HTML::Template

Un ponte tra il programmatore e il web designer

Dott. Emiliano Bruni

YAT,oJ!

Dott. Emiliano Bruni

YAT,oJ!

Yet Another Template

Dott. Emiliano Bruni

YAT,oJ!

Yet Another Template
oh Jesus!

Dott. Emiliano Bruni

Dove vogliamo andare a parare

- Siti Web a carattere dinamico medio/grandi
- Due soggetti coinvolti
  - Programmatore
  - Web Designer
- Circa il 40% del lavoro sarà relativo all'integrazione delle pagine web disegnate dal Web designer

Come si lavora di solito nella realizzazione di un sito web dinamico?

- Realizzazione della parte dinamica con un modello HTML che verrà poi rimosso
- Realizzazione di un sito statico
- Fusione del sito statico con la parte dinamica
- Manutenzione del sito

Cosa c'è di sbagliato in questo modo di lavorare?

- Il prodotto finale è un file che mischia codice di scripting e codice HTML
- Il Web designer non è un programmatore
- Il Web designer usa un editor WYSIWYG
- Il Web designer non è più in grado di manutenere il prodotto finito
- La manutenzione del sito è a carico del programmatore
- Il Web designer fa un nuovo layout => DISASTRO

Qual'è un modello migliore? Dare a Cesare quel che è di Cesare.

- Split logica funzionale da quella di layout
- La pagina deve essere generata da due file
  - Elemento funzionale con solo codice di scripting
  - Elemento di layout con solo codice HTML
- Web designer userà il suo editor WYSIWYG
- Il programmatore non è più distratto dall 'HTML
- Manutenzione bilanciata tra Web designer e programmatore

Fare questo è un utopia? Proviamo cosi!

- Il browser richiama uno script in Perl (no HTML)
- Lo script esegue la logica funzionale
- Lo script valorizza delle variabili “segnaposto”
- Lo script carica il template HTML fatto dal web designer
- Lo script sostituisce i segnaposto
- La pagina dinamica viene stampata.

Perchè usare proprio HTML::Template per realizzare la nostra utopia.

- Segnaposto HTML-Like
  - `<TMPL_VAR> <TMPL_IF> <TMPL_ELSE> <TMPL_UNLESS> <TMPL_LOOP> <TMPL_INCLUDE>`
- Sintassi alternative per i segnaposto
  - `<TMPL_VAR>`
  - `%...%`
  - `<!-- TMPL_VAR -->`
- Possibilità di estendere il motore di templating
  - Opzione **filter**
  - HTML::Template::Extension

Case study

- Script all-in-one
- Caso degenere: pagina statica
- Script con separazione logica/layout
  - Ebay(z).it
    - Pagina di login
    - Elenco prodotti

II TAG <TMPL_VAR>

- <TMPL_VAR NAME=”parameter_name”>
- <!-- TMPL_VAR NAME=”parameter_name” -->
- %parameter_name%
- <...ESCAPE=”URL|HTML|JS”
- <...DEFAULT=”...”
- </TMPL_VAR>

Gestione decisionale <TMPL_IF> e <TMPL_UNLESS>

- <TMPL_IF NAME=”boolean”>...</TMPL_IF>
- ...<TMPL_ELSE>...
- <TMPL_UNLESS NAME=”boolean”>...

Gli utilissimi <TMPL_LOOP>

- <TMPL_LOOP NAME=”...”>...</TMPL_LOOP>
- __first__ (usare con TMPL_IF)
- __odd__ (usare con TMPL_IF)
- __even__ (usare con TMPL_IF)
- __inner__ (usare con TMPL_IF)
- __last__ (usare con TMPL_IF)
- __counter__ (usare con TMPL_VAR)

I plugin di H::T::Extension

- DO_NOTHING :-D
  - Skel di esempio per nuovi plugin
- SLASH_VAR
  - Aggiunge </TMPL_VAR>
- TAG_ATTRIBUTE_NORMALIZER
  - Rimuove attributi non H::T ai tag
- CSTART
  - Aggiunge <TMPL_CSTART> per selezionare solo una parte del template

I plugin di H::T::Extension

- DOC
  - Aggiunge <TMPL_DOC>...</TMPL_DOC> per inserire commenti inline
- HEAD_BODY
  - Ritorna separatamente il BODY e le proprietà dell'HEAD di un file template HTML
- IF_TERN
  - Aggiunge l'operatore bool_param?if_true:else

HTML::Template::Extension: differenze dal modulo “padre”

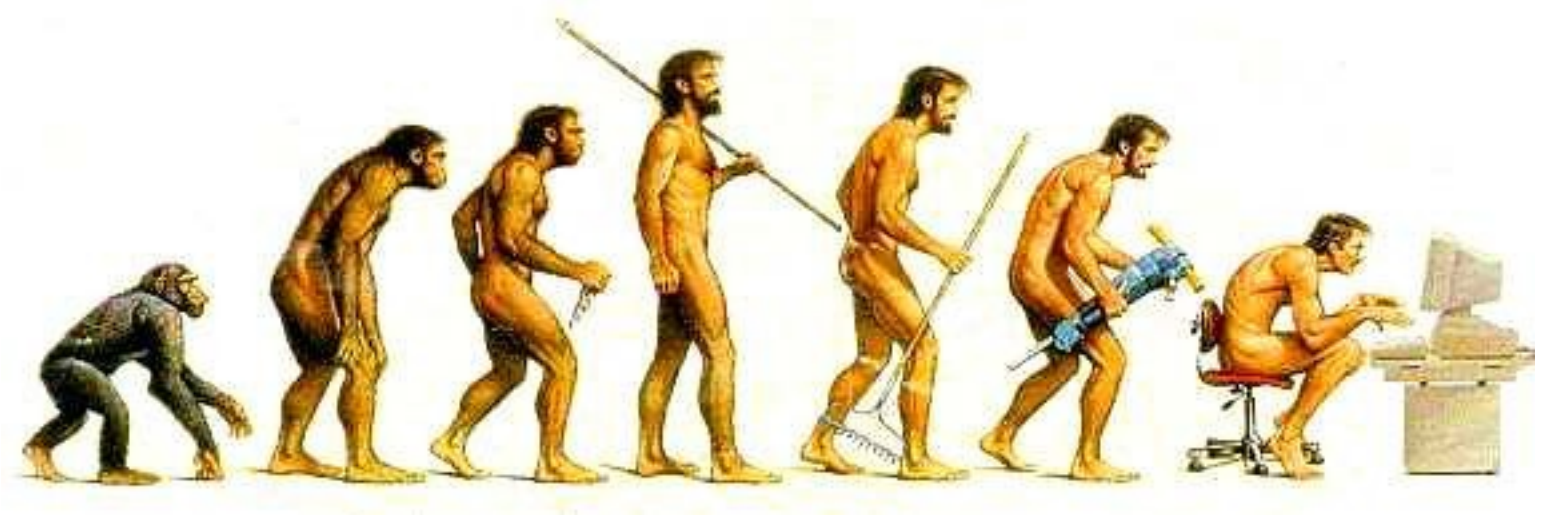
- No </TMPL_VAR> e quindi no segnaposto “visuali” in un editor WYSIWYG
- No TAG_ATTRIBUTE_NORMALIZER e quindi no class=”...” attribute nei TAG H::T per evidenziare i segnaposto
- No CSTART e quindi impossibile usare solo una parte del template

HTML::Template::Extension: differenze dal modulo “padre”

- No %...% e quindi problemi nei segnaposto dentro ai tag
  - `<IMG SRC=”<TMPL_VAR NAME=”id”>”>`
  - `<IMG SRC=”%id%”>`
- No IF_TERN e quindi problemi con checked
  - `<OPTION name=”xxx” <TMPL_IF is_checked>checked</TMPL_IF>>`
  - `<OPTION name=”xxx” %is_checked?checked%>`

L'evoluzione della specie

- 1989: Nascita del WWW
- 1993: Nascita dell'interfaccia CGI
- 1999: Nascita del modulo HTML::Template



Riferimenti

- HTML::Template sul CPAN
  - http://search.cpan.org/~samtregar/HTML-Template-2.7/Template.pm
- H::T::Extension sul CPAN
  - http://search.cpan.org/~ebruni/HTML-Template-Extension-0.24/
- Il mio sito
  - http://www.ebruni.it