Licence d'informatique
Module de C/Unix

Projet Unix

ptml — Préprocesseur HTML

Philippe MARQUET

Octobre 1999

La commande `ptml` est inspirée de la commande `gtml` développée initialement par Gihan PERERA et maintenant par Bruno BEAUFILS.

1 Pourquoi ptml ?

HTML est un langage relativement puissant de mise en forme de pages Web. Quelques informations et références sur HTML sont données en annexe A de ce document. Une des limitations majeures d'HTML est la difficulté de maintenir un ensemble de pages de manière cohérente. Les outils tels `gtml`, ou la commande `ptml` que vous allez écrire, ajoutent des fonctionnalités au langage HTML et facilitent par exemple la mise en place d'un ¡¡ look'n fell ¿¿ unique pour un ensemble de pages.

De nombreux documents HTML contiennent des informations redondantes supposées être cohérentes pour un ensemble de documents : des barres de navigation, une couleur ou une image de fond, des haut ou bas de page, un logo... Une approche est d'identifier ces informations et de factoriser leur mise à jour par l'utilisation d'un préprocesseur.

Par exemple, une simple macro permet de gérer un bas de page standard. Ce bas de page pourra contenir des icônes de navigations, des renseignements sur l'auteur... En modifiant la définition de cette macro, on pourra changer d'un seul coup la présentation de tout un ensemble de documents HTML. De même, on pourra définir une fois pour toutes des macros pour les URL souvent utilisées.

Le langage PTML est une extension du langage HTML. `ptml` est un préprocesseur de code HTML : il transforme du code PTML en code HTML.

Les fonctionnalités de base de `ptml` sont :
- l'inclusion de fichiers ;
- la définition et l'expansion de macros ;
- la production conditionnelle de texte.

`ptml` est donc un préprocesseur pour le langage HTML comme `cpp` est un préprocesseur pour le langage C.

2 Les commandes PTML

Les commandes PTML sont les suivantes :

`include` permet l'inclusion d'un autre fichier source.

`define` crée une nouvelle macro qui sera utilisé comme un raccourci.

`undef` supprime la définition d'une macro.

`ifdef/ifndef/endif` permettent la production conditionnelle de code HTML.

Une commande PTML doit apparaître seule sur une ligne et être précédée d'un caractère `#`.

Id : ptml.tex,v 1.9 2000/12/03 10 :47 :49 marquet Exp

2.1 La commande #include

Un texte commun à plusieurs fichiers peut être placé dans un fichier commun (par exemple `common.ptml`) et inclus dans chacun des fichiers par la commande

```
#include "common.ptml"
```

`ptml` remplace cette ligne par le contenu du fichier `common.ptml`.

Ce fichier peut contenir d'autres commandes PTML.

Le nom du fichier peut être contenu dans une macro PTML (voir ci-dessous).

2.2 La commande #define et l'utilisation des macros

La commande `#define` permet de définir des macros qui sont des noms auxquels sont associés une valeur. Par exemple

```
#define MY_EMAIL phm@lifl.fr
```

définit la macro `MY_EMAIL` dont la valeur est ¡¡ `phm@lifl.fr` ¿¿.

L'utilisation de cette macro, c'est-à-dire la substitution de son nom par sa valeur, est spécifiée dans le texte PTML par l'utilisation de chevrons :

```
<<MY_EMAIL>>
```

D'autres choses sont possibles avec les macros :

- la valeur de macros peut être définie sur la ligne de commande de `ptml` (voir section 4) ;
- il existe un certain nombre de macros prédéfinies (voir section 5) ;
- il est parfaitement légal de trouver plusieurs `#define` pour une même macro. Il est ainsi possible de changer la valeur d'une macro lors du traitement de différentes parties de vos documents ;
- la valeur d'une macro peut elle même contenir le nom d'une autre macro, par exemple :

  ```
  #define MAILTO_ME <a href="mailto:<<MY_EMAIL>>">
  ```

  Notez que les définitions imbriquées sont évaluées lors de leur utilisation et non lors de leur définition. Ainsi, sur l'exemple, il est possible de modifier la valeur de la macro `MY_EMAIL` après la définition de la macro `MAILTO_ME`.
- la valeur d'une macro non définie est la chaîne vide.

La commande `#undef` supprime la définition d'une macro :

```
#undef MY_EMAIL
```

Si la macro n'existe pas préalablement, la commande ne fait rien.

2.3 Les commandes #ifdef, #ifndef, et #endif

Ces commandes permettent de traiter ou non un bloc de lignes PTML selon qu'une macro PTML est définie ou non. Par exemple,

```
#ifdef FIL_URL
<A HREF="<<FIL_URL>>">Formations en informatique de Lille</A>
#endif
#ifndef FIL_URL
Formations en informatique de Lille
#endif
```

Les constructions `#if*def/#endif` peuvent être imbriquées.

Le `#endif` correspondant à un `#if*def` doit être dans le même fichier source PTML.

3 Fichier de projet

La commande `ptml` est particulièrement utilisée pour la gestion de plusieurs fichiers PTML. Pour faciliter cette gestion, `ptml` propose la notion de *fichier de projet* (ou *ptml project file*, *ppf*). Ce sont des fichiers d'extension `.ppf` qui contiennent

– des commandes PTML (`#define`...);
– des lignes de commentaires. Ce sont des lignes qui débutent par les caractères `//`. Ces lignes sont ignorées. Les lignes vides sont aussi ignorées;
– des noms de fichier PTML. Ces fichiers peuvent se trouver sous le répertoire courant, leur nom sera alors donné relativement au répertoire courant.

L'exécution de `ptml` avec un nom de fichier de projet en argument va traiter les uns après les autres les fichiers PTML du projet. Chacun de ces traitements hérite des `#define` et autres commandes PTML définies dans le fichier de projet.

4 Syntaxe et options de la commande `ptml`

La syntaxe de la commande `ptml` est la suivante :

```
ptml [options] [ptml-file...|ppf-file...]
```

ainsi on peut traiter un ou plusieurs fichiers PTML, ou un ou plusieurs fichiers projet.

Les options sont les suivantes :

-D Permet de définir des macros comme ceci :

```
-DMY_EMAIL=phm@lifl.fr
```

Ce qui est équivalent à un `#define` dans le fichier.

--verbose `ptml` indique l'état d'avancement des traitements au cours de son exécution.

--silent Rend l'exécution silencieuse.

--help Affiche un message d'usage de la commande sur la sortie standard et termine sur un succès.

--version Affiche les informations de version sur la sortie standard et termine sur un succès.

L'exécution de `ptml` génère un fichier `.html` [1] pour chacun des fichiers PTML traités.

5 Macros spéciales

5.1 Nom du fichier généré

La macro `ROOT_PATH` a pour valeur le chemin relatif du répertoire courant au répertoire racine, c'est-à-dire au répertoire dans lequel réside le fichier de projets.

La macro `FILENAME` a pour valeur le nom du fichier HTML produit, sans extension et sans information de répertoire.

5.2 Chemin de recherche pour les fichiers inclus

`ptml` recherche les fichiers inclus dans le répertoire contenant le fichier traité puis dans le répertoire courant (c'est-à-dire le répertoire dans lequel la commande `ptml` a été lancée).

La macro `INCLUDE_PATH` peut être définie. `ptml` interprète cette macro comme une liste de répertoires (séparés par des deux-points `:`) dans lesquels les fichiers inclus sont recherchés. Ces répertoires peuvent être absolus ou relatif au fichier de projet.

[1] Cette extension peut être changée, voir la macro `EXTENSION`, section 5.

5.3 Répertoire de production

Par défaut, `ptml` produit ses fichiers résultats dans le même répertoire que les fichiers sources. Il est possible de préciser le nom d'un répertoire dans lequel produire les résultats en définissant la macro `OUTPUT_DIR`.

Dans le cas de l'utilisation d'un fichier de projet, `OUTPUT_DIR` est la racine des fichiers produits. Une hiérarchie équivalente à celle des fichiers PTML est créée pour les fichiers résultats.

5.4 Suffixe des fichiers résultats

Par défaut, les fichiers créés portent l'extension `.html` au lieu de l'extension `.ptml` des fichiers sources. La valeur de la macro `EXTENSION` permet de choisir l'extension que porteront les fichiers produits.

Notez que la définition de cette macro dans un fichier PTML n'a pas de sens, le fichier résultat ayant déja été créé. La définition de cette macro devrait normalement se trouver dans un fichier de projet ou sur la ligne de commande.

6 Exemples et tests

Un exemple de source PTML et sa traduction ainsi que des jeux de tests sont disponibles sous forme d'une archive (fichier tar comppressé) : `http://www.lifl.fr/~marquet/ens/cu/ptml/ptml-xampl.tgz`.

A Le langage HTML

HTML est un langage de composition de documents/pages destinés à être visualisés par un navigateur/butineur. Un ensemble de *balises* déterminent la présentation et l'organisation de la page. Un élément HTML se compose

- de la balise ouvrante (de la forme `<tag>`) ; cette balise ouvrante peut comporter des attributs et leur valeur ;
- d'un texte ;
- de la balise fermante (de la forme `</tag>`).

Un élément prend donc l'une des formes suivantes :

```
<tag_name> text </tag_name>
<tag_name attribute_nalme=argument> text </tag_name>
```

Par exemple,

```
<TITLE>ptml --- Préproceseur HTML</TITLE>
```

indique le titre du document alors que

```
ceci est <em>important</em>
```

précise que cette partie du texte doit être mise en valeur (*emphasis* en anglais) ; elle sera par exemple présentée en italique ou dans une couleur donnée.

En particulier l'élément

```
<a href="URL">text</a>
```

définit un *hyperlien* vers un autre document (celui référencé par `URL`).

Un bon moyen d'apprentissage/compréhension du langage HTML est de regarder les sources de ¡¡ bonnes ¿¿ pages que vous trouverez sur le Web. (Les navigateurs proposent tous une fonction ¡¡ afficher le source ¿¿ ou ¡¡ view page source ¿¿.)

De nombreuses informations sur HTML sont disponibles sur le Web :

- Daniel Boivin et Laurent Gauthier de l'université de Laval au Québec proposent leur ¡¡ Manuel illustré de programmation en HTML ¿¿.
  `http://www.grr.ulaval.ca/grrwww/manuelhtml.html`

- Les ¡¡ Academic Computing Services ¿¿ de l'université du Kansas proposent une brève introduction à HTML (en anglais).
  `http://www.cc.ukans.edu/~acs/docs/other/HTML_quick.shtml`
- La NCSA de l'université d'Illinois à Urbana-Champaign propose une autre introduction à HTML (en anglais).
  `http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html`
- Éric Tilton donne quelques conseils de style pour la rédaction de documents HTML (en anglais).
  `http://www.cs.cmu.edu/~tilt/cgh/`

L'ensemble de ces références est accessible ¡¡ d'un simple clic ¿¿ à partir de la version en ligne de ce sujet : `http://www.lifl.fr/~marquet/ens/cu/ptml/`.