Pacchetti del gruppo 'graphics'

D. P. Carlisle

13/01/1999

Indice

1 Introduzione

Questo documento serve come manuale utente per i pacchetti color, graphics e graphicx. Si può ottenere ulteriore documentazione compilando i file sorgenti (`dtx`) di ciascun pacchetto (la documentazione risultante sarà in inglese).

Nota alla traduzione italiana

Una copia di questo documento e altre traduzioni in italiano di manuali LaTeX sono reperibili presso:

- `http://guild.prato.linux.it`
- `ftp://lorien.prato.linux.it/pub/guild`

e su ogni sito CTAN nella directory `/info/italian`. Per un elenco aggiornato dei siti mirror CTAN consultate `http://www.ctan.org/` alla voce "CTAN mirror".

2 Supporto del *driver*

Tutti questi pacchetti fanno uso di funzionalità che non sono fornite dal TeX stesso, ma dal *driver* usato per stampare il file `dvi`. Sfortunatamente non tutti i *driver* supportano le stesse funzionalità, e anche il metodo interno di accesso a queste estensioni varia tra i *driver*. Come conseguenza tutti questi pacchetti hanno delle opzioni, come ad esempio '`dvips`', per specificare quale *driver* deve essere usato.

In genere è bene specificare un'opzione di *default*, che indichi il *driver* che normalmente viene usato. Supponendo che per il pacchetto color, normalmente, si vogliano usare i comandi speciali definiti nel *driver* PostScript `dvipsone`, allora si deve creare un file `color.cfg` che contiene la linea:
`\ExecuteOptions{dvipsone}`
Di solito si crea anche un file `graphics.cfg`, per impostare un'opzione analoga per il pacchetto graphics.

Nei pacchetti sono dichiarate le seguenti opzioni per i *driver*:

> `dvips`, `xdvi`, `dvipdf`, `dvipdfm`, `pdftex`, `dvipsone`, `dviwindo`, `emtex`, `dviwin`, `pctexps`, `pctexwin`, `pctexhp`, `pctex32`, `truetex`, `tcidvi`, `vtex`, `oztex`, `textures`

Se si vuole usare un *driver* che non è presente nella lista precedente, si può aggiungere un'opzione per quel *driver* inserendo una riga di tipo `\DeclareOption` adatta in `graphics.cfg` e `color.cfg`, prima di renderla l'opzione predefinita tramite `\ExecuteOptions` come descritto in precedenza.

Per esempio, per aggiungere l'opzione '`dvi2ps`' per il *driver* originario Unix di traduzione da `dvi` a `ps`, e per renderla l'opzione standard, bisogna che i file di configurazione contengano:

```
\DeclareOption{dvi2ps}{\def\Gin@driver{dvi2ps.def}}
\ExecuteOptions{dvi2ps}
```

C'è un apposito file `dvi2ps.def` nella distribuzione standard. Non è abilitato normalmente, dal momento che non è stato controllato a fondo, poiché il *driver* non è più disponibile. Analogamente, i seguenti file per *driver* sono distribuiti, ma normalmente disabilitati.

`dvi2ps`, `dvialw`, `dvilaser`, `dvitops`, `psprint`, `pubps`, `ln`

La maggior parte di questi file per *driver* sono generati dal file sorgente `drivers.dtx`. Questo file contiene i sorgenti per altre versioni dei *driver* (per esempio vecchie versioni di dvips e textures) che non sono normalmente generati.

Se si usa un *driver* che non è coperto da nessuna di queste possibilità, si può provare a scrivere un file `.def` per analogia con uno dei *driver* esistenti e poi specificare un'opportuna opzione in `graphics.cfg` e `color.cfg`, come per l'esempio precedente riguardante `dvi2ps`.

3 Colore

Il supporto per il colore è basato sull'idea di un sistema di *modelli di colore*. I modelli di colore supportati da ciascun *driver* sono diversi, ma tipicamente includono:

rgb Rosso [G]Verde Blu: una lista di tre numeri, compresi tra 0 e 1, separati da virgole, che indicano le componenti del colore.

cmyk Ciano Magenta [Y]Giallo [K]Nero: una lista di quattro numeri, compresi tra 0 e 1, separati da virgole, che indicano le componenti del colore in accordo con il modello additivo usato in molte stampanti.

gray Toni di grigio: un solo numero compreso tra 0 e 1.

named Colori scelti per nome, per esempio 'JungleGreen'. Non tutti i *driver* supportano questo modello. Il nome deve essere 'noto' al *driver* o aggiunto usando i comandi descritti in `color.dtx`. Alcuni *driver* supportano una forma estesa del modello **named**, in cui si può anche specificare un'intensità del colore, cosicché 'JungleGreen, 0.5' indicherà quel colore con intensità dimezzata.

Occorre notare come il modello **named** sia in realtà solo un esempio di un modello di colore che usa nomi al posto di specifiche numeriche. Altre opzioni possono essere fornite localmente per avere differenti modelli di colore, per esempio **pantone** (un set di colori standard industriale), **x11** (nomi di colori del sistema X Window), etc. La distribuzione standard al momento non ha questi modelli, ma il modello **named** può essere usato come esempio di come definire un nuovo modello di colore. I nomi usati nel modello **named** sono quelli suggeriti da Jim Hafner nei suoi pacchetti colordvi e foiltex, e implementati in origine nell'header file `color.pro` del *driver* `dvips`.

3.1 Opzioni del pacchetto

La maggior parte delle opzioni del pacchetto color serve solamente a specificare il *driver*, ad esempio `dvips`, come discusso nel paragrafo 2.

Un'interessante opzione speciale per il pacchetto color è `monochrome.` Se si seleziona questa opzione, tutti i comandi per il colore vengono disabilitati, in modo che non generino errori, ma nemmeno colori. Questo può essere utile se si usa un sistema di visualizzazione che non gestisce i colori.

Altre tre opzioni del pacchetto controllano l'uso del modello **named**. Il *driver* `dvips` (di *default*) predefinisce 68 nomi di colori. L'opzione `dvips`, normalmente, rende questi nomi disponibili nel modello di colore **named**. Se non si vuole che questi nomi siano definiti in questo modello (risparmiando così un po' di memoria TEX) si può usare l'opzione `nodvipsnames`. Al contrario, se si usa un *driver* differente, si può voler aggiungere questi nomi al modello **named** di quel *driver* (specialmente se si sta lavorando ad un documento originariamente prodotto con `dvips`); in questo caso si può usare l'opzione `dvipsnames`. Infine l'opzione `usenames` rende tutti i nomi del modello **named** direttamente disponibili, come descritto in seguito.

3.2 Definire i colori

I colori `black` (nero), `white` (bianco), `red` (rosso), `green` (verde), `blue` (blu), `cyan` (ciano), `magenta` e `yellow` (giallo) sono predefiniti, ma si possono creare nuove definizioni di colori tramite il comando `\definecolor`.

`\definecolor{`⟨*nome*⟩`}{`⟨*modello*⟩`}{`⟨*specifica del colore*⟩`}`

Questo comando definisce ⟨*nome*⟩ come un colore che può essere usato nei successivi comandi di colore. Ad esempio:

```
\definecolor{light-blue}{rgb}{0.8,0.85,1}
\definecolor{mygrey}{gray}{0.75}
```

Ora `light-blue` e `mygrey` possono essere usati in aggiunta ai colori definiti precedentemente.

3.3 Usare i colori

3.3.1 Usare i colori predefiniti

La sintassi per cambiare colore ricalca quella del cambio di font. La sintassi base è:

`\color{`⟨*nome*⟩`}`

Questa è una *dichiarazione*, come `\bfseries`. Cambia il colore corrente in ⟨*nome*⟩ fino alla fine del gruppo o ambiente corrente.

Un metodo alternativo è quello di usare una forma di tipo *comando* che prende come *argomento* il testo da colorare. Questo è simile ai comandi dei font come `\textbf`:

`\textcolor{`⟨*nome*⟩`}{`⟨*testo*⟩`}`

Quindi il comando precedente è sostanzialmente equivalente a:
`{\color{`⟨*nome*⟩`}`*testo*`}`.

3.3.2 Usare direttamente le specifiche di colore

`\color[`⟨*modello*⟩`]{`⟨*specifica*⟩`}`
`\textcolor[`⟨*modello*⟩`]{`⟨*specifica*⟩`}{`⟨*testo*⟩`}`

Normalmente si dichiarano tutti i colori da usare successivamente in un pacchetto o nel preambolo del documento, ma a volte è conveniente usare direttamente un colore senza prima definirlo. Per rendere questo possibile, `\color` (e tutti gli altri comandi per il colore) hanno un argomento opzionale per specificare il modello. Se si usa questo argomento opzionale, allora l'argomento obbligatorio deve contenere una ⟨*specifica di colore*⟩ al posto di un ⟨*nome*⟩. Per esempio:
`\color[rgb]{1,0.2,0.3}`
selezionerà direttamente quel colore.

Questo metodo è particolarmente utile per accedere al modello **named**:
`\color[named]{BrickRed}`
seleziona il colore `dvips` BrickRed.

Piuttosto che usare ripetutamente `[named]`, si può usare `\definecolor` per creare opportuni alias:
`\definecolor{myred}{named}{WildStrawberry}` ... `\color{myred}` ...

In alternativa, se si vogliono usare i nomi originali del modello **named**, si può usare l'opzione di pacchetto `usenames`, che in realtà attiva una `\definecolor` per ciascun colore del modello **named**, rendendo così `\color{WildStrawberry}` equivalente a `\color[named]{WildStrawbery}`.

3.4 Colori con nomi

L'uso del modello di colore **named** presenta alcuni vantaggi rispetto ad usare gli altri modelli.

Per prima cosa, dal momento che il file `dvi` contiene una richiesta di colore fatta per *nome*, la miscela di colori primari che viene realmente usata per ottenere il colore richiesto può essere tarata sulle caratteristiche di una particolare stampante. Nel *driver* `dvips` il significato dei nomi dei colori è definito nel file di intestazione `color.pro`. Gli utenti sono invitati a creare versioni differenti di questo file per ogni stampante che usano. In questa maniera lo stesso file dvi produrrà colori simili anche se stampato su stampanti con differenti caratteristiche cromatiche.

In secondo luogo, oltre ai cosiddetti 'colori di processo' che sono prodotti miscelando colori primari durante la fase di stampa, si possono usare colori 'spot' o 'custom'. In questo caso un particolare nome di colore non si riferisce ad un mix di colori primari, ma ad un particolare inchiostro. Le parti di un documento che usano questo colore saranno stampate separatamente usando l'inchiostro che è rappresentato da quel nome di colore.

3.5 Colore di pagina

```
\pagecolor{⟨nome⟩}
\pagecolor[⟨modello⟩]{⟨specifica⟩}
```

Si può impostare il colore dello sfondo di un'intera pagina tramite `\pagecolor`. Questo comando usa lo stesso tipo di argomenti di `\color`, ma imposta il colore dello sfondo della pagina corrente e di tutte le successive. È una dichiarazione globale, quindi occorre usare `\pagecolor{white}` per 'ritornare alla normalità'.

3.6 Sfondi dei riquadri

Esistono due comandi simili a `\fbox` per produrre riquadri con lo sfondo colorato.

```
\colorbox{⟨nome⟩}{⟨text⟩}
\colorbox[⟨modello⟩]{⟨specifica⟩}{⟨testo⟩}
\fcolorbox{⟨nome1⟩}{⟨nome2⟩}{⟨testo⟩}
\fcolorbox[⟨modello⟩]{⟨specifica1⟩}{⟨specifica2⟩}{⟨testo⟩}
```

Il primo crea una scatola colorata con ⟨*nome*⟩, come questa. Il terzo è simile, ma mette un bordo di colore ⟨*nome1*⟩ attorno a una scatola di colore ⟨*nome2*⟩.

Questi comandi usano i parametri di `\fbox`, `\fboxrule` e `\fboxsep`, per determinare lo spessore della linea e la dimensione dell'area colorata.

3.7 Possibili problemi

TeX non è stato concepito con il colore in mente, e produrre colori richiede un grande aiuto da parte del *driver*. Perciò, a seconda del *driver*, potrebbero non essere disponibili tutte le possibilità del pacchetto color.

Alcuni *driver* non mantengono uno speciale 'stack dei colori'. Questi *driver*, con tutta probabilità, produrranno strani risultati se si annidano cambi di colore, o se si usano colori in ambienti flottanti[1].

Alcuni *driver* non mantengono i colori durante i salti di pagina, quindi se la pagina finisce in mezzo ad un capoverso colorato, l'ultima parte del testo sarà stampata, erroneamente, in nero.

[1] ambienti `figure`, `table` e similari [N.d.T.].

C'è inoltre un altro tipo di problema che si presenta con tutti i *driver*. A causa di certe difficoltà tecniche[2], è possibile che, nei punti di cambio di colore, la spaziatura sia diversa dal normale. Per questo motivo l'opzione `monochrome` non disabilita completamente i comandi di colore, ma li ridefinisce in modo che scrivano su un file di log. In questo modo si otterrà lo stesso effetto sulla spaziatura, in modo da ottenere un documento in bianco e nero che però ha la stessa spaziatura del documento finale a colori.

4 I pacchetti grafici

Ci sono due pacchetti grafici:

graphics Il pacchetto grafico 'standard'.

graphicx Il pacchetto grafico 'esteso', o 'potenziato'.

I due differiscono solo nel formato degli argomenti opzionali dei vari comandi; i nomi dei comandi e gli argomenti obbligatori sono gli stessi in tutti e due i pacchetti.

4.1 Opzioni del pacchetto

Come osservato nel paragrafo 2, i pacchetti grafici condividono il *driver* con il pacchetto color. Come per il colore, è bene impostare un *driver* predefinito nel file `graphics.cfg` che conterrà la linea (per usare `dvips`, ad esempio):
`\ExecuteOptions{dvips}`

I pacchetti grafici hanno altre opzioni per controllare quali caratteristiche abilitare:

draft sopprime tutte le funzionalità di tipo 'special'. In particolare non saranno inserite le figure (ma saranno ugualmente lette per determinarne le dimensioni); al loro posto verrà stampato un riquadro della giusta dimensione, contenente il nome del file.

final L'opposto di `draft`. Utile per annullare l'effetto di un'opzione globale `draft` specificata nel comando `\documentclass`.

hiderotate Non fa vedere il testo ruotato (presumibilmente perché il *driver* non può effettuare la rotazione).

hidescale Non fa vedere il testo ridimensionato (presumibilmente perché il *driver* non può effettuare il ridimensionamento).

hiresbb Cerca le indicazioni di dimensione nelle linee `%%HiResBoundingBox` invece che nelle linee standard `%%BoundingBox`. Nuova caratteristica 29/10/1996

[2] Almeno due le cause: 1) La presenza di uno `\special` ⟨*qualcosa*⟩ impedisce a `\addvspace` di 'vedere' lo spazio della lista verticale corrente, causando quindi un'incorretta aggiunta di spazio verticale. 2) Un ⟨*qualcosa*⟩ come primo elemento in un `\vtop` fa spostare il punto di riferimento della scatola.

4.2 Rotazione

graphics:	`\rotatebox{`⟨*angolo*⟩`}{`⟨*testo*⟩`}`
graphicx:	`\rotatebox[`⟨*lista valori chiave*⟩`]{`⟨*angolo*⟩`}{`⟨*testo*⟩`}`

Questo comando mette ⟨*testo*⟩ in una scatola, come `\mbox`, ma ruota la scatola di ⟨*angolo*⟩ gradi, come questo .

La versione standard esegue la rotazione sempre attorno al punto di riferimento della scatola, ma la versione keyval accetta le seguenti chiavi:

`origin`=⟨*etichetta*⟩
`x`=⟨*dimensioni*⟩
`y`=⟨*dimensioni*⟩
`units`=⟨*numero*⟩

Si possono, quindi, specificare `x` e `y`, che indicano le coordinate del centro di rotazione relativamente al punto di riferimento della scatola, per esempio `[x=2mm, y=5mm]`. Alternativamente, per i punti più comuni, si può usare `origin` con una ⟨*etichetta*⟩ contenente uno o più dei seguenti dati: `lrctbB`[3] (`B` indica la *baseline*, come nel pacchetto PSTricks). Per esempio, si osservi la differenza tra una rotazione standard di 180° ... Come Questa ... e l'effetto ottenuto usando la chiave `origin`:
`[origin = c]` rotazione intorno al centro della scatola,... Come Questa ...
`[origin = tr]` rotazione intorno all'angolo superiore destro Come Questa ...

La chiave `units` permette di cambiare l'unità di misura predefinita, che è gradi antiorari. Si deve dare il numero di nuove unità in una rotazione antioraria completa. Per esempio:
`[units = -360]` specifica i gradi in senso orario.
`[units = 6.283185]` specifica i radianti.

4.3 Ridimensionamento

4.3.1 Ridimensionamento di un fattore di scala

`\scalebox{`⟨*scala-o*⟩`}[`⟨*scala-v*⟩`]{`⟨*testo*⟩`}`

Analogamente al comando precedente, questo comando è simile a `\mbox`, ma ridimensiona il ⟨*testo*⟩. Se non viene specificata la ⟨*scala-v*⟩, essa viene posta di *default* allo stesso valore di ⟨*scala-o*⟩. Se viene specificata, il testo viene deformato, dal momento che gli allungamenti verticale e orizzontale sono differenti, Così.

`\reflectbox{`⟨*testo*⟩`}`

Un'abbreviazione per `\scalebox{-1}[1]{`⟨*testo*⟩`}`.

[3] l=sinistra (left), r=destra (right), c=centro, t=alto (top), b=basso [N.d.T.].

4.3.2 Ridimensionamento ad una data dimensione

`\resizebox*{⟨lunghezza-o⟩}{⟨lunghezza-v⟩}{⟨testo⟩}`

Ridimensiona ⟨*testo*⟩ in modo che la sua larghezza sia ⟨*lunghezza-o*⟩. Se viene usato `!` al posto di uno dei due argomenti, viene usato l'altro per determinare il fattore di scala da usare in entrambe le direzioni. Normalmente ⟨*lunghezza-v*⟩ si riferisce all'altezza della scatola, ma nella forma asteriscata si riferisce a 'altezza + profondità'. Come in tutti gli argomenti dei riquadri LaTeX 2_ε, si possono usare `\height`, `\width`, `\totalheight` e `\depth` (rispettivamente altezza, larghezza, altezza totale e profondità) per riferirsi alle dimensioni originarie della scatola.

`\resizebox{1in}{\height}{Qualcosa}`: Qualcosa

`\resizebox{1in}{!}{Qualcosa}`: Qualcosa

4.4 Includere file grafici

Le funzioni per l'inclusione di file grafici tentano di offrire all'utente la medesima sintassi per includere tutti i tipi di file grafici che possono essere trattati dal *driver*. Questo si basa sul fatto che il file abbia un'estensione che ne identifichi il tipo. Le 'opzioni del *driver*' definiranno un insieme di estensioni che il *driver* può trattare, anche se questa lista può essere estesa usando le dichiarazioni descritte in seguito.

Se l'estensione del file è sconosciuta per il *driver*, il sistema può provare ad usare un tipo di *default*. Il *driver* PostScript imposta questo *default* a `eps` (PostScript), ma questo comportamento può essere modificato se si vuole un altro tipo di trattamento predefinito.

graphics: `\includegraphics*[⟨llx,lly⟩][⟨urx,ury⟩]{⟨file⟩}`
graphicx: `\includegraphics*[⟨lista valori chiave⟩]{⟨file⟩}`

Questo comando include un file grafico.

Se l'asterisco è presente, l'immagine è 'tagliata' alla dimensione specificata. Se l'asterisco è omesso, allora qualsiasi parte dell'immagine che si trova al di fuori del *bounding box*[4] specificato sovrascriverà il testo circostante.

Se l'argomento opzionale viene omesso, allora la dimensione dell'immagine verrà determinata leggendo un file esterno, come descritto in seguito.

versione graphics Se `[⟨urx,ury⟩]` è presente, deve indicare le coordinate dell'angolo in alto a destra dell'immagine, come coppia di dimensioni TeX. Se vengono omesse le unità di misura, si assume che siano `bp`, quindi `[1in,1in]` e `[72,72]` sono equivalenti. Se viene usato un solo argomento opzionale, si assume che l'angolo in basso a sinistra sia a `[0,0]`; in alternativa, si può usare `[⟨llx,lly⟩]` per indicare le coordinate di questo punto.

[4] Letteralmente 'scatola delimitante', è la scatola che delimita la porzione da visualizzare dell'immagine. In realtà esso definisce più propriamente la dimensione dell'area riservata all'immagine nel documento. [N.d.T.].

versione graphicx In questo caso la forma asteriscata è presente solo per compatibilità con la versione standard, e aggiunge solamente `clip` alla lista delle chiavi specificate (in più, per incrementare la compatibilità, se si usano *due* argomenti opzionali, viene usata la versione 'standard' di `\includegraphics` anche se è caricato il pacchetto graphicx).

Sono ora elencate le possibili chiavi.

bb Gli argomenti devono essere quattro dimensioni, separate da spazi. Queste indicano il *bounding box* della regione da stampare all'interno del file[5].

bbllx,bblly,bburx,bbury Imposta il *bounding box*. Principalmente per compatibilità con i vecchi pacchetti. `bbllx=a,bblly=b,bburx=c,bbury=d` è equivalente a `bb = a b c d`.

natwidth,natheight Un'altra alternativa a `bb`. `natheight=h,natwidth=w` è equivalente a `bb = 0 0 h w`.

hiresbb Chiave a valori booleani. Se impostata a `true` (vero) (specificare solo `hiresbb` è equivalente a `hiresbb=true`) TeX cercherà linee del tipo `%%HiResBoundingBox` invece che `%%BoundingBox`. Può essere impostato a `false` (falso) per annullare l'effetto di un'impostazione di *default* a `true` determinata dall'opzione di pacchetto `hiresbb`. Nuova caratteristica 29/10/1996

viewport La chiave `viewport` accetta quattro argomenti, proprio come `bb`. In questo caso, però, i valori sono riferiti all'origine indicata dal *bounding box* nel file. In questo modo per 'vedere' il quadrato di `1in` a partire dall'angolo in basso a sinistra dell'area specificata dal *bounding box*, si può usare l'argomento `viewport=0 0 72 72`. Nuova caratteristica 01/06/1995

trim È simile a `viewport`, ma qui le quattro lunghezze indicano la quantità di spazio da levare o aggiungere a ciascun lato. `trim= 1 2 3 4` 'taglia' l'immagine di `1bp` a sinistra, `2bp` in basso, `3bp` a destra e `4bp` in alto. Nuova caratteristica 01/06/1995

angle Angolo di rotazione.

origin Origine della rotazione. Si veda la documentazione di `\rotatebox`. Nuova caratteristica 28/09/1995

width Larghezza richiesta. L'immagine viene ridimensionata a questa larghezza.

height Altezza richiesta. L'immagine viene ridimensionata a questa altezza.

totalheight Specifica l'altezza totale (altezza + profondità) della figura. Questa sarà differente dall''altezza' se vi è stata una rotazione. In particolare se la figura è stata ruotata di $-90°$, essa avrà un'altezza nulla e una notevole profondità. Nuova caratteristica 01/06/1995

keepaspectratio Chiave a valore booleano, come `clip`. Se impostata a `true` (vero), allora anche specificando entrambe `width` e `height` (o `totalheight`) l'immagine non viene distorta, ma viene ridimensionata in modo tale che non si superi nessuna delle due dimensioni. Nuova caratteristica 27/09/1995

[5] il pdfTeX 14h non gestisce correttamente i *bounding box*, quindi è consigliabile usare l'opzione viewport [N.d.T.].

scale Fattore di scala.

clip Alternativamente `true` (vero) o `false` (falso) (o nessun valore, il che è equivalente a `true`). L'immagine viene tagliata sul *bounding box*.

draft Chiave a valore booleano, come `clip`. Passa temporaneamente alla modalità *draft*.

type Specifica il tipo di immagine.

ext Specifica l'estensione del file. Va usato *solamente* insieme a `type`.

read Specifica l'estensione del 'file di lettura'. Va usato *solamente* insieme a `type`.

command Specifica un comando che deve essere applicato al file. Va usato *solamente* insieme a `type`.

Per le chiavi che specificano le dimensioni originali (ossia le chiavi per il *bounding box*, `trim`, e `viewport`) si può omettere l'unità di misura, e in questo caso si assume che essa sia `bp` (ossia punti PostScript).

Le prime sei chiavi specificano le dimensioni originali dell'immagine. Queste devono essere specificate nel caso in cui il file non possa essere letto da TeX o contenga una specifica di *bounding box* di dimensioni errate.

`bbllx`... `\bbury` sono principalmente per compatibilità con i vecchi pacchetti.
`bbllx=a, bblly=b, bburx=c, bbury=d`
è equivalente a
`bb = a b c d`.

`natheight` e `natwidth` sono solo delle abbreviazioni per impostare la coordinata in basso a sinistra a `0 0`, e quella in alto a destra alla larghezza e altezza specificate.

Le rimanenti chiavi servono a specificare un ridimensionamento o una rotazione che deve essere applicata all'immagine. Per ottenere questi effetti usando il pacchetto graphics, la chiamata a `\includegraphics` deve essere posta all'interno dell'argomento di un comando `\rotatebox` o `\scalebox`.

Le chiavi vengono lette da sinistra a destra, quindi `[angle=90, height=1in]` significa ruotare di 90 gradi e dopo ridimensionare l'altezza a `1in`.
`[height=1in, angle=90]` avrebbe portato ad una *larghezza* finale di `1in`.

Se viene caricato anche il pacchetto calc, le lunghezze possono usare la sintassi di calc, per esempio per specificare una larghezza di 2 cm inferiore alla larghezza del testo: `[width=\textwidth-2cm]`.

TeX lascia lo spazio equivalente al *bounding box* che viene specificato nel file, o negli argomenti opzionali. Se vi è una parte dell'immagine che eccede quest'area, essa di *default* sovrascriverà il testo circostante. Se viene usata la forma asteriscata, o viene specificata l'opzione `clip`, qualsiasi parte dell'immagine al di fuori di quest'area non verrà stampata.

Le ultime quattro chiavi sopprimono la lettura dell'estensione del nome del file. Se vengono usate, l'argomento principale ⟨*file*⟩ non deve contenere l'estensione. Esse corrispondono agli argomenti di `\DeclareGraphicsRule`, descritto più avanti.

Per vedere l'effetto che hanno le varie opzioni, si consideri il file `a.ps`. Questo file contiene la specifica di *bounding box*

```
%%BoundingBox:100 100 172 172
```

Ossia, la regione stampata è formata da un quadrato di dimensione 1 pollice a 100 pt dall'angolo in basso a sinistra del foglio.

In tutti gli esempi che seguono l'input sarà del tipo

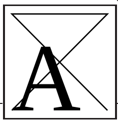
```
sinistra---\fbox{\includegraphics{a}}---destra
```

con differenti opzioni passate a `\includegraphics`.

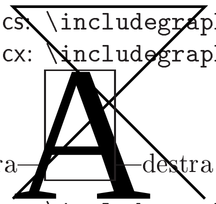
Nessun argomento opzionale.

sinistra—  —destra

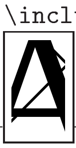
graphics: `\scalebox{0.5}{\includegraphics{a}}`
graphicx: `\includegraphics[scale=.5]{a}`

sinistra—  —destra

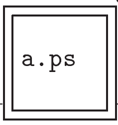
graphics: `\includegraphics[115,110][135,145]{a}}`
graphicx: `\includegraphics[bb= 115 110 135 145]{a}`

sinistra—  —destra

graphics: `\includegraphics*[115,110][135,145]{a}}`
graphicx: `\includegraphics[bb= 115 110 135 145,clip]{a}`

sinistra—  —destra

graphics: `\scalebox{0.5}{\includegraphics{a}}` e l'opzione `draft`.
graphicx: `\includegraphics[scale=.5, draft]{a}`

sinistra— 

—destra

4.5 Altri comandi nei pacchetti graphics e graphicx

`\graphicspath{⟨dir-list⟩}`

Questa dichiarazione opzionale può essere usata per specificare una lista di directory in cui cercare i file grafici. Il formato è lo stesso della primitiva LaTeX 2ε `\input@path`: una lista di directory, ciascuna racchiusa in un gruppo `{}` (anche se nella lista ce n'è una sola). Per esempio:
`\graphicspath{{eps/}{tiff/}}`
farà in modo che il sistema cerchi nelle sottodirectory `eps` e `tiff` della directory corrente. Questa è la sintassi Unix, su un Mac sarebbe stato:
`\graphicspath{{:eps:}{:tiff:}}`
Si noti la differenza di convenzioni, sui Macintosh è necessario un `:` per indicare la cartella corrente, mentre su Unix un `/` iniziale avrebbe significato la directory radice 'root'.

L'impostazione predefinita per questo percorso è `\input@path`, cioè: i file grafici saranno cercati in tutti i posti in cui vengono cercati i file TeX.

`\DeclareGraphicsExtensions{⟨ext-list⟩}`

Questa dichiarazione specifica il comportamento del sistema quando non viene specificata un'estensione tra gli argomenti di `\includegraphics`. ⟨*ext-list*⟩ deve essere una lista di estensioni separate da virgole (spazi bianchi tra gli elementi vengono ignorati). Viene creato un nome di file aggiungendo la prima estensione presente nella lista; se il file viene trovato, il sistema agisce come se fosse stata specificata quell'estensione, altrimenti viene provata la successiva estensione nella ⟨*ext-list*⟩.

Nuova descrizione 01/12/1994

Si noti che se non viene indicata un'estensione nel comando `\includegraphics`, il file grafico deve esistere al momento della compilazione LaTeX, in quanto il nome completo del file viene usato per determinare quale estensione della lista deve essere usata. D'altro canto, se viene specificata l'estensione, per esempio `\includegraphics{a.ps}` invece di `\includegraphics{a}`, allora non è necessario che il file esista quando viene effettuata la compilazione (in particolare può essere creato 'al volo' dal ⟨*comando*⟩ specificato in `\DeclareGraphicsRule`, spiegato più avanti). LaTeX deve, però, essere in grado di determinare le dimensioni dell'immagine, quindi le dimensioni devono essere specificate negli argomenti o nel 'file di lettura', che deve essere presente al momento della compilazione LaTeX.

`\DeclareGraphicsRule{⟨ext⟩}{⟨type⟩}{⟨file-di-lettura⟩}{⟨comando⟩}`

Si possono fare un numero qualsiasi di dichiarazioni di questo tipo. Esse determinano il modo in cui il sistema reagisce quando viene specificata un'estensione ⟨*ext*⟩. L'estensione può essere specificata esplicitamente o, se l'argomento di `\includegraphics` non ha un'estensione, può essere un'estensione predefinita dalla lista ⟨*ext-list*⟩ specificata con `\DeclareGraphicsExtensions`.

⟨*ext*⟩ è l'estensione dei file a cui si applica questa regola. Come caso speciale, ⟨*ext*⟩ può essere `*`, per indicare il comportamento predefinito per tutte le estensioni non dichiarate (si veda l'esempio più avanti).

⟨*type*⟩ è il 'tipo' del file in questione. Tutti i file dello stesso tipo saranno letti dallo stesso comando interno (che deve essere definito in un 'driver file'). Per esempio, i file con estensione `ps`, `eps` e `ps.gz` possono essere tutti classificati come di tipo `eps`.

⟨*file-di-lettura*⟩ determina l'estensione del file che deve essere letto per determinare le informazioni sulla dimensione dell'immagine. Può essere lo stesso di ⟨*ext*⟩, ma può essere differente. Per esempio, i file `.ps.gz` non sono facilmente leggibili da TeX, così si possono mettere le informazioni riguardanti il *bounding box* in un file separato con estensione `.ps.bb`. Se ⟨*file-di-lettura*⟩ non è indicato, `{}`, allora il sistema non cercherà di trovare un file esterno contenente le informazioni di dimensione, e queste dovranno essere specificate negli argomenti di `\includegraphics`. Se il *driver* specifica una procedura per leggere le dimensioni di file di tipo ⟨*type*⟩, verrà usata quella, altrimenti verrà usata la procedura per leggere i file `eps`. In questo modo la dimensione di un file di tipo *bitmap* può essere specificata in un file contenente una linea `%%BoundingBox` come se fosse un PostScript, se non è disponibile un formato più specifico.

Come caso speciale, si può usare `*` per indicare la stessa estensione del file grafico. Questo è utile principalmente nel caso in cui si usi `*` come estensione, dal momento che in questo caso l'estensione è ignota. Per esempio

```
\DeclareGraphicsRule{*}{eps}{*}{}
```

Questo dichiara una regola di *default* tale che tutte le estensioni ignote sono trattate come file EPS, e il file sia letto alla ricerca di un commento di tipo `%%BoundingBox`.

⟨*comando*⟩ è di solito vuoto, ma se non lo è sarà usato al posto del nome del file nello `\special`. In questo argomento si può usare `#1` per indicare il nome del file. In questo modo, se si usa il *driver* `dvips`, si può usare
`\DeclareGraphicsRule{.ps.gz}{eps}{.ps.bb}{`zcat #1}`
l'argomento finale fa in modo che dvips usi il comando `zcat` per decomprimere il file prima di inserirlo nell'output PostScript.

Si noti che LaTeX cercherà i file grafici per mezzo di una ricerca nelle directory definite in TEXINPUTS (ed eventualmente in altri posti, come specificato con `\graphicspath`), ma può darsi che il comando che si specifica in quest'argomento non sia in grado di trovare i file a meno che essi non siano nella directory corrente. Su alcuni sistemi è possibile modificare il comando in modo che possa cercare i file in tutte le directory in cui vengono cercati da LaTeX. Per esempio su Unix, con le più recenti versioni di TeX web2c, si può modificare il comando precedente in modo che l'ultimo argomento sia:
``{`zcat `kpsewhich -n latex tex #1`}``
che fa in modo che il programma `kpsewhich` trovi il file, cercandolo all'interno dei percorsi LaTeX, e poi passi il percorso completo al programma `zcat` in modo che possa decomprimere il file. Qualsiasi costrutto di questo tipo è strettamente dipendente dal sistema, e sarebbe meglio metterlo in un file `graphics.cfg`, mantenendo così portabile il documento.

4.6 Impostazione globale delle chiavi

La maggior parte delle chiavi di tipo keyval usate nel pacchetto graphicx possono essere impostate anche attraverso il comando `\setkeys` fornito dal pacchetto keyval.

Per esempio, si supponga di volere che tutte le figure da inserire nel documento corrente siano ridimensionate al 75% della larghezza delle linee del testo, in questo caso si può dare il seguente comando:
`\setkeys{Gin}{width=0.75\textwidth}`
'`Gin`' è il nome usato per le chiavi keyval associate alla 'Inclusione Grafica'. Tutti i successivi comandi `\includegraphics` (all'interno dello stesso gruppo o ambiente) agiranno come se fosse specificato `[width=0.75\textwidth]` in aggiunta alle altre chiavi impostate negli argomenti opzionali.

In maniera simile, per fare in modo che tutti gli argomenti di `\rotatebox` accettino un argomento in radianti, occorre specificare:
`\setkeys{Grot}{units=6.28318}`

4.7 Compatibilità tra graphics e graphicx

Per un autore di documenti non ci sono effettivi problemi di compatibilità tra i due pacchetti. Si può scegliere l'interfaccia che si preferisce e usare il pacchetto appropriato.

Per un autore di classi o pacchetti la situazione è leggermente differente. Si supponga di dover scrivere una classe per lettere che deve stampare un logo della compagnia come parte dell'intestazione.

Come autore della classe si può voler dare all'utente la possibilità di usare entrambe le interfacce all'interno delle lettere (nel caso in cui debbano includere ulteriori immagini nel corpo della lettera). In questo caso la classe deve caricare il pacchetto graphics, e non il graphicx, in quanto altrimenti si obbligherebbe ogni utente della classe ad usare l'interfaccia di keyval. Il logo deve essere incluso tramite `\includegraphics` *senza* argomenti opzionali (se la dimensione corretta è nel file) o con *tutti e due* gli argomenti opzionali. Non si deve usare la forma con *un solo* argomento opzionale dal momento che il significato di quest'argomento cambierebbe (e genererebbe errori) se l'utente caricasse il pacchetto graphicx insieme a questa classe di esempio.

5 Gli altri pacchetti del gruppo 'graphics'

5.1 epsfig

Questo è un piccolo pacchetto ed è essenzialmente una serie di ridefinizioni costruite sul pacchetto graphicx, in modo da definire un comando `\psfig` che ha sintassi
`\psfig{file=xxx,...}` invece di `\includegraphics[...]{xxx}`.
Ha anche altri comandi per renderlo leggermente più compatibile con il vecchio stile LaTeX 2.09 che aveva lo stesso nome.

5.2 trig

Il pacchetto trig non è concepito per essere usato direttamente nei documenti. Calcola le funzioni trigonometriche seno, coseno e tangente. Queste sono usate per calcolare lo spazio usato dai riquadri che hanno subito una rotazione. Questo pacchetto è usato anche dal programma `fontinst`, che converte i file PostScript in una forma usabile da TEX.

Oltre ad essere usate come un pacchetto LATEX, le macro possono essere estratte tramite docstrip con le opzioni `plain,package`. In questo caso le macro possono essere usate direttamente come parte di un altro file di macro (funzionano con qualsiasi formato basato sul plain TEX[6]).

5.3 keyval

Il pacchetto keyval è concepito per essere usato da altri pacchetti. Fornisce un metodo generico per impostare 'chiavi' come quelle usate dal pacchetto graphicx, e per dividere le liste, separate da virgole, di coppie ⟨*chiave*⟩ = ⟨*valore*⟩.

Come il pacchetto trig, queste macro possono essere estratte e usate come parte di un'altro file di macro, basato sul plain TEX, oppure essere usate come normale pacchetto LATEX.

Di *default* una chiave non dichiarata genera un errore, ma se viene usata l'opzione `unknownkeysallowed`, allora le chiavi sconosciute verranno silenziosamente ignorate (lasciando un messaggio nel file `log`). Questa opzione è accettata anche dal pacchetto graphicx.

5.4 pstcol

PSTricks, di Timothy Van Zandt, è un pacchetto di immensa potenza che implementa un'interfaccia veramente molto completa tra PostScript e TEX. Sfortunatamente, il supporto per il colore di PSTricks è lievemente incompatibile con il meccanismo dei colori definito nel pacchetto color. Il pacchetto pstcol è un pacchetto (temporaneo, si spera) che modifica un numero molto piccolo di funzioni interne a PSTricks, in modo da rimuovere questa incompatibilità. Se PSTricks viene caricato tramite questo pacchetto, si possono usare tutti i colori definiti dai comandi del pacchetto color all'interno dei comandi PSTricks e viceversa.

5.5 lscape

Il pacchetto lscape richiede e accetta le stesse opzioni del pacchetto graphics. Definisce un ambiente `landscape` all'interno del quale il corpo della pagina è ruotato di 90 gradi. Le intestazioni e i piè di pagina rimangono invariati, mantenendo il loro normale posizionamento.

[6] Con "plain TEX" si intende il TEX senza macro aggiuntive. LATEX, ConTEXt, PDFTEX, etc. *non* sono plain TEX. [N.d.T.].