di pagina, sezione corrente, contatori) devi usare il blocco `context`. È il modo in cui Typst gestisce le informazioni che cambiano nel documento:

```
#set page(
  numbering: "1",
  header: context {
    // Numero di pagina corrente
    let pn = counter(page).get().first()
    // Mostra l'intestazione solo dopo la prima pagina
    if pn > 1 {
      grid(
        columns: (1fr, auto),
        [Il Mio Documento],
        counter(page).display("1 / 1",
        both: true // "pagina corrente / totale pagine"
      ),
    )
    line(length: 100%, stroke: 0.5pt + luma(200))
  }
},
)
```

23.1.4. Intestazione con titolo sezione corrente

```
#set page(
  header: context {
    let headings = query(heading.where(level: 1).before(here()))
    if headings.len() > 0 {
      let current = headings.last()
      [_#current.body_]
      h(1fr)
      counter(page).display()
    }
  },
)
```

23.2. Numerazione pagine

Un po' come per un elenco puntato, ci sono diversi formati di numerazione delle pagine:

| Stringa | Risultato | Uso tipico |
|---------|------------|------------------|
| "1" | 1, 2, 3... | Corpo principale |

| | | |
|---------|---------------|----------------------|
| "i" | i, ii, iii... | Prefazione/appendice |
| "I" | I, II, III... | Numeri romani |
| "a" | a, b, c... | Appendici |
| "A" | A, B, C... | Appendici maiuscole |
| "1 / 1" | 3 / 12 | Pagina su totale |
| "- 1 -" | - 3 - | Stile rivista |

23.2.1. Cambiare numerazione nel documento

```
// Prefazione: numeri romani
#set page(numbering: "i")
[... prefazione ...]
// Reset e cambio a numeri arabi per il corpo
#counter(page).update(1)
#set page(numbering: "1")
[... capitoli ...]
```

23.3. Colonne

Per mettere il testo su più colonne:

```
// Due colonne per tutto il documento
#set page(columns: 2)
// Due colonne per un blocco specifico
#columns(2)[
  Questo testo è su due colonne.
  Qui c'è abbastanza testo per vedere
  come funziona il layout a colonne.
]
```

23.3.1. Saltare una colonna

```
#columns(2)[
  Testo nella prima colonna.
  #colbreak() // Salta alla colonna successiva
  Testo nella seconda colonna.
]
```

Se per esempio scriviamo il codice:

```
#columns(2, gutter: 1.5em)[  
  == Colonne affiancate  
  Questo testo fluisce naturalmente nella  
  prima colonna. Quando la prima è piena,  
  il testo continua nella seconda.  
  Le colonne sono molto utili per bollettini,  
  newsletter, o articoli in stile rivista.  
  #colbreak()  
  == Seconda colonna  
  Con #raw("colbreak()") puoi forzare il testo  
  a saltare immediatamente alla colonna successiva,  
  utile per iniziare una nuova sezione.  
]
```

Otteniamo:

Colonne affiancate

Questo testo fluisce naturalmente nella prima colonna. Quando la prima è piena, il testo continua nella seconda. Le colonne sono molto utili per bollettini, newsletter, o articoli in stile rivista.

Seconda colonna

Con `colbreak()` puoi forzare il testo a saltare immediatamente alla colonna successiva, utile per iniziare una nuova sezione.

23.4. Pagine speciali

23.4.1. Pagina con margini diversi

```
// Margini maggiori per la copertina  
#page(  
  margin: 0pt,  
  header: none,  
  footer: none,  
  numbering: none,  
)[
```

```
// Contenuto della copertina a piena pagina  
]
```

23.4.2. Pagina in landscape (orizzontale)

```
#page(flipped: true)[  
  // Contenuto su pagina orizzontale  
  // Utile per tabelle molto larghe  
]
```

23.4.3. Copertina libro

```
#page(  
  margin: 0pt,  
  header: none,  
  footer: none,  
  numbering: none,  
)[  
  // Sfondo con colore unito  
  #place(  
    top + left,  
    block(  
      width: 100%,  
      height: 100%,  
      fill: rgb("#1a3a5c"),  
    ),  
  )  
  
  // Logo / titolo  
  #place(  
    center + horizon,  
    block(width: 88%)[  
      #align(center)[  
        #v(1.2fr)  
      ]  
    ]  
  
    // Titolo principale  
    #text(  
      fill: white,  
      size: 56pt,
```

```
weight: "bold",
  font: ("Libertinus Serif"),
)[Titolo libro]

#v(0.4em)

// Sottotitolo
#text(fill: rgb("#aed6f1"), size: 22pt,
  font: ("Libertinus Serif"),[
  Sottotitolo del libro
]
#v(3em)

// Autore del libro
#text(fill: rgb("#d8e2f2"), size: 13pt,
  font: ("Libertinus Serif"),[
#text(fill: white)[Autore del libro]
]

#v(1.8em)

// Descrizione
#text(fill: rgb("#d6eaf8"), size: 12.5pt, style: "italic")[]

#v(2.5em)

#h(0.5em)

#h(0.5em)

#h(0.5em)

#h(0.5em)

#h(0.5em)

#h(0.5em)

#h(0.5em)

#v(1fr)
]
],
```

)
]

24. Layout a griglia con grid()

`#grid()` è la funzione più versatile per il layout strutturato. Ti permette di disporre elementi in righe e colonne con controllo preciso.

24.1. Sintassi di base

```
#grid(  
  columns: (...), // Definizione colonne  
  rows: (...), // Definizione righe (opzionale)  
  gutter: ..., // Spazio tra celle  
  align: ..., // Allineamento  
  [Cella 1], [Cella 2], [Cella 3],  
  [Cella 4], [Cella 5], [Cella 6],  
)
```

24.2. Definire le colonne

Identico a `table`:

```
columns: 3 // 3 colonne uguali  
columns: (1fr, 2fr, 1fr) // Proporzioni  
columns: (auto, 1fr) // Prima adattata, seconda riempie  
columns: (3cm, 4cm, 5cm) // Misure fisse
```

24.3. Esempi pratici di layout

24.3.1. Layout copertina (icona + testo)

Il codice:

```
#grid(
  columns: (auto, 1fr),
  gutter: 1em,
  align: (center + horizon, left),
  block(
    fill: blue, width: 3cm, height: 3cm,
    radius: 8pt,
  ),
  block[
    #text(size: 18pt, weight: "bold")[Mio Documento]
    #v(0.3em)
    #text(fill: luma(100))[Autore: Mario Rossi]
    #v(0.3em)
    #text(fill: luma(100))[Data: Giugno 2026]
  ],
)
```

Produce:



Mio Documento

Autore: Mario Rossi

Data: Giugno 2026

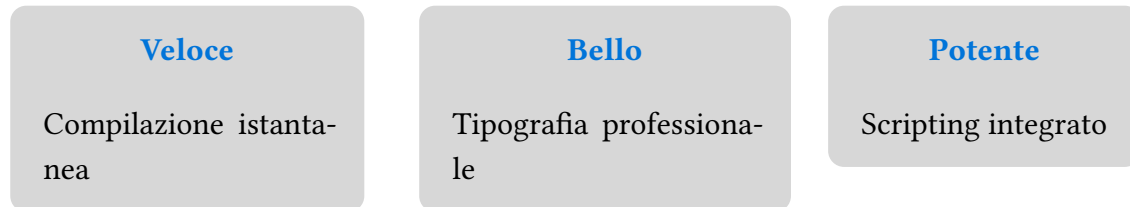
24.3.2. Layout a tre colonne informative

Il codice:

```
#let info-box(titolo, corpo) = block(
  fill: luma(215), inset: 1em, radius: 6pt,
  stroke: 1pt + luma(215),
)[
  #align(center)[
    #text(weight: "bold", fill: blue)[#titolo]
  ]
  #v(0.5em)
  #corpo
]
#grid(
  columns: (1fr, 1fr, 1fr),
  gutter: 1em,
```

```
info-box("Veloce")[Compilazione istantanea],  
info-box("Bello")[Tipografia professionale],  
info-box("Potente")[Scripting integrato],  
)
```

Produce:



24.4. La funzione `layout()`

Per layout adattativi che dipendono dalla dimensione disponibile:

```
#layout(size => {  
  // 'size' contiene la larghezza e altezza disponibili  
  let cols = if size.width > 400pt { 3 } else { 1 }  
  grid(columns: cols, ...)  
})
```

25. File multipli e moduli

Quando il documento diventa lungo o vuoi riusare stili in più documenti, conviene organizzarlo su più file.

25.1. `#include` — includere contenuto

`#include "file.typ"` inserisce il **contenuto** di un altro file esattamente dove si trova l'istruzione. È come incollare il contenuto del file nel punto corrente:

```
// Struttura cartelle:  
documento-principale.typ  
capitoli/  
  cap01-introduzione.typ  
  cap02-metodi.typ  
  cap03-risultati.typ  
stili/  
  configurazione.typ
```

```
// documento-principale.typ  
#include "stili/configurazione.typ"  
= Il Mio Documento  
#include "capitoli/cap01-introduzione.typ"  
#include "capitoli/cap02-metodi.typ"  
#include "capitoli/cap03-risultati.typ"
```

25.1.1. Vantaggi di `#include`

- I file sono separati → più facile trovare le cose
- Puoi lavorare su un capitolo alla volta
- I collaboratori lavorano su file diversi → meno conflitti
- Puoi riordinare i capitoli semplicemente spostando le righe

25.2. `#import` — importare funzioni e variabili

`#import` permette di **importare** definizioni (funzioni, variabili, costanti) da un altro file, senza includerne il contenuto testuale:

```
// Nel file 'stili/macro.typ':
#let nota(body) = block(fill: blue.lighten(80%), ...)[#body]
#let avviso(body) = block(fill: red.lighten(80%), ...)[#body]
#let colore-principale = rgb("#1a3a5c")
```

```
// Nel file principale:
#import "stili/macro.typ": nota, avviso, colore-principale
// Ora puoi usare:
#nota[Questa è una nota.]
#avviso[Questo è un avviso.]
```

25.2.1. Importare tutto con `*`

```
#import "stili/macro.typ": *
// Importa tutte le definizioni pubbliche
```

25.2.2. Importare con alias

```
#import "stili/macro.typ": nota as n, avviso as warn
// Usa #n[] invece di #nota[], e #warn[] invece di #avviso[]
```

25.2.3. Differenza tra import e include

| | <code>#include</code> | <code>#import</code> |
|----------------|----------------------------------|---|
| Cosa fa | Inserisce il contenuto del file | Importa definizioni (funzioni, variabili) |
| Produce testo? | Sì | No |
| Uso tipico | Capitoli, sezioni | Librerie di stile, macro |
| Accede a | Tutto il file viene renderizzato | Solo ciò che esporti |

25.3. Organizzare un progetto grande

Ecco una struttura tipica per una tesi o un libro:

```
tesi/  
├─ main.typ ← File principale  
├─ config/  
│   ├─ pagina.typ ← set page(), set text(), ecc.  
│   ├─ stili.typ ← show rules, heading styles  
│   └─ macro.typ ← Funzioni personalizzate  
├─ capitoli/  
│   ├─ 00-frontespizio.typ  
│   ├─ 01-introduzione.typ  
│   ├─ 02-teoria.typ  
│   ├─ 03-metodi.typ  
│   ├─ 04-risultati.typ  
│   ├─ 05-conclusioni.typ  
│   └─ appendici.typ  
├─ immagini/  
│   ├─ logo.svg  
│   └─ grafici/  
│       ├─ fig01.png  
│       └─ fig02.png  
└─ bib/  
    └─ riferimenti.bib
```

```
// main.typ  
#import "config/macro.typ": *  
#include "config/pagina.typ"  
#include "config/stili.typ"  
#include "capitoli/00-frontespizio.typ"  
#outline(depth: 3)  
#counter(page).update(1)  
#include "capitoli/01-introduzione.typ"  
#include "capitoli/02-teoria.typ"  
// ...  
#bibliography("bib/riferimenti.bib")
```

26. Pacchetti

L'ecosistema di Typst cresce ogni giorno grazie ai pacchetti della comunità. I pacchetti aggiungono funzionalità non presenti nella libreria standard.

26.1. Dove trovare i pacchetti

Il repository ufficiale è **Typst Universe**: <https://typst.app/universe> Puoi cercare per categoria: matematica, presentazioni, CV, tabelle, diagrammi, grafica, citazioni...

26.2. Come importare un pacchetto

```
#import "@preview/nome-pacchetto:versione": funzione1, funzione2
```

Il prefisso `@preview/` indica i pacchetti della comunità.

Nell'editor online di `typst.app`, i pacchetti vengono scaricati automaticamente. Se usi Typst localmente, vengono scaricati automaticamente la prima volta che li usi e poi messi in cache.

26.3. Pacchetti principali

26.3.1. showybox — Box stilizzate

```
#import "@preview/showybox:2.0.4": showybox
#showybox(
  title: "Teorema di Pitagora",
  frame: (title-color: blue, border-color: blue),
)[
  In un triangolo rettangolo, il quadrato dell'ipotenusa è
  uguale alla somma dei quadrati dei cateti:
```

```
$ a^2 + b^2 = c^2 $  
]
```

26.3.2. codly — Codice con numeri di riga

```
#import "@preview/codly:1.3.0": *  
#show: codly-init.with()  
// Ora i blocchi di codice hanno numeri di riga e stile avanzato
```

26.3.3. cetz — Disegno vettoriale

```
#import "@preview/cetz:0.5.2": canvas, draw  
#canvas({  
  import draw: *  
  circle((0, 0), radius: 1)  
  line((0, 0), (1, 0))  
  content((0, -1.5), [Cerchio])  
})
```

26.3.4. fletcher — Diagrammi e frecce

```
#import "@preview/fletcher:0.5.8" as fletcher: diagram, node, edge  
#diagram(  
  node((0,0), [Stato A]),  
  edge("->", [trasformazione]),  
  node((2,0), [Stato B]),  
)
```

26.3.5. physica — Notazione fisica

```
#import "@preview/physica:0.9.8": *  
$dv(f, x)$ // df/dx  
$pdv(f, x, y)$ // Derivata parziale  
$laplacian(f)$ // Laplaciano  
$grad f$ // Gradiente  
$div bold(F)$ // Divergenza
```

26.3.6. polylux — Presentazioni

```
#import "@preview/polylux:0.4.0": *
#set page(paper: "presentation-16-9")
#set text(size: 20pt)
#polylux-slide[
  = Titolo della Slide
  Contenuto della prima slide.
]
#polylux-slide[
  == Seconda Slide
  #pause // Pausa per le animazioni
  Questo appare dopo la pausa.
]
```

26.3.7. tablex — Tabelle avanzate

```
#import "@preview/tablex:0.0.9": tablex, cellx
#tablex(
  columns: 3,
  // Celle con colspan/rowspan più semplici
  cellx(colspan: 2)[Unità],
  [Normale],
)
```

26.3.8. lovelace — Pseudocodice e algoritmi

```
#import "@preview/lovelace:0.3.1": *
#pseudocode-list[
  *Input:* Lista  $A$  di  $n$  elementi
  *Output:*  $A$  ordinata
  + *for*  $i$  *from*  $1$  *to*  $n-1$  *do*
  + *for*  $j$  *from*  $0$  *to*  $n-i-1$  *do*
  + *if*  $A[j] > A[j+1]$  *then*
  + scambia  $A[j]$  e  $A[j+1]$ 
]
```

26.3.9. diagraph — Grafi con GraphViz

```
#import "@preview/diagraph:0.3.7": raw-render
#raw-render(
  digraph {
    A -> B
    B -> C
    A -> C
  })
```

26.4. Come scegliere la versione giusta

Su typst.app/universe ogni pacchetto mostra le versioni disponibili. In generale usa **l'ultima versione stabile** (il numero più alto). Se aggiorni un pacchetto, controlla se ci sono breaking changes nella documentazione.

```
// Sempre specificare la versione esatta:
#import "@preview/showybox:2.0.1": showybox
// Non usare wildcards: "@preview/showybox:*" non funziona
```