

longmedal: un pacchetto per medaglioni divisi su più pagine

Agostino De Marco, Massimiliano Dominici

Sommario

Alcuni manuali usano riquadri incorniciati, eventualmente estesi su più pagine, per separare dal testo principale materiale a carattere avanzato o di altro tipo. Il pacchetto `longmedal` si propone di fornire un metodo semplice per riprodurre questo tipo di oggetti in un documento prodotto con $\text{\LaTeX}$.

Abstract

Some textbooks organize different kinds of advanced or secondary material in framed boxes, possibly spanning several pages. The `longmedal` package aims at providing an easy interface to reproduce such objects in a $\text{\LaTeX}$ document.

1 Introduzione

In manuali di argomento tecnico, ma non solo, accade spesso che una parte del testo, considerata non appartenente al testo principale, venga separata da questo anche visivamente, oltre che strutturalmente. Si può trattare di approfondimenti, di note storiche, di materiale a carattere esercitativo. Questo materiale viene di solito evidenziato da un riquadro munito di un'intestazione che ne identifica il contenuto, e può eventualmente estendersi per più di una pagina. Inoltre, in alcuni casi il materiale contenuto nel riquadro è separato anche dal flusso normale del testo e confinato in una precisa collocazione della pagina (di norma nella parte superiore). La figura 1, tratta da ANDERSON (2004), illustra appunto uno di questi riquadri.

La gestione di un layout di pagina come questo può risultare complicata e macchinosa con programmi di impaginazione WYSIWYG, anche con quelli progettati per un uso professionale; e la modifica di poche righe di testo può comportare la perdita di ore di lavoro, in particolare se a seguito di essa cambia la lunghezza del testo stesso. Con $\text{\LaTeX}$ questo processo può invece essere, entro certi limiti, automatizzato, rendendo più semplice il compito di gestire un layout del genere. Sottolineare questo aspetto, che è uno dei punti di forza di $\text{\LaTeX}$, è ancora più importante in un momento come quello attuale in cui assistiamo ad una fase di transizione nel mondo di $\text{\TeX}$. Il definitivo perfezionamento di $\text{\XeTeX}$ porterà, infatti, alla possibilità di una gestione senza precedenti dei font di sistema, un settore in cui fino ad ora era innegabile il van-

taggio acquisito dalle applicazioni commerciali. A questo va aggiunto l'auspicato avvento di $\text{\LuaTeX}$. Se questa applicazione verrà resa facilmente fruibile in tempi brevi, gli utilizzatori di $\text{\LaTeX}$ avranno a disposizione uno strumento di lavoro straordinariamente più potente dell'attuale, sotto tutti gli aspetti, non ultimo quello della programmabilità.

Il pacchetto `longmedal` si propone di fornire una soluzione di facile uso al problema di riprodurre il tipo di oggetti in questione, utilizzando a questo scopo il meccanismo di gestione degli oggetti mobili (*float*) implementato in $\text{\LaTeX}$. Questa soluzione comporta alcune limitazioni, quali l'impossibilità di avere all'interno del riquadro oggetti mobili o note a piè di pagina. Queste limitazioni, tuttavia, non sembrano irragionevoli e rappresentano un male minore rispetto ai problemi che sarebbero sorti se si fosse voluto percorrere la via di una ridefinizione della routine di output. Questa alternativa, oltre a rappresentare tecnicamente un compito di rilevante difficoltà, avrebbe potenzialmente introdotto incompatibilità con altri pacchetti che, in maniera simile, vanno a ritoccare la medesima routine.

Lo strumento più adatto per la creazione e gestione di tali *float* è risultato essere il pacchetto `float`, che insieme a pacchetti come `calc`, `ifthen`, `chngpage`, `multicol`, `xkeyval` e `xcolor` è servito a facilitare notevolmente i compiti di programmazione degli autori.¹

Le “radici” del pacchetto `longmedal` si possono trovare in due discussioni di fine 2007 all'interno del Forum del GuIT: <http://www.guit.sssup.it/phpBB2/viewtopic.php?t=3186&highlight=e> e <http://www.guit.sssup.it/phpBB2/viewtopic.php?t=3316&highlight=>. Il Prof. Claudio Beccari ha fornito, poi, un preziosissimo apporto nelle fasi iniziali del progetto, con i suoi suggerimenti e con la revisione del codice, oltre ad aver ispirato il nome del pacchetto (il termine “medaglione” appare infatti in BECCARI (2007, pp. 206-207)).

2 L'algoritmo principale

L'idea fondamentale alla base dell'implementazione del pacchetto è, come si è detto, quella di sfruttare il meccanismo di posizionamento dei *float*, così co-

1. Per maggiori dettagli sui singoli pacchetti si rimanda alla relativa documentazione: ADRIAENS (2008); CARLISLE (2001); KERN (2007); LINGNAU (2001); MITTELBACH (2006); THORUP *et al.* (2005); WILSON (2003).

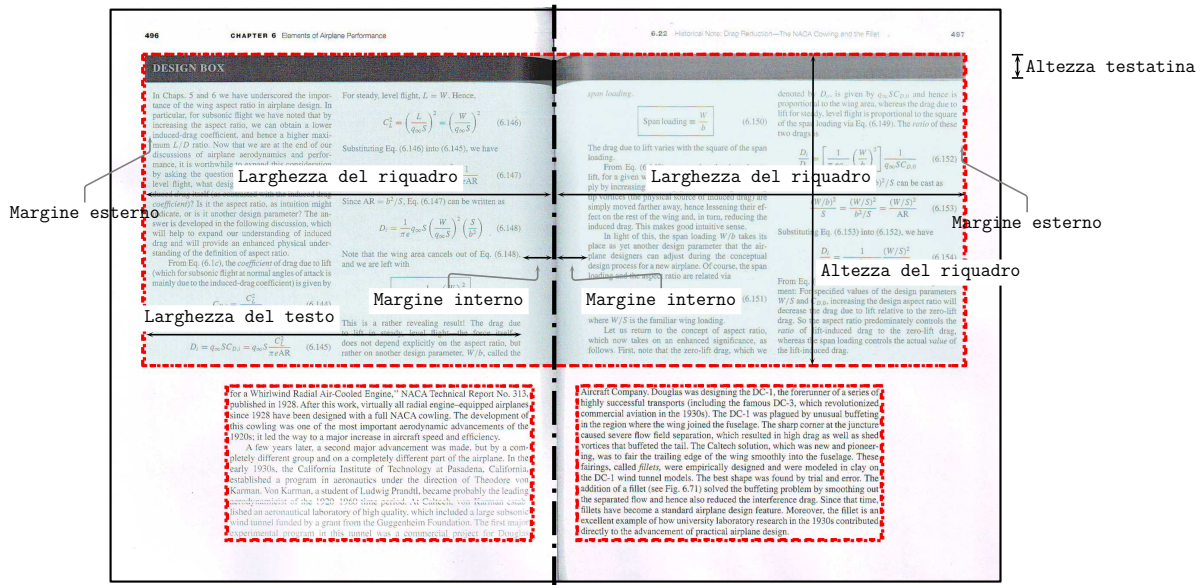


FIGURA 1: Illustrazione di un riquadro di approfondimento.

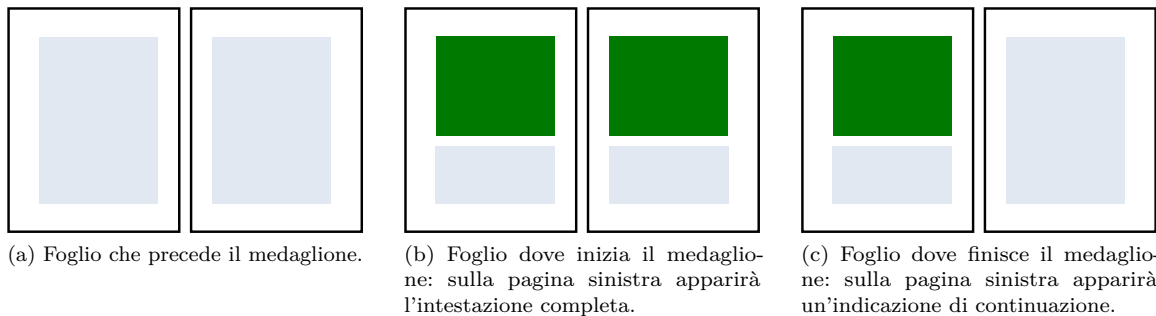


FIGURA 2: Schema di posizionamento delle singole parti di un medaglione.

me è definito in `\LTeX`. Il contenuto del medaglione viene suddiviso in un numero sufficiente di `float` dotati di adeguata intestazione e posizionato in pagine adiacenti.

Il comando che l'utente usa nel proprio documento per avere un medaglione, (`\longmedal`), non agisce direttamente sul materiale in questione, ma si limita semplicemente a inserirlo in un box (`\LMD@box`), dopo aver raccolto eventuali indicazioni dall'utente sul valore dei parametri che controllano l'aspetto del medaglione ed aver eseguito una serie di calcoli sulle varie dimensioni che lo caratterizzano. In particolare vengono determinate l'altezza e la larghezza dell'area del testo vera e propria del riquadro, e queste due dimensioni vengono immagazzinate nei registri `\LMD@height` e `\LMD@widthc`. Inoltre viene incrementato il contatore interno che regola il numero progressivo del riquadro.²

Il lavoro vero e proprio viene poi eseguito dalla macro interna `\LMD@makeboxes`. Essa prende, come unico argomento, il nome dell'ambiente mobile

2. Questa operazione va compiuta adesso, e non quando il materiale viene diviso nei singoli `float`.

in cui verrà inserito l'argomento di `\longmedal`.³

All'interno di `\LMD@makeboxes` viene eseguito un ciclo che termina soltanto quando `\LMD@box` risulta vuoto. Ad ogni iterazione del ciclo, la primitiva `\vsplit` estrae da `\LMD@box` tanto materiale quanto basta per riempire un box di dimensioni `\LMD@height × \LMD@widthc`. A questo materiale vengono aggiunti eventuali sfondi e cornici, dopo di che può essere inserito in un ambiente mobile.

Alla fine del ciclo avremo quindi ottenuto una serie di ambienti mobili, tutti dello stesso tipo, e tutti con lo stesso posizionamento (di norma nella parte superiore della pagina), pronti per essere accodati ed emessi uno per pagina.⁴ Se il documento che si sta scrivendo è un documento a doppia facciata, al momento dell'effettivo posizionamento dei singoli `float`, viene controllata la parità della pagina su

3. Come vedremo in seguito nella sezione 4.5, questo è utile per definire nuovi tipi di riquadro, ognuno corredato del proprio, esclusivo, ambiente mobile.

4. `\LTeX` mantiene code diverse per diversi tipi di oggetti mobili, per cui non c'è pericolo di interferenza, tuttavia tabelle e figure posizionate in alto possono essere bloccate da questa coda e spostate più o meno lontano dal punto di richiamo.

cui ci si trova⁵ e viene di conseguenza generata l'intestazione più pertinente: sul primo *float* apparirà comunque l'intestazione completa; a partire dal secondo, se questo si trova su una pagina pari apparirà l'indicazione di continuazione, altrimenti l'intestazione sarà vuota (la figura 2 illustra la situazione indicata). Se invece si sta scrivendo un documento a facciata singola, apparirà l'intestazione completa sul primo *float* e l'indicazione di continuazione su tutti gli altri.

Inoltre ad ogni primo *float* emesso verranno anche scritte su un file ausiliario tutte le informazioni necessarie per generare un elenco dei medaglioni.

2.1 Accorgimenti per hyperref

Alcuni accorgimenti devono essere presi affinché il pacchetto possa collaborare bene con *hyperref*. In particolare alcuni problemi sorgono in quanto *hyperref* cerca di inserire un'ancora per i suoi riferimenti, ogni volta che viene emesso un nuovo *float*, contando sul fatto che il contatore sia di volta in volta incrementato. Questo ovviamente nel nostro caso non succede, perché singoli *float* appartengono allo stesso medaglione e di conseguenza devono condividere lo stesso numero progressivo. In questo modo verrebbero definite in punti diversi ancora con lo stesso nome, confondendo il meccanismo dei riferimenti, e *hyperref* è costretto a segnalare l'errore.

Inoltre un'ulteriore ancora viene definita nel momento in cui viene incrementato il contatore `longmedalFloat`. Poiché a noi serve in realtà soltanto questa, è sufficiente inibire il comportamento di *hyperref* nei confronti dei nuovi *float* generati. Il codice che segue controlla che *hyperref* sia stato caricato e, in caso affermativo, fa in modo che esso non posizioni ancora all'inizio di ogni *float*.

```
\AtBeginDocument{%
  \ifpackageloaded{hyperref}{%
    \def\LMD@nomultipleanchors{%
      \let\float@makebox\HyOrg@float@makebox%
    }
    \let\LMD@nomultipleanchors\relax
  }
}
```

2.2 Le cornici

Le cornici vengono ottenute grazie ad una serie di comandi modificati del pacchetto *framed*. In particolare, le modifiche apportate consentono di assemblare insieme i diversi *box* che compongono il medaglione (intestazione e area del testo) e di inserire un colore di sfondo dietro l'area del testo.

5. A questo scopo viene, per il momento, impiegato il pacchetto *chngpage*. Poiché il suo stesso autore lo considera ormai obsoleto, esso verrà sostituito, nelle prossime versioni, con il suo successore *change page*.

3 Comandi per l'utente

3.1 Il comando principale

Il comando principale messo a disposizione dal pacchetto è `\longmedal`. Il comando accetta un argomento opzionale e uno obbligatorio:

```
\longmedal[⟨opzioni⟩]{⟨testo⟩}
```

L'argomento obbligatorio (*⟨testo⟩*) contiene il materiale che dovrà essere posizionato nel medaglione, mentre l'argomento opzionale contiene le istruzioni per modificare l'aspetto di default del medaglione stesso. Esamineremo in seguito (sezione 4) i dettagli relativi a queste ultime.

3.2 Elenco dei medaglioni

È possibile anche generare automaticamente un elenco dei singoli medaglioni, così come può essere fatto per figure e tabelle, e, più in generale, per tutti i *float*. Tuttavia il pacchetto non mette a disposizione, almeno per il momento, un comando apposito, ma si appoggia a quanto fornito da *float*, cioè:

```
\listof{longmedalFloat}{lom}
```

Il comando `\listof{⟨nome⟩}{⟨ext⟩}` è stato infatti progettato per trattare in maniera generale ogni tipo di elenco, senza che si debba definire ogni volta un comando apposito. Si noti però che nel primo argomento di `\listof` è stato inserito non `longmedal`, ma `longmedalFloat` che è il nome del tipo di *float* per medaglioni. Questo non è del tutto soddisfacente e potrebbe essere cambiato in futuro.⁶

4 Personalizzazione dell'ambiente

Alcuni parametri relativi all'aspetto dei medaglioni possono essere modificati dall'utente secondo le proprie preferenze o le esigenze specifiche del progetto. Si è deciso di lasciare all'utente la possibilità di configurare il valore di questi parametri, che esamineremo con maggior dettaglio in seguito, sia ad un livello globale, valido per tutti i medaglioni di uno stesso tipo,⁷ sia a livello del singolo medaglione.

In un'ottica di questo tipo, e considerando l'elevato numero di parametri modificabili dall'utente, si è ritenuto che la soluzione più semplice fosse quella di associare a ciascun parametro una variabile contrassegnata da una parola chiave. Esistono diversi pacchetti che permettono di utilizzare una struttura di questo tipo. Quello fino ad oggi più

6. Tuttavia il meccanismo più generale di `\listof` continuerà comunque a funzionare, cosicché non si avranno problemi di compatibilità.

7. Vedremo in seguito (sezione 4.5) che è possibile definire comandi per nuovi tipi di medaglione, nel caso il documento ne preveda più di uno.

TABELLA 1: Opzioni per il layout generale.

Variabile	Tipo	Valore di default
<code>height</code>	dim.	<code>0.6\textheight</code>
<code>width</code>	dim.	<code>\textwidth</code>
<code>innermargin</code>	dim.	<code>0.5cm</code>
<code>outermargin</code>	dim.	<code>0.5cm</code>
<code>framesep</code>	dim.	<code>0pt</code>
<code>framerule</code>	dim.	<code>2pt</code>
<code>frameheadrule</code>	dim.	<code>2em</code>
<code>framecolor</code>	col.	<code>black</code>
<code>framebgcolor</code>	col.	<code>gray</code>
<code>columns</code>	int.	<code>1</code>
<code>drawframe</code>	alt.	<code>none</code>

usato, e contenuto in tutte le principali distribuzioni anche nella versione di base, è `keyval`. Tuttavia, con `keyval` è difficile implementare un meccanismo a cascata che permetta di ereditare modifiche precedenti, perché a quelle variabili a cui l'utente non assegna un valore specifico, viene assegnato, se è stato definito, un valore di default iniziale, che non può essere cambiato, sovrascrivendo le precedenti modifiche.

Una soluzione, ovviamente, può essere quella di specificare nuovamente, assieme alla variabile da modificare, anche tutte le altre. Ma questa soluzione, oltre ad essere estremamente scomoda, è anche soggetta ad errori, perché, in caso di un successivo ripensamento su quelli che sono i parametri ottimali, costringe l'autore ad intervenire sulla stessa variabile in diversi luoghi del documento.

Fortunatamente, il pacchetto `xkeyval` ha introdotto un sistema per memorizzare le assegnazioni. In questo modo è possibile assegnare alle variabili valori che sono ereditati nel momento in cui vengono effettuate nuove assegnazioni.

Questa possibilità di poter gestire in maniera semplice differenti livelli di configurazione è stato il motivo determinante nella scelta di `xkeyval` nei confronti di `keyval`. Del resto, con l'ultima versione 2008 anche T_EX Live, dopo M_iKT_EX, si è dotata di un sistema per la gestione dei pacchetti, per cui diventano sempre meno rilevanti le obiezioni all'adozione di pacchetti meno diffusi.

Terminata questa introduzione alla gestione delle variabili, possiamo esaminarle nei dettagli, dopo averle distinte in tre diverse categorie:

- variabili che controllano il layout generale;
- variabili che controllano l'aspetto dell'intestazione;
- altre.

4.1 Layout generale

Le variabili che rientrano in questa categoria sono elencate nella tabella 1. Ne fanno parte, in primo luogo, i parametri che specificano le dimensioni

TABELLA 2: Opzioni per l'intestazione.

Variabile	Tipo	Valore di default
<code>headernamelfont</code>	font	<code>\bfseries</code>
<code>headernumfont</code>	font	<code>\bfseries</code>
<code>headertitlefont</code>	font	<code>\bfseries</code>
<code>headercontfont</code>	font	<code>\bfseries\footnotesize</code>
<code>headerconttext</code>	testo	(continua da pagina precedente)
<code>headername</code>	testo	<code>Insight Box</code>
<code>headernum</code>	testo	<code>\thelongmedalFloat</code>
<code>headertitle</code>	testo	
<code>label</code>	testo	
<code>caption</code>	testo	<code>\LMDtitle</code>
<code>headersep</code>	testo	<code>{\ }</code>
<code>headertitlesep</code>	testo	<code>{\ }</code>
<code>headeralignment</code>	macro	
<code>headertextcolor</code>	col.	<code>white</code>
<code>preheaderskip</code>	dim.	<code>1em</code>

dei vari elementi che compongono il medaglione. Queste dimensioni, tranne quelle che si riferiscono allo spessore della cornice, sono indicate nella figura 1. Va specificato che le variabili `height` e `width` indicano, rispettivamente, l'altezza e la larghezza *totali* del riquadro. Altezza e larghezza dell'area del testo vengono calcolate a partire da queste, sottraendo i margini e altri *spazi bianchi*, come `framesep` che indica lo spazio che separa la cornice dall'area interna del riquadro (area del testo + margini). Le dimensioni possono essere passate anche sotto forma di espressioni, dato che `longmedal` carica comunque il pacchetto `calc` per eseguire internamente i propri calcoli. La seguente assegnazione, quindi, è del tutto legittima, all'interno dell'argomento opzionale di `\longmedal`:

```
width=\textwidth+2cm+2\fbboxsep
```

Le variabili con prefisso `frame` controllano invece elementi visuali come lo spessore della cornice, il suo colore, o il colore di sfondo. Può essere specificata, come colore, ogni espressione valida per `xcolor` (che viene caricato automaticamente da `longmedal`).

Infine `columns` specifica il numero di colonne in cui va divisa l'area del testo (il pacchetto `multicol` viene caricato appositamente a questo scopo), mentre `drawframe` controlla quale parte della cornice vada tracciata, oltre a quella superiore che riporta l'intestazione. Le alternative possibili sono quattro: `none` (opzione di default, verrà tracciata solo la parte superiore), `bottom` (solo la parte inferiore), `side` (solo i lati), `all` (tutta la cornice verrà tracciata).

4.2 Intestazioni

Nella tabella 2 sono indicate le variabili relative alla composizione dell'intestazione dei vari riquadri, insieme ai loro valori di default.

TABELLA 3: Altre opzioni.

Variabile	Tipo	Valore di default
defaults	bool.	false

L'intestazione è composta dal nome dell'ambiente, da un numero progressivo e da un eventuale titolo, specifico del singolo medaglione, che l'utente può scegliere di inserire oppure no. Questi dati controllano indirettamente anche la voce nell'elenco dei medaglioni. In particolare, la variabile `caption` indica ciò che va aggiunto a numero e nome dell'ambiente (di default, l'eventuale titolo).

4.3 Altre opzioni

L'ultima opzione serve a ripristinare i valori di default impostati dal pacchetto (o al momento della definizione di un nuovo tipo di medaglioni, si veda la sezione 4.5). Si tratta, naturalmente di una variabile che può assumere solo i valori *vero* o *falso* (default).

4.4 Impostazioni globali

Per quanto sia conveniente avere la possibilità di scegliere i valori dei parametri per ogni singolo medaglione, è ancora più conveniente poter disporre di un meccanismo per impostare questi parametri globalmente per tutti i medaglioni appartenenti a un determinato tipo. Il pacchetto definisce a questo scopo il comando `\LMDSetup` con la seguente sintassi:

```
\LMDSetup{⟨comando⟩}{⟨opzioni⟩}
```

dove `⟨comando⟩` è il nome del tipo di medaglione a cui si vuole applicare la configurazione e `⟨opzioni⟩` è la solita lista di assegnazioni. Come è stato già spiegato in precedenza (sezione 4), `\LMDSetup` usa internamente il comando `\presetkeys` di `xkeyval` per memorizzare i valori assegnati, in modo che un successivo uso di `\setkeys` nell'argomento opzionale, ad esempio, di `\longmedal` non faccia sì che alle variabili non specificate venga associato il valore di default.

4.5 Nuovi tipi di medaglione

`\longmedal` è abbastanza configurabile da poter rispondere a esigenze di tipo diverso. Tuttavia è possibile che il disegno di un documento richieda la presenza contemporanea di diversi tipi di medaglioni, ad esempio degli *Approfondimenti tecnici* e delle *Note storiche*. Questi diversi medaglioni devono essere tenuti rigorosamente separati, quanto ad aspetto, numerazione, ecc. senza tener conto del fatto che di norma avranno un nome differente.

È utile, quindi avere un comando per definire nuovi ambienti:

```
\DeclareNewMedalFloat{%
  ⟨comando⟩}{⟨nome⟩}{⟨pos⟩}{⟨ext⟩}
```

I quattro argomenti indicano nell'ordine:

`⟨comando⟩` il nome del comando che verrà poi usato dall'utente nel documento (deve essere specificato *senza* la barra rovescia);

`⟨nome⟩` il nome dell'ambiente, come deve apparire nell'intestazione;

`⟨pos⟩` la posizione del medaglione (una di quelle possibili per un ambiente mobile, ovviamente `t` è la più adatta);

`⟨ext⟩` l'estensione del file ausiliario per generare l'elenco (si veda la documentazione del pacchetto `float`).

Di fatto, anche il comando `\longmedal` viene definito dal pacchetto proprio usando `\DeclareNewMedalFloat`:

```
\DeclareNewMedalFloat{%
  longmedal}{Insight Box}{t}{lom}
```

Per avere un'idea di come opera effettivamente `\DeclareNewMedalFloat` riportiamo il codice relativo:

```
\newcommand{\DeclareNewMedalFloat}[4]{%
  \@namedef{LMD@#1@name}{#2}
  \LMD@newfloat{#1Float}{#3}{#4}
  \LMD@definekeys{#1}
  \LMD@keydefaults{#1}
  \LMD@makecommand{#1}
  \@namedef{theH#1Float}{%
    \@nameuse{the#1Float}}
}
```

Come si può vedere dal codice, i nomi dei vari ambienti e comandi vengono costruiti a partire dal nome del comando (primo argomento) suggerito dall'utente. `\LMD@newfloat` combina insieme le funzioni di `\newfloat` e `\floatname`, definiti dal pacchetto `float`, dichiarando, così, un nuovo ambiente mobile e assegnandogli, allo stesso tempo, il nome che deve apparire nell'intestazione. I tre comandi che seguono associano al nuovo tipo di medaglione le variabili che ne controllano l'aspetto, assegnano i valori di default e definiscono il comando che sarà usato per generare i singoli medaglioni. Infine viene creato un doppione del contatore che regola la numerazione dei successivi medaglioni, doppione che verrà usato da `hyperref`, se caricato.

All'atto di definire un nuovo tipo di medaglione sarà bene accertarsi di aver scelto come estensione del file ausiliario (argomento `⟨ext⟩`) un suffisso che non confligga con altri già esistenti e usati dal compilatore per altri scopi: come `lof` (*list of figures*), `lot`, (*list of tables*), `lom`, o altri eventualmente definiti in precedenza dall'utente.⁸

8. Diamo per scontato che l'utente non usi, come suffisso, `aux`, `toc`, ecc.

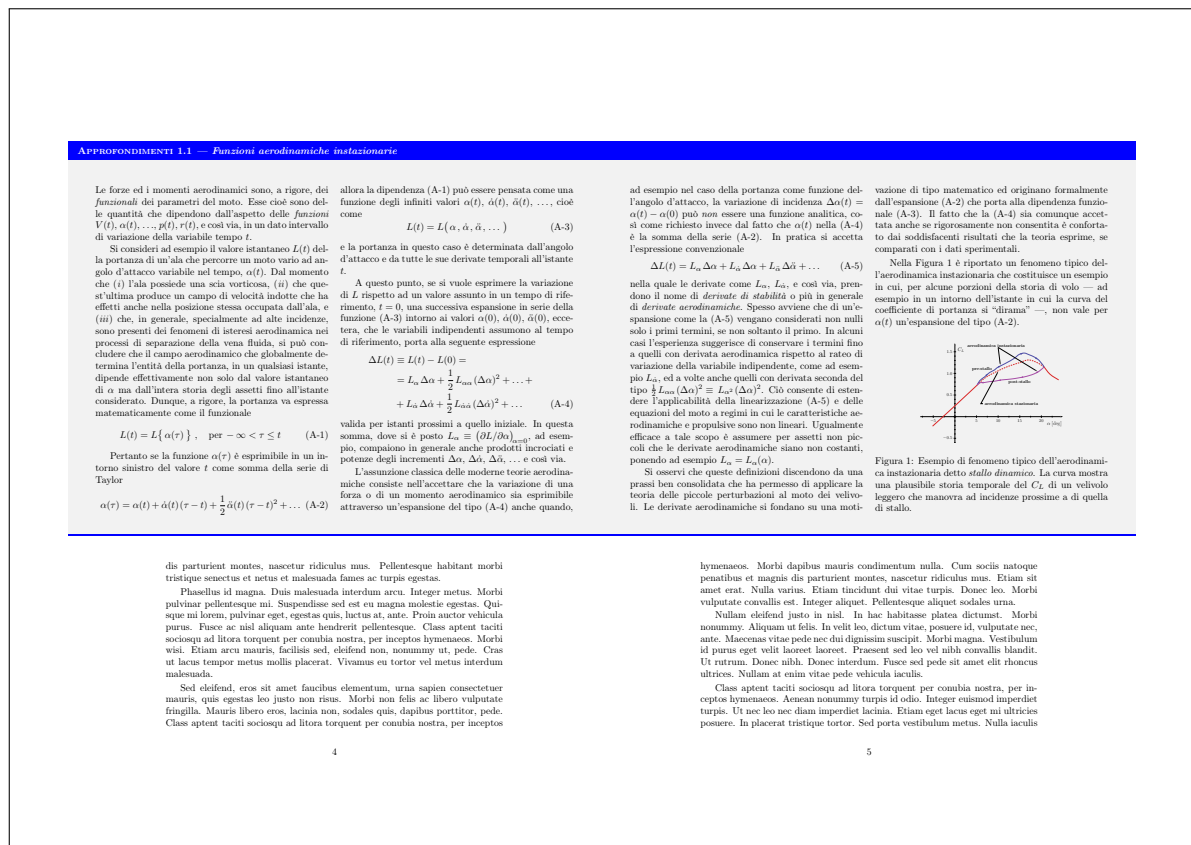


FIGURA 3: Un medaglione su due pagine, in un documento a doppia facciata.

5 Un esempio

Nella figura 3 è possibile vedere un esempio di medaglione diviso su due pagine, in un documento a doppia facciata. L'esempio è stato ottenuto con il seguente codice (per ragioni di spazio è stato omissso il contenuto del medaglione):

```
\definecolor{mylightgray}{rgb}{%
  0.95,0.95,0.95}
\columnsep 12pt
\columnseprule 0pt
\longmedal[%
  width=\textwidth+2\oddsidemargin+2in,
  innermargin=1.0cm,
  outermargin=1.0cm,
  height=0.62\textheight,
  columns=2,
  drawframe=bottom,
  framerule=2pt,
  framesep=2mm,
  framecolor=mydarkblue,
  framebgcolor=mylightgray,
  headernamefont=\bfseries\scshape,
  headertitlefont=\bfseries\itshape,
  headername=Approfondimenti,
  headertitlesep={\ ---\ },
  headertitle={Funzioni aerodinamiche
    istazionarie},
  label={\lmd:Aerodinamica:Instazionaria}
]{%

```

{testo}

L'opzione `width`, come detto in precedenza, è quella che controlla la larghezza totale dell'ambiente. Volendo che i medaglioni su due pagine affacciate si tocchino, come si vede nella figura, è necessario aggiungere, a destra e a sinistra, una misura pari allo spazio bianco che costituisce il margine interno. Nelle classi standard, questa misura è definita pari a `\oddsidemargin+1in`, il che spiega perché nel codice sopra riportato vengano mescolate diverse unità di misura (pratica da evitare ogni volta che è possibile).

6 Sviluppi futuri

Il pacchetto si trova già in uno stadio di sviluppo abbastanza avanzato e, naturalmente, una volta adeguatamente testato, potrà essere rilasciato su CTAN. Tuttavia ci sono ancora una serie di punti che meritano di essere ulteriormente approfonditi ed eventualmente adottati nelle versioni future di longmedal:

- attualmente l'altezza dei singoli elementi è fissa per un dato medaglione, mentre l'utente dovrebbe avere la possibilità di effettuare, entro certi limiti, una regolazione fine;

- attualmente il nome del medaglione non è configurato per adeguarsi automaticamente alla lingua selezionata con `babel`;
- attualmente `longmedal` si limita ad usare il comando `\listof`, del pacchetto `float` per generare l'elenco dei vari tipi di medaglione, mentre sarebbe più conveniente avere, per ogni elenco, comandi del tipo `\listoflongmedal`, più in linea con `\listoffigures`, ecc.;
- attualmente, ogni tipo di medaglione ha il proprio elenco separato, mentre l'utente potrebbe desiderare un elenco comprendente tutti i tipi di medaglione.

7 Ringraziamenti

Gli autori desiderano ringraziare in primo luogo il Prof. Claudio Beccari, al quale si devono preziosi suggerimenti nelle fasi iniziali della progettazione di questo pacchetto.

Un altro ringraziamento va a tutti coloro che hanno partecipato alle due discussioni sul Forum di GuIT menzionate nell'introduzione.

Infine è giusto menzionare qui l'anonimo recensore di Ar_sTeXnica che ha contribuito a rendere più chiari per il lettore diversi punti di questo articolo.

Riferimenti bibliografici

- ADRIAENS, H. (2008). «The xkeyval package». <http://www.ctan.org/tex-archive/macros/latex/contrib/xkeyval/doc/xkeyval.pdf>.
- ANDERSON, J. D. (2004). *Introduction to Flight*. McGraw Hill Higher Education, 5th Edition.
- BECCARI, C. (2007). «Introduzione all'arte della composizione tipografica con L^AT_EX». <http://www.guit.sssup.it/downloads/GuidaGuIT.pdf>.
- CARLISLE, D. (1999). «The keyval package». <http://www.ctan.org/tex-archive/macros/latex/required/graphics/keyval.dtx>.
- (2001). «The ifthen package». <http://www.ctan.org/tex-archive/macros/latex/base/ifthen.dtx>.
- KERN, U. (2007). «Extending L^AT_EX's color facilities. the xcolor package». <http://www.ctan.org/tex-archive/macros/latex/contrib/xcolor/xcolor.pdf>.
- LINGNAU, A. (2001). «An improved environment for floats». <http://www.ctan.org/tex-archive/macros/latex/contrib/float/float.pdf>.
- MITTELBAACH, F. (2006). «An environment for multicolumn output». <http://www.ctan.org/tex-archive/macros/latex/required/tools/multicol.pdf>.
- THORUP, K. K., JENSEN, F. e ROWLEY, C. (2005). «The calc package. infix arithmetic notation in L^AT_EX». <http://www.ctan.org/tex-archive/macros/latex/required/tools/calc.pdf>.
- WILSON, P. (2003). <http://www.ctan.org/tex-archive/obsolete/macros/latex/contrib/misc/chngpage.sty>.
- ▷ Agostino De Marco
Università degli Studi di Napoli "Federico II"
Dipartimento di Ingegneria Aerospaziale
`agostino dot demarco at unina dot it`
- ▷ Massimiliano Dominici
`mlgdominici at interfree dot it`