GUIT meeting 2015

Generare documenti LaTeX programmando

Roberto Giacomelli, Gianluca Pignalberi

Trento, 17 ottobre 2015

Realizzare documentazione

In moltissimi casi è necessario produrre documentazione a partire da un insieme non banale di dati.

definiamo questi lavori come
→ ***progetti di documentazione*** ←

I dati sono caratterizzati da almeno uno di questi fattori:

- richiedono un'elaborazione che è troppo onerosa da svolgere manualmente;
- sono disponibili all'interno di un'applicazione che crea report.

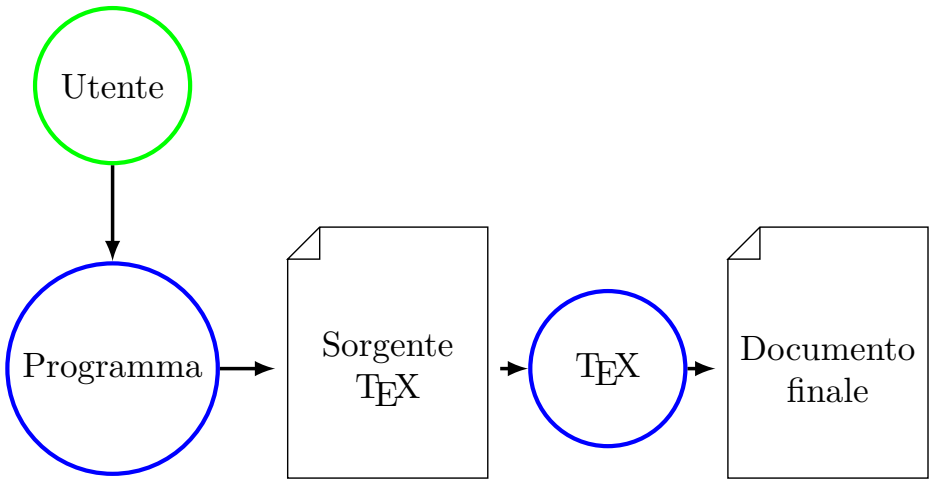
Alcuni esempi

Solo una piccola casistica:

- il report complesso con dati residenti in un database;
- il classico *mail merge*;
- il computo metrico con centinaia di voci e migliaia di misurazioni;
- la griglia di attività.

Una possibile soluzione con TEX

Introduzione nel *progetto di documentazione* del sistema TEX!



Ecco alcuni progetti concreti illustrati sulle pagine di ArsTEXnica:

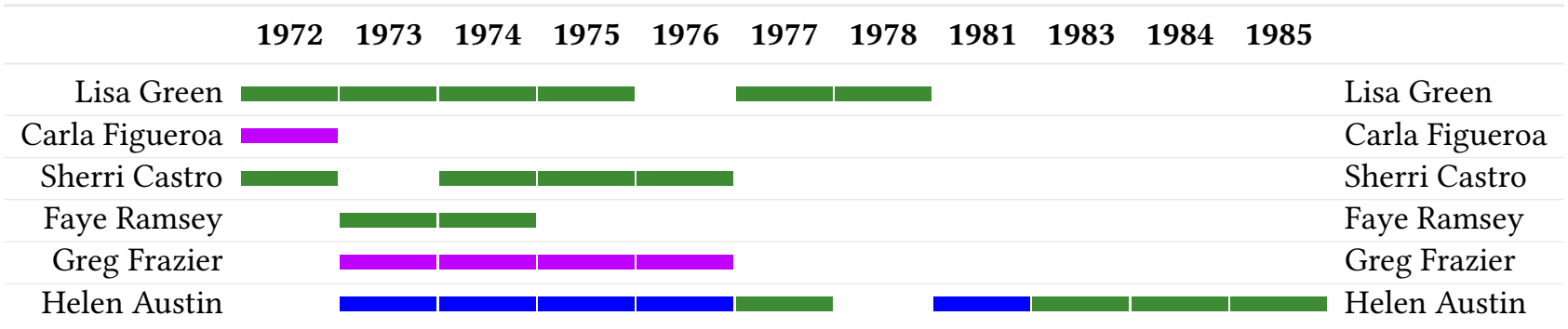
- Applicazione "Facile" del Comune di Napoli — lettere interne;
- Menù per ristorante — web app in PHP;
- Rapporti di prova — documenti di valore tecnico/legale.

Gli esempi reali

- griglia di attività
  - linguaggio Lua;
  - produzione del sorgente tramite il tipo stringa;
  - formato dei dati d'ingresso: Lua data description.
- l'inventario di magazzino
  - linguaggio Python;
  - produzione del sorgente tramite template;
  - formato dei dati d'ingresso: database relazionale;

Griglia di attività

Uno degli esempi reali riportato nell'articolo tratta di una tabella proposta da Francesco Endrici: barre colorate rappresentano le attività svolte anno per anno da membri di un gruppo.
Un piccolo estratto della griglia è qui riportato con nomi casuali e senza legenda:



Lua, a scripting language

Lua è un linguaggio dalla sintassi essenziale ideato dal gruppo di Roberto Ierusalimschy della Pontifical Catholic University di Rio de Janeiro in Brazile sin dal 1993.

Un breve corso su Lua :-)
`http://parliamodi-ubuntu.blogspot.it/p/corso-lua-indice.html`

Scriveremo un programma Lua per:

1. leggere i dati della griglia nel formato *data description*;
2. *espandere* gli intervalli di attività in due strutture intermedie;
3. creare la lista da usare come *buffer* delle righe di codice LaTeX di un normale ambiente `tabular`;
4. salvare — in modo efficiente — la lista in un file su disco pronto per essere compilato.

Il costruttore di tabelle Lua

La tabella in Lua è l'unica struttura dati predefinita nel linguaggio.
Si tratta di un *dizionario* chiave/valore.
Il *costruttore* istanzia oggetti tabella dal formato letterale — fu ispirato, ironia della sorte, dai record di BibTEX.

```
t1 = {} -- empty table
t2 = {1994, 2001} -- == {[1] = 1994, [2]= 2001}
t3 = { name = "Lisa Green",
       age = 24,
       city = {"UK","London",}
     }
-- indexing
print( t3["name"] ) --> stampa "Lisa Green"

-- dot notation
print( t3.name ) --> stampa "Lisa Green"
```

Formato Lua *data description*

Il formato *data description* altri non è che il costruttore di tabelle di Lua. Come rappresentare le attività annuali di un gruppo di persone?

```
{name = "Lisa Green",
   task = {
      ASCI = {{1972,1974},},
      EG = {1975, {1977,1978},},
   }
}
{name = "Carla Figueroa",
   task = {
      AsE = {1972,},
   }
}
...
```

Formato Lua *data description*

Il formato *data description* altri non è che il costruttore di tabelle di Lua. Come rappresentare le attività annuali di un gruppo di persone?

```
      {name = "Lisa Green",
   task = {
      ASCI = {{1972,1974},},
      EG = {1975, {1977,1978},},
   }
}
      {name = "Carla Figueroa",
   task = {
      AsE = {1972,},
   }
}
...
```

Formato Lua *data description*

Il formato *data description* altri non è che il costruttore di tabelle di Lua. Come rappresentare le attività annuali di un gruppo di persone?

```
service{name = "Lisa Green",
    task = {
        ASCI = {{1972,1974},},
        EG = {1975, {1977,1978},},
    }
}
service{name = "Carla Figueroa",
    task = {
        AsE = {1972,},
    }
}
...
```

Vantaggi del formato Lua data description

Il formato *data description* consente di

- dare una struttura gerarchica ai dati con un modello aderente al loro significato;
- costruire una sorta d'archivio dati nel semplice formato testo editabile dall'utente;
- identare liberamente il codice e inserire commenti;
- inserire come valori espressioni complesse come ad esempio formule numeriche;
- non occorre fare nessuno sforzo per il parsing dei dati trattandosi di codice Lua.

Lettura ed elaborazione dei dati

Il caricamento del file di descrizione da luogo all'esecuzione della funzione `service()` che prepara in memoria i dati che riletti genereranno il codice opportuno dell'ambiente `tabular`:

```
service{name = "Helen Austin",
    task = {
        EG = {1977, {1983,1985},},
        LC = {{1973,1975},1981,},
    }
}
```

Lettura ed elaborazione dei dati

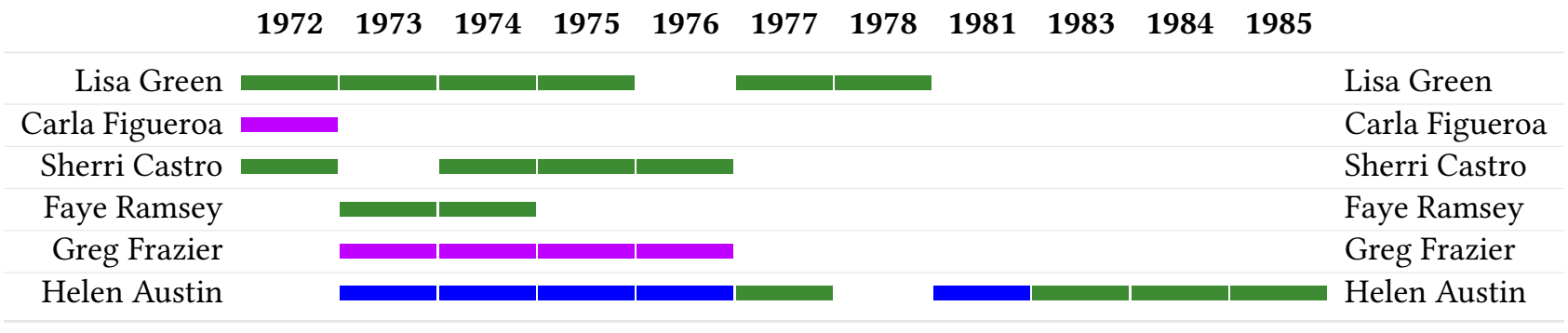
Il caricamento del file di descrizione da luogo all'esecuzione della funzione `service()` che prepara in memoria i dati che riletti genereranno il codice opportuno dell'ambiente `tabular`:

```
{   name = "Helen Austin",
    years = {
        [1977] = "EG",
        [1983] = "EG",
        [1984] = "EG",
        [1985] = "EG",
        [1973] = "LC",
        [1974] = "LC",
        [1975] = "LC",
        [1981] = "LC",
}}
```

Lettura ed elaborazione dei dati

Il caricamento del file di descrizione da luogo all'esecuzione della funzione `service()` che prepara in memoria i dati che riletti genereranno il codice opportuno dell'ambiente `tabular`:

```
...
Helen Austin & & \LC & \LC & \LC & \LC & \EG
& & & \EG & \EG & \EG & Helen Austin\\
...
```



L'inventario di un esercizio commerciale

L'inventario annuale è l'elenco delle giacenze di magazzino alla data del 31 dicembre.
Il report dovrà contenere un frontespizio, l'elenco della merce suddivisa per reparto e fornitore in una tabella multipagina a due colonne, e la tabella riassuntiva delle giacenze totali per reparto.

Supplier: *Good Shirts*

Description	Q.ty	Cost ($)	Total ($)
AQ12 PL21 — T-Shirt Pluton series	12	12.50	150.00
SZ02 PL24 — T-Shirt Pluton series	8	19.10	152.80
W964 JP02 — T-Shirt Jupiter series	6	12.90	77.40
BP01 JP10 — T-Shirt Jupiter series	20	15.80	316.00
XVYY E5 — T-Shirt Earth series	5	14.80	74.00
XVYY E19 — T-Shirt Earth series	3	12.20	36.60
XVYY E41 — T-Shirt Earth series	19	11.30	214.70
NEAR 956 — T-Shirt Moon series	5	14.80	74.00
	78		1 095.50

Python, another scripting language

Python è un linguaggio dalla ricca dotazione di tipi dati e librerie ideato da Guido van Rossum dal 1980 presso il National Research Institute for Mathematics and Computer Science di Amsterdam.

Scriveremo un programma Python per:

1. leggere i dati da un database SQLite3;
2. definire le funzioni accessorie per la presentazione dei dati;
3. caricare il *template* e unirvi i dati;
4. salvare il risultato — il sorgente del report d'inventario — su disco per la successiva compilazione con `pdflatex`.

La tecnologia dei template

Lavorare con il tipo di dato *stringa* limita le modifiche al report e il riuso del codice. La tecnologia dei *template* introduce invece nel programma il potente concetto di **modello di documento**.

È possibile:

- utilizzare funzioni di formattazione predefinite o personalizzate;
- nidificare i template uno dentro l'altro;
- ereditare da un template di base per condividere la configurazione generale di un gruppo di documenti;
- semplificare il programma per incentrarlo solo sulla logica.

A patto che:

- vi sia corrispondenza biunivoca **template** <-> **dati** sia per struttura che per nomi.

Esempio di template con la sofisticata libreria **Jinja**

→ **Jinja2** — http://jinja.pocoo.org/ ←

```
from jinja2 import Environment
env = Environment(
    variable_start_string="<<|",
    variable_end_string="|>>"
)
tmpl = r"""
\documentclass{minimal}
\usepackage{pdfpages}
\begin{document}
{% for doc in doc_list -%}
    \includepdf[pages=-]{<<|doc|>>}
{% endfor -%}
\end{document}
"""
t_eng = env.from_string(tmpl)
docs = ("as541","as656","as163","as048","as525","as854")
print(t_eng.render(doc_list=docs))
```

Esempio di template con la sofisticata libreria **Jinja**

→ **Jinja2** — `http://jinja.pocoo.org/` ←

```
\documentclass{minimal}
\usepackage{pdfpages}
\begin{document}
\includepdf[pages=-]{as541}
\includepdf[pages=-]{as656}
\includepdf[pages=-]{as163}
\includepdf[pages=-]{as048}
\includepdf[pages=-]{as525}
\includepdf[pages=-]{as854}
\end{document}
```

Lettura ed elaborazione dei dati

Schema della struttura a più livelli in cui sono memorizzati tutti gli articoli d'inventario. I simboli sono gli stessi dei tipi in Pyhton ovvero le parentesi graffe definiscono un dizionario e le parentesi quadre una lista. I tre puntini indicano la ripetibilità dell'elemento. Nel template questo dizionario è facilmente iterato su tre livelli per generare il codice LaTeX di ambienti tabular.

```
{'dep-key': {'sup-key':[{'descr':<val>, 'qty':<val>, 'cost':<val>},
                        {'descr':<val>, 'qty':<val>, 'cost':<val>},
                        {'descr':<val>, 'qty':<val>, 'cost':<val>},
                        {'descr':<val>, 'qty':<val>, 'cost':<val>},
                        {'descr':<val>, 'qty':<val>, 'cost':<val>},
                        {'descr':<val>, 'qty':<val>, 'cost':<val>},
                        ...
                       ]
             ...
            }
   ...
}
```

Conclusioni

Vantaggi del processo:

- ottima qualità tipografica dei report;
- completo controllo dei report (elementi grafici, tabelle, geometria, font, eccetera);
- adattabilità del report alle esigenze future siano esse nuove o di affinamento (evoluzione).

Conoscenze necessarie:

- ovviamente conoscenza del sistema TEX;
- padronanza dei princìpi di programmazione;
- scelta del linguaggio e delle eventuali librerie di templating (in questo l'articolo aiuta);
- eventuale approfondimento di un linguaggio specifico e utilizzo di librerie esterne.

Un'evoluzione interessante in sostituzione del templating

Scrivere una libreria che implementi direttamente il concetto di sorgente TEX!

Un qualcosa di simile alla creazione di un oggetto d'alto livello che rappresenta il documento, composto a sua volta da oggetti tipografici inseriti in sequenza tramite metodi, in modo simile alla libreria iText.

Ulteriore soluzione tutta da esplorare...

Programmare in Lua all'interno di LuaTEX o LuajitTEX!

La compilazione del sorgente `.tex` del report si potrà distinguere in due fasi:

1. lettura dei dati ed elaborazione (con eventuale accesso a database);
2. una volta pronto in memoria, composizione del risultato con inserimento delle informazioni nel flusso di *token* o tramite gli oggetti *nodo*.

Ringraziamenti

Grazie per l'attenzione

e grazie mille Trento!

Domande?