Produzione di documenti navigabili con Stata

Rosa Gini Jacopo Pasquini
rosa.gini@arsanita.toscana.it
jacopo.pasquini@arsanita.toscana.it

Agenzia Regionale di Sanità della Toscana

Milano, ottobre 2005

- Tim Berners-Lee stava cercando un modo per gestire e distribuire fra i colleghi grandi quantità d'informazioni e nel 1989 propose l'adozione del linguaggio HTML al laboratorio europeo per la fisica delle particelle (CERN).

- Tim Berners-Lee stava cercando un modo per gestire e distribuire fra i colleghi grandi quantità d'informazioni e nel 1989 propose l'adozione del linguaggio HTML al laboratorio europeo per la fisica delle particelle (CERN).
- Egli propose una rete di documenti connessi tramite collegamenti ipertestuali e ospitati da computer chiamati server ipertestuali. Seguendo i progressi di questa idea, la piccola rete iniziale divenne il World Wide Web.

Cos'è un file HTML?

- Un file HTML non è altro che un file di testo, sostanzialmente identico ai file di tipo `txt` che si possono scrivere con un comune editor di testo. Per funzionare come pagina web, questo file di testo deve essere rinominato con estensione `.html` o `.htm` e deve contenere, oltre al testo vero e proprio, anche le istruzioni che consentono al browser di riconoscerlo e gestirlo correttamente.

Cos'è un file HTML?

- Un file HTML non è altro che un file di testo, sostanzialmente identico ai file di tipo `txt` che si possono scrivere con un comune editor di testo. Per funzionare come pagina web, questo file di testo deve essere rinominato con estensione `.html` o `.htm` e deve contenere, oltre al testo vero e proprio, anche le istruzioni che consentono al browser di riconoscerlo e gestirlo correttamente.
- Queste istruzioni sono chiamate *tag* (marcatori).

Cos'è un file HTML?

- Un file HTML non è altro che un file di testo, sostanzialmente identico ai file di tipo `txt` che si possono scrivere con un comune editor di testo. Per funzionare come pagina web, questo file di testo deve essere rinominato con estensione `.html` o `.htm` e deve contenere, oltre al testo vero e proprio, anche le istruzioni che consentono al browser di riconoscerlo e gestirlo correttamente.
- Queste istruzioni sono chiamate *tag* (marcatori).
- Un tag è un comando racchiuso tra i segni $<$ (minore) e $>$ (maggiore).

Cos'è un file HTML?

- Un file HTML non è altro che un file di testo, sostanzialmente identico ai file di tipo `txt` che si possono scrivere con un comune editor di testo. Per funzionare come pagina web, questo file di testo deve essere rinominato con estensione `.html` o `.htm` e deve contenere, oltre al testo vero e proprio, anche le istruzioni che consentono al browser di riconoscerlo e gestirlo correttamente.
- Queste istruzioni sono chiamate *tag* (marcatori).
- Un tag è un comando racchiuso tra i segni $<$ (minore) e $>$ (maggiore).
- Quando il browser incontra questi simboli capisce che si tratta d'istruzione da eseguire e non di testo da visualizzare sullo schermo.

Nella maggior parte dei casi i tag funzionano in coppia, richiedono cioè un'istruzione d'apertura ed una di chiusura secondo lo schema:

<TAG DI INIZIO>
elemento cui applicare l'istruzione
</TAG DI FINE>

- Il comando contenuto nel tag finale è *sempre uguale* a quello del tag iniziale, tranne che è preceduto da una barra /.

- Il comando contenuto nel tag finale è *sempre uguale* a quello del tag iniziale, tranne che è preceduto da una barra /.
- A differenza di quanto accade in altri linguaggi, per HTML non fa differenza se i comandi sono in *MAIUSCOLO* o in *minuscolo*.

A titolo d'esempio, il tag <B> identifica il grassetto ed il tag <I> il corsivo. Così per formattare in grassetto e corsivo una parola si deve scrivere:

```
<B><I>Testo</I></B>
```

oppure: `<I><B>Testo</B></I>`,

ma non: `<B><I>Testo</B></I>`

I tag principali sono:

- `<b>grassetto</b>` per **grassetto**
- `<i>corsivo</i>` per *corsivo*
- `<p align = ? >` per allineare un paragrafo a destra, sinistra o centro

I tag di tabella

I tag più importanti per noi sono:

- `<table>` che indica la tabella;
- `<tr>` che indica la riga;
- `<th>` che indica un particolare elemento della tabella, da formattare come "titolo";
- `<td>` che indica un normale elemento di tabella.

Se un elemento di tabella dve occupare *più di una colonna*, all'interno del tag `<td>` o `<th>` si indicherà il comando `colspan='3'` (se si deve occupare 3 colonne).

Esempio

Ad esempio la tabella seguente (che ritroveremo nel seguito)

```
<table>
<tr><th colspan='6' align=center> Height</th></tr>
<tr><td> Gender</td><td>N</td><td>Mean</td><td>St. Dev.</td><td>Lower Bound</td><td>Upper Bound</td></tr>
<tr><td>Female</td><td>11</td><td>118.6</td><td>2.1</td><td>113.9</td><td>123.2</td></tr>
<tr><td>Male</td><td>9</td><td>113.1</td><td>2.2</td><td>108.1</td><td>118.1</td></tr>
</table>
```

viene letta da un qualsiasi browser come tabella, e il contenuto viene quindi allineato di conseguenza.

Un *documento* HTML non è costituito da un *solo* file, bensì da *tanti* file, scritti in linguaggio HTML, e collegati l'uno all'altro in modo logico.

> *A segment of text or a graphical item that serves as a cross-reference between parts of a hypertext document or between files or hypertext documents. Also called hotlink, hyperlink.*

Il tag utilizzato per i link si chiama *ancora*:

- `<a href="url">...</a>`
  Collegamento ipertestuale
- `<a href="#name">...</a>`
  Collegamento ad un'ancora nel documento stesso
- `<a href="URL#name">...</a>`
  Collegamento ad un'ancora in un altro documento
- `<a name="name">...</a>`
  Ancora in un documento
- `<a href="mailto:e-mail">...</a>`
  Collegamento ad una e-mail

È possibile collegare tra loro oggetti diversi (stringhe, immagini, caratteri, grafici . . .)

L'esistenza di questi collegamenti logici è ciò che rende la pagina HTML *navigabile*. Questo permette al documento di condensare molte informazioni. I collegamenti (link) quindi sono fondamentali per generare output complessi.

HTML consente di integrare le funzioni di

formattazione + contenuti + controllo

Alcune risorse HTML

- http://www.w3.org

Alcune risorse HTML

- http://www.w3.org
- http://www.manuali.it

Alcune risorse HTML

- http://www.w3.org
- http://www.manuali.it
- http://www.html.it

Alcune risorse HTML

- http://www.w3.org
- http://www.manuali.it
- http://www.html.it
- ...

Il comando

- Il comando

genera una directory *nomedir*, oppure, se essa esiste già, genera un errore;

Il comando

- Il comando

genera una directory *nomedir*, oppure, se essa esiste già, genera un errore;

Il comando

- Il comando

  `mkdir` *`nomedir`*

  genera una directory *nomedir*, oppure, se essa esiste già, genera un errore;

Il comando

- Il comando

  `mkdir` *`nomedir`*

  genera una directory *nomedir*, oppure, se essa esiste già, genera un errore;
- per dire a Stata di generare la directory *solo se essa non esiste ancora*, e di non far nulla altrimenti, si dà il comando

Il comando

- Il comando

  `mkdir` *`nomedir`*

  genera una directory *nomedir*, oppure, se essa esiste già, genera un errore;
- per dire a Stata di generare la directory *solo se essa non esiste ancora*, e di non far nulla altrimenti, si dà il comando

Il comando

- Il comando

  `mkdir` *nomedir*

  genera una directory *nomedir*, oppure, se essa esiste già, genera un errore;
- per dire a Stata di generare la directory *solo se essa non esiste ancora*, e di non far nulla altrimenti, si dà il comando

  `capture mkdir` *nomedir*

Il nome delle directory

- Il nome di una directory può essere specificato

Il nome delle directory

- Il nome di una directory può essere specificato
  - o *relativamente* al punto in cui si trova Stata;

Il nome delle directory

- Il nome di una directory può essere specificato
  - o *relativamente* al punto in cui si trova Stata;
  - oppure in modo *assoluto*, a partire dalla radice dell'albero delle directory (se si è sotto Windows `C:/`, se si è sotto Unix/Linux semplicemente `/`).

Il nome delle directory

- Il nome di una directory può essere specificato
  - o *relativamente* al punto in cui si trova Stata;
  - oppure in modo *assoluto*, a partire dalla radice dell'albero delle directory (se si è sotto Windows `C:/`, se si è sotto Unix/Linux semplicemente `/`).
- In un programma di Stata non conviene mai usare i nomi assoluti; se questo si rende necessario, piuttosto si può mettere in una macro il nome della directory di partenza, e scrivere i nomi assoluti delle directory usando la macro come radice; in questo modo se si sposta la radice basta cambiare il valore della macro.

Esempio

- Se mi trovo nella directory `C:/pippo` e dò il comando

  verrà generata la sottodirectory `figure` all'interno della directory `C:/pippo`;

Esempio

- Se mi trovo nella directory `C:/pippo` e dò il comando

  verrà generata la sottodirectory `figure` all'interno della directory `C:/pippo`;

Esempio

- Se mi trovo nella directory `C:/pippo` e dò il comando

```
capture mkdir figure
```

verrà generata la sottodirectory `figure` all'interno della directory `C:/pippo`;

Esempio

- Se mi trovo nella directory `C:/pippo` e dò il comando

  ```
  capture mkdir figure
  ```

  verrà generata la sottodirectory `figure` all'interno della directory `C:/pippo`;
- se lo stesso comando viene dato dall'interno della directory `C:/pluto` la sottodirectory `figure` viene creata nella directory `C:/pluto`;

Esempio

- Se mi trovo nella directory `C:/pippo` e dò il comando

  ```
  capture mkdir figure
  ```

  verrà generata la sottodirectory `figure` all'interno della directory `C:/pippo`;
- se lo stesso comando viene dato dall'interno della directory `C:/pluto` la sottodirectory `figure` viene creata nella directory `C:/pluto`;
- se invece di dare il comando con il cammino relativo si dà il comando assoluto, in questo modo

  lanciando il comando verrà creata la directory C:/pippo/figure, *indipendentemente* da quale sia la directory da cui viene lanciato il comando.

Esempio

- Se mi trovo nella directory `C:/pippo` e dò il comando

```
capture mkdir figure
```

  verrà generata la sottodirectory `figure` all'interno della directory `C:/pippo`;
- se lo stesso comando viene dato dall'interno della directory `C:/pluto` la sottodirectory `figure` viene creata nella directory `C:/pluto`;
- se invece di dare il comando con il cammino relativo si dà il comando assoluto, in questo modo

  lanciando il comando verrà creata la directory C:/pippo/figure, *indipendentemente* da quale sia la directory da cui viene lanciato il comando.

Esempio

- Se mi trovo nella directory `C:/pippo` e dò il comando

```
capture mkdir figure
```

  verrà generata la sottodirectory `figure` all'interno della directory `C:/pippo`;
- se lo stesso comando viene dato dall'interno della directory `C:/pluto` la sottodirectory `figure` viene creata nella directory `C:/pluto`;
- se invece di dare il comando con il cammino relativo si dà il comando assoluto, in questo modo

```
local dir C:/pippo/
capture mkdir ‘dir’figure
```

  lanciando il comando verrà creata la directory C:/pippo/figure, *indipendentemente* da quale sia la directory da cui viene lanciato il comando.

Premessa

- Un archivio Stata comunemente serve per contenere dati grezzi, su cui si compiono delle analisi i cui risultati appaiono a schermo, in una forma tabellare propria di Stata stesso.

Premessa

- Un archivio Stata comunemente serve per contenere dati grezzi, su cui si compiono delle analisi i cui risultati appaiono a schermo, in una forma tabellare propria di Stata stesso.
- La tecnica di produzione di output formattato più supportata da Stata prevede invece che l'analisi produca un *nuovo* archivio, che sia la versione `dta` della tabella di risultati che vogliamo pubblicare.

Premessa

- Un archivio Stata comunemente serve per contenere dati grezzi, su cui si compiono delle analisi i cui risultati appaiono a schermo, in una forma tabellare propria di Stata stesso.
- La tecnica di produzione di output formattato più supportata da Stata prevede invece che l'analisi produca un *nuovo* archivio, che sia la versione `dta` della tabella di risultati che vogliamo pubblicare.
- L'esempio più semplice di questa tecnica è il comando `contract`, che "contrae" una variabile (solitamente categoriale), ovvero genera un archivio che contiene un'osservazione per ogni modalità della variabile e genera la variabile `_freq`, che contiene la frequenza di ciascuna modalità.

Premessa

- Un archivio Stata comunemente serve per contenere dati grezzi, su cui si compiono delle analisi i cui risultati appaiono a schermo, in una forma tabellare propria di Stata stesso.
- La tecnica di produzione di output formattato più supportata da Stata prevede invece che l'analisi produca un *nuovo* archivio, che sia la versione `dta` della tabella di risultati che vogliamo pubblicare.
- L'esempio più semplice di questa tecnica è il comando `contract`, che "contrae" una variabile (solitamente categoriale), ovvero genera un archivio che contiene un'osservazione per ogni modalità della variabile e genera la variabile `_freq`, che contiene la frequenza di ciascuna modalità.
- Tra i numerosi comandi che lavorano in questa logica citiamo `xcontract`, `statsby`, `parmby`...

- Il comando `listtex` (scritto da Roger Newson) serve a scrivere in un file la lista dell'archivio attualmente in memoria.

- Il comando `listtex` (scritto da Roger Newson) serve a scrivere in un file la lista dell'archivio attualmente in memoria.
- Questo comando serve da complemento alla tecnica appena illustrata: prima (usando `xcontract`, `statsby`...) genero un archivio contenente la tabella che voglio stampare, e poi con `listtex` la stampo in un file.

- Il comando `listtex` (scritto da Roger Newson) serve a scrivere in un file la lista dell'archivio attualmente in memoria.
- Questo comando serve da complemento alla tecnica appena illustrata: prima (usando `xcontract`, `statsby`...) genero un archivio contenente la tabella che voglio stampare, e poi con `listtex` la stampo in un file.
- Il comando è molto ricco di opzioni. Quella fondamentale che serve al nostro scopo è `rstyle(html)`, che fa in modo che le righe dell'archivio vengano stampate in formato `html` (e quindi precedute dai tag di inizio riga,...).

- Il comando `listtex` (scritto da Roger Newson) serve a scrivere in un file la lista dell'archivio attualmente in memoria.
- Questo comando serve da complemento alla tecnica appena illustrata: prima (usando `xcontract`, `statsby`...) genero un archivio contenente la tabella che voglio stampare, e poi con `listtex` la stampo in un file.
- Il comando è molto ricco di opzioni. Quella fondamentale che serve al nostro scopo è `rstyle(html)`, che fa in modo che le righe dell'archivio vengano stampate in formato `html` (e quindi precedute dai tag di inizio riga,...).
- Per stampare una tabella `html` completa, però, bisognerà specificare anche il suo header e il suo footer, usando le opzioni `headlines()` e `footlines()`, rispettivamente.

Esempio

In un archivio, `height.dta`, sono archiviate l'altezza e il sesso di 20 bambini. Si vuole calcolare le altezze medie, con intervalli di confidenza, divise per sesso, e pubblicarle su una pagina HTML.

```
use height.dta, clear
capture mkdir tabelle
preserve
statsby "ci height" N=r(N) mean=r(mean) se=r(se) lb=r(lb) ub=r(ub),by(gender)
listtex using tabelle/tab_height.htm, rstyle(html) headlines("<table>" "<tr><td
>  colspan='6' align=center> Height</td></tr>" "<tr><td> Gender</td><td>N</td><
> td>Mean</td><td>St. Dev.</td><td>Lower Bound</td><td>Upper Bound</td></tr>")
> footlines("</table>") replace
restore
```

Esempio

Questo codice genera, nella diretory `tabelle`, il file `tab_height.htm`, che contiene il seguente testo:

```
<table>
<tr><td colspan='6' align=center> Height</td></tr>
<tr><td> Gender</td><td>N</td><td>Mean</td><td>St. Dev.</td><td>Lower Bound</td><td>Upper Bound</td></tr>
<tr><td>Female</td><td>11</td><td>118.5727</td><td>2.084274</td><td>113.9287</td><td>123.2168</td></tr>
<tr><td>Male</td><td>9</td><td>113.1111</td><td>2.187874</td><td>108.0659</td><td>118.1564</td></tr>
</table>
```

che viene letto da un qualsiasi browser.

Nota

Quando si stampano i risultati di una stima, spesso si preferisce stampare solo alcune cifre decimali. Per far questo prima di dare il comando `listtex` è necessario trasformare le varibili numeriche che si vanno a stampare in stringhe, con un adeguato numero di decimali. Nell'esempio precedente sarebbe stato più opportuno, prima di stampare con `listtex`, dare il comando

```
foreach var of varlist mean se lb ub{
tostring ‘var’, force format(%4.1f) replace
}
```

- Il formato preferito per la pubblicazione su web di immagini è il png.

- Il formato preferito per la pubblicazione su web di immagini è il png.
- Stata può esportare le immagini in formato png, con la sintassi

- Il formato preferito per la pubblicazione su web di immagini è il png.
- Stata può esportare le immagini in formato png, con la sintassi

```
graph export nomefile, as(png)
```

- Il formato preferito per la pubblicazione su web di immagini è il png.
- Stata può esportare le immagini in formato png, con la sintassi

  `graph export` *`nomefile`*`, as(png)`

- Tuttavia questa esportazione non ammette opzioni, in particolare non permette di fissare la risoluzione

- Il formato preferito per la pubblicazione su web di immagini è il png.
- Stata può esportare le immagini in formato png, con la sintassi

  `graph export` *`nomefile`*`, as(png)`

- Tuttavia questa esportazione non ammette opzioni, in particolare non permette di fissare la risoluzione
- Per ovviare a questo problema, se necessario, si può esportare (temporaneamente) il grafico in eps, e da qui convertirlo tramite il software libero *Ghostscript*, che gestisce moltissime opzioni.

Sintassi tipica

```
capture mkdir figure
/* sintassi per generare il grafico a partire dai dati*/
graph ...
/* sintassi per esportare il grafico in png*/
graph export nomefile, as(png)
/* ALTERNATIVAMENTE: sintassi per invocare ghostscript */
local gho=cond(c(os)=="Unix","gs","gswin32")
tempfile graph
graph export ‘graph’.eps,as(eps) replace
winexec ‘gho’ -q -dEPSCrop -dNOPROMPT -dBATCH -dNOPAUSE  -sDEVICE=png16m -r200
> -dGraphicsAlphaBits=4 -sOutputFile=figure/nomefile.png ‘graph’.eps
```

- Una volta generati tutti i 'pezzi' (tabelle, grafici) che devono essere presentati nel documento HTML, resta ancora il problema di unirli insieme.

- Una volta generati tutti i 'pezzi' (tabelle, grafici) che devono essere presentati nel documento HTML, resta ancora il problema di unirli insieme.
- Per far questo è necessario chieder a Stata di scrivere direttamente un file HTML...

- Una volta generati tutti i 'pezzi' (tabelle, grafici) che devono essere presentati nel documento HTML, resta ancora il problema di unirli insieme.
- Per far questo è necessario chieder a Stata di scrivere direttamente un file HTML...
- ...cosa che Stata può fare facilmente.

- Come una segretaria diligente, Stata trascrive ciò che gli si dice di scrivere, eventualmente sostituendo, nelle stringhe che gli si indica, i valori di macro definite precedentemente.

- Come una segretaria diligente, Stata trascrive ciò che gli si dice di scrivere, eventualmente sostituendo, nelle stringhe che gli si indica, i valori di macro definite precedentemente.
- Questa sua capacità, che è utilissima in moltissime circostanze, casca a fagiolo anche in questa.

La sintassi

- Per scrivere il file `main.htm`, in cui metteremo in una tabella i link a tutti i file che abbiamo generato, dobbiamo scegliere un nomignolo, chiamato *handle*, che useremo durante il processo di scrittura del file. La cosa migliore è scegliere come nomignolo un nome temporaneo.

La sintassi

- Per scrivere il file `main.htm`, in cui metteremo in una tabella i link a tutti i file che abbiamo generato, dobbiamo scegliere un nomignolo, chiamato *handle*, che useremo durante il processo di scrittura del file. La cosa migliore è scegliere come nomignolo un nome temporaneo.
- Per scrivere il file bisogna prima *aprirlo in scrittura* (comando `file open ..., write replace`), poi scriverci (comando `file write`), infine chiuderlo (comando `file close`).

La sintassi

- Per scrivere il file `main.htm`, in cui metteremo in una tabella i link a tutti i file che abbiamo generato, dobbiamo scegliere un nomignolo, chiamato *handle*, che useremo durante il processo di scrittura del file. La cosa migliore è scegliere come nomignolo un nome temporaneo.
- Per scrivere il file bisogna prima *aprirlo in scrittura* (comando `file open ..., write replace`), poi scriverci (comando `file write`), infine chiuderlo (comando `file close`).
- Supponiamo di aver costruito per ogni variabile di una lista contenuta nella macro `varlist` una tabella e una figura. Il file principale verrà scritto come segue.

```
use brfeed.dta,replace
local varlist " pos natio edu brfeed"
tempname main
file open ‘main’ using main.htm,write replace
file write ‘main’ ‘"<!--File automatically generated by Stata-->"’ _n
file write ‘main’ ‘"<h2 align=center>Title of the page<h2>"’ _n
file write ‘main’ ‘"<br><br>"’ _n
file write ‘main’ ‘"<table align=center width=30%>"’ _n
file write ‘main’ ‘"<tr><td><b>Variable</b></td><td><b>Table</b></td><td><b>Gra
> ph</b></td></tr>"’ _n
foreach var of local varlist{
        local title:variable label ‘var’
        file write ‘main’ ‘"<tr><td>‘title’</td><td><a Href=’tabelle/tab_‘var’.
> htm’>[X]<a></td><td><a Href=’figure/tabfig_‘var’.png’>[X]<a></td><tr>"’ _n
        }
file write ‘main’ ‘"</table>"’ _n
file close ‘main’
```

Esempio

Un sito prodotto con questa modalità: Isa 65+.

A partire da questi strumenti si possono generare con Stata documenti `tex`, codice `php`, codice `sql`...


Agenzia Regionale di Sanità della Toscana.
Isa 65+. Indicatori sulla salute e l'assistenza agli anziani.
www.arsanita.toscana.it, 2005.


Rosa Gini, Jacopo Pasquini.
Automatic generation of documents.
Submitted for publication, 2005.


Ghostscript, Ghostview and GSview.
http://www.cs.wisc.edu/~ghost/.


Roger Newson.
2003. st0043:
Confidence intervals and p-values for delivery to the end user.
The Stata Journal 3(3) 245-269.


HyperText Markup Language Home Page.
www.w3.org/MarkUp, 2004.