Le système de document Bocal

Tutorial

Edouard Thiel

1er décembre 2008

Table des matières

Chapitre 1

Structure d'un document

1.1 Compilation

Le langage Boc est un langage à balises dédié à la rédaction de documentation de logiciels. Le compilateur s'appelle `bocal`, et il est capable de générer du Html et du Latex. Le source et l'exécutable du compilateur sont situés dans le répertoire `src`.

Le compilateur est écrit en lex, et fonctionne en plusieurs passes : il y a toujours une passe initiale, où bocal mémorise la structure logique du document (titres), les labels des titres, les macros, etc. Puis il y a une passe pour générer le Html, et/ou une passe pour générer le Latex.

La syntaxe pour appeler bocal est :

```
bocal <filein.boc> [--index] [--macro]
    [--html <fileout.html>] [--latex <fileout.tex>]
```

L'option `--index` affiche dans la console la table des matières et les labels des titres ; l'option `--macro` affiche le contenu des macros en mémoire.

1.2 Document principal

Le document principal commence par `<bocal>` et finit par `</bocal>`.

Le corps du document est composé de texte, entrecoupé de sections qui lui donnent sa structure logique.

Le texte est une suite de caractères et de balises Boc délimités par < et >.

1.3 Sections

Il y a quatre niveaux logiques de section, numérotés de 1 à 4 ; ils correspondent en Latex à chapter, section, subsection et subsubsection.

On commence un chapitre par `<t1>Titre du chapitre</t1>`, une section par `<t2>Titre de la section</t2>`, etc.

La numérotation des sections est automatique, et chaque section produit automatiquement une entrée dans la table des matières.

1.4 Fichiers inclus

On peut inclure des fichiers Boc par la commande : `<include "fichier.boc">`. Le découpage du document en fichiers Boc n'a rien à voir avec la structure logique du document, qui est établie à partir des sections.

1.5 Table des matières

Pour faire apparaître la table des matières complète, on fait `<tabmat long "toc-long" "Table des matières">` ; pour avoir l'index des chapitres on fait `<tabmat short "toc-short" "Index rapide">`.

Les chaînes `"toc-long"` et `"toc-short"` sont utilisés pour les références (voir 6.4).

1.6 Fichier résultat unique ou multiple

Après compilation, le résultat en Latex est toujours un fichier unique.

En Html on a la possibilité de demander un fichier unique ou un fichier par chapitre. Par défaut le compilateur bocal est en mode fichier unique ; pour passer en mode fichiers multiples il suffit d'insérer quelque part la balise `<split-html>`.

Chapitre 2

Le texte

2.1 Paragraphes

Comme en html, on termine une ligne par un `<br>` et un paragraphe par `<p>`.

2.2 Attributs

Les balises suivantes permettent de mettre le texte en `<b>`**gras**`</b>`, en `<i>`*italique*`</i>` ou en `<c>`code`</c>`.

2.3 Indice ou exposant

On met le texte en $_{\text{indice}}$ ou en $^{\text{exposant}}$ en tapant `<sub>indice</sub>` ou `<sup>exposant</sup>`.

2.4 Texte centré

On centre le texte en tapant `<center>texte centré</center>`, ce qui donne :

texte centré

2.5 Espace

On génère un espace insécable avec `<nbsp>`.

2.6 Césure

On rajoute une possibilité de césure dans un mot avec la balise `<->`. Cette commande est sans effet en Html ; elle consiste à rajouter un `\-` en Latex.

Si un paragraphe déborde et qu'on n'a pas la possibilité de rajouter des césures, on peut mettre la balise `<sloppy>` au début du paragraphe. Cette commande est sans effet en Html ; elle consiste à rajouter un `\sloppy` en Latex.

2.7 Lettres accentuées

Les lettres accentuées sont automatiquement traduites : àâä éèêë îï ôö ùûü ÀÂÄ ÉÈÊË ÎÏ ÔÖ ÙÛÜ

2.8 Caractères spéciaux

Les seuls caractères spéciaux du langage Boc sont < , > et \ . Pour les produire il faut taper \< , \> ou \\ .

2.9 Symboles

Les autres caractères sont automatiquement traduits, par exemple : $ & % # _ { } ˜ .

2.10 Commentaires

On peut mettre du texte en commentaire en le plaçant entre `<!` et `>`. Par exemple, `<!ceci est un commentaire>`. Les commentaires n'apparaissent pas en Html ni en Latex.

Chapitre 3

Les environnements

3.1 Verbatim

Pour insérer du texte non interprété contenant par exemple des balises Boc, on encadre le texte entre <verb> et </verb>. Le résultat sera traduit en verbatim Html ou Latex.

Par exemple, <verb>du texte <i>italique</i></verb>
produit : du texte <i>italique</i>.

3.2 Code Html

On peut insérer du code Html, qui ne figurera que dans le fichier Html produit ; il suffit de l'encadrer entre <lang html> et </lang>.

Par exemple, <lang html>du texte italique</lang>
produit : .

3.3 Code Latex

On peut insérer du code Latex, qui ne figurera que dans le fichier Latex produit ; il suffit de l'encadrer entre <lang latex> et </lang>.

Par exemple, <lang latex>du texte {\em italique}</lang>
produit : du texte *italique*.

3.4 Le test if

L'environnement if permet d'insérer du texte qui n'apparaît que dans un type de fichier précis, voire n'apparaît pas du tout : le texte encadré entre <if> et </if> n'apparaît pas (c'est une façon de commenter du texte) ; entre <if html> et </if> le texte n'apparaît qu'en mode Html ; entre <if latex> et </if> le texte n'apparaît qu'en mode Latex.

Attention, pour le moment on ne peut pas emboiter plusieurs if.

3.5 Listes

On peut créer deux types de listes : les listes numérotées et les listes non numérotées. Une liste numérotée est délimitée par les balises <ol> et . Une liste non numérotée est délimitée par les balises <ul> et . Chaque entrée d'une liste commence par <li>. Par exemple la liste ordonnée :

1. Oranges ;
2. pêches.

est obtenue avec le code

```
<ol>
  <li> Oranges ;
  <li> pêches.
</ol>
```

tandis que la liste non ordonnée :

▷ Pommes ;
▷ poires.

est obtenue avec le code

```
<ul>
  <li> Pommes ;
  <li> poires.
</ul>
```

3.6 Code préformaté

On peut faire apparaître un bout de code préformatté, qui apparaît en police code avec un espacement fixe, et un fond coloré ; il suffit d'encadrer le paragraphe entre <codepre> et </codepre>. Par exemple :

```
<codepre>
void Exemple1 (int i, char *s)
{
printf ("i = %d , s = %s\n", i, s);
}
</codepre>
```

affiche :

```
void Exemple1 (int i, char *s)
{
    printf ("i = %d , s = %s\n", i, s);
}
```

Ceci correspond au style par défaut <codepre "codedef">. Les autres styles sont :

<codepre "codeusage">

```
void Exemple1 (int i, char *s)
{
    printf ("i = %d , s = %s\n", i, s);
}
```

<codepre "codeterm">

```
void Exemple1 (int i, char *s)
{
    printf ("i = %d , s = %s\n", i, s);
}
```

<codepre "codealert">

```
void Exemple1 (int i, char *s)
{
    printf ("i = %d , s = %s\n", i, s);
}
```

3.7 Listing

Pour faire apparaître un listing avec un habillage coloré, il suffit de mettre <filec "chemin/fichier.c">.

Par exemple, voici ce que donne <filec "hello.c"> :

```
#include <stdio.h>
#include <stdlib.h>

int main (int argc, char *argv[])
{
    printf ("Hello World!\n");
    exit (0);
}
```

On peut aussi donner le numéro de ligne de départ, ou encore le numéro de ligne de départ et de fin, avec <filec "chemin/fichier.c" "depart"> ou <filec "chemin/fichier.c" "depart" "fin">.

Chapitre 4

Labels, références et liens

4.1 Labels

On peut donner un label à une section, comme en Latex, puis y faire référence dans le document. Pour donner un label à une section on fait `<t1 "label-du-chapitre">Titre du chapitre</t1>`, ou `<t2 "label-de-section">Titre de la section</t2>`, etc. Le label doit être une chaîne sans blancs. Si un label n'est pas donné, bocal donne un label par défaut pour que les mécanismes internes fonctionnent.

Pour voir tous les labels d'un document on appelle bocal avec l'option `--index`.

4.2 Références

Pour insérer dans le texte le numéro correspondant au label d'une section, il suffit de faire `<refn "label-de-section">`. De même on peut insérer le titre d'une section avec « `<reft "label-de-section">` » (les guillemets sont purement esthétiques).

4.3 Liens

Comme en Html, on insère un lien sur du texte par : `<a href="mon-url">texte souligné</a>`. En Latex, le texte apparaît sans aucune autre indication.

On peut aussi rajouter une ancre (invisible) : `<a name="mon-ancre">texte ancré</a>`.

4.4 Liens muets

Une écriture équivalente de `<a href="mon-url">Texte</a>` est `<a-url-str "mon-url" "Texte">` : le résultat est identique si `"mon-url"` n'est pas vide. Dans le cas inverse, le texte apparaît mais non souligné. Cette fonctionnalité est exploitée pour afficher « Précédent » ou « Suivant » avec ou sans lien si on est en début/fin de document ou non.

4.5 Liens sur labels

On peut mettre un lien sur un numéro de section généré par une référence, ou sur un titre, ou sur les deux :
`<a-refn "label-section">` produit le numéro,
`<a-reft "label-section">` produit le titre,
`<a-refnt "label-section">` produit le numéro + titre,
`<a-ref-str "label-section" "autre texte">` produit un lien avec un autre texte.

Chapitre 5

Les variables

5.1 Principe

Les variables de Boc ont la forme `$identificateur` ; on les place dans des balises, et elles sont substituées par leur valeur au moment de la compilation du document Boc.

5.2 Variables prédéfinies

Les variables prédéfinies sont `$thecur`, `$thenext`, `$theprev`, `$thecurt1`, `$thenextt1`, `$theprevt1`, `$thecurt2`, `$thenextt2`, `$theprevt2`. Chacune de ces variable est remplacée par le label d'une partie ; respectivement de la partie courante, suivante ou précédente, du chapitre courant, suivant ou précédent, de la section courante, suivante ou précédente.
On n'a pas la possibilité pour le moment de définir d'autres variables.

5.3 Utilisation

On se sert des variables de la famille `$thecur` avec les commandes de la famille <`ref`> et <`a-ref`> :

<`refn $thecur`> donne 5.3,
« <`reft $thecur`> » donne « Utilisation » ;
<`a-refn $theprev`> donne 5.2 ,
« <`a-reft $theprev`> » donne « Variables prédéfinies »,
« <`a-refnt $theprev`> » donne « 5.2 Variables prédéfinies » ;
<`a-ref-str $thenext "section suivante"`> donne section suivante.

La commande `echo` a été introduite pour afficher dans le texte la valeur des arguments de la balise, à des fins de débogage. Par exemple <`echo $thecur $thenext`> donne sec-var-util sec-var-lienmu .

5.4 Liens muets

Les variables de la famille `$thecur` sont substituées par la chaîne vide si on est en extrémité de document (il n'y a pas de précédent ou de suivant par exemple). Comme indiqué dans la section 4.4 , les commandes de la famille <`a-ref`> produisent du texte *non* souligné si le label est vide, ce qui est le but recherché.

Chapitre 6

Les macros

6.1 Principe

Les macros de Boc sont des blocs de code Boc mémorisé en mémoire et accessibles par une clé.

Pour mémoriser une macro, on fait <macro "cle-de-macro">Corps de la macro</macro> et pour replacer le corps de la macro dans le texte on fait <expand "cle-de-macro">.

6.2 Compilation

Le compilateur bocal mémorise toutes les macros lors de la passe initiale. Si on mémorise une macro avec une clé déjà utilisée, la nouvelle macro remplace l'ancienne. Pour afficher dans la console la liste des macros à l'issue de la première passe, appeler bocal avec l'option --macro.

Lors de la deuxième passe, les macros sont replacées littéralement dans le texte, si bien qu'on peut très bien mettre un <expand "cle2"> dans un bloc <macro "cle1"> ... </macro>. Attention toutefois à ne pas provoquer d'imbrication en boucle, actuellement aucun test n'est fait dans bocal pour le détecter.

6.3 Macros prédéfinies

Un certain nombre de macros sont prédéfinies ; il suffit de les redéfinir pour surcharger leur contenu :

- ▷ "title" le titre du document.
- ▷ "body-html" définition des couleurs de fond et de liens en Html.
- ▷ "begin-doc-html" le début du document Html ; utilise les macros "title" et "body-html".
- ▷ "end-doc-html" la fin du document Html.

- ▷ "begin-split-html" le début de chaque sous-fichier en Html ; utilise les macros "title" et "body-html".
- ▷ "end-split-html" la fin de chaque sous-fichier en Html.
- ▷ "begin-t1-html" inséré juste avant un titre de chapitre en Html.
- ▷ "end-t1-html" inséré juste avant la fin d'un chapitre en Html.
- ▷ "begin-t2-html" inséré juste avant un titre de section en Html.
- ▷ "begin-doc-latex" le début du document Latex.
- ▷ "end-doc-latex" la fin du document Latex.
- ▷ "begin-t1-latex" inséré juste après un titre de chapitre en Latex.

6.4 Exemples

Pour générer dans ce document la barre « Index Table Précédent Suivant » au début de chaque chapitre on a redéfini la macro "begin-t1-html" :

```
<macro "begin-t1-html">
  <center>
  <a-url-str "#toc-short" "Index"><nbsp>
  <a-url-str "#toc-long" "Table"><nbsp>
  <a-ref-str $theprevt1 "Précédent"><nbsp>
  <a-ref-str $thenextt1 "Suivant">
  </center><lang html><hr></lang>
</macro>
```

Le source Boc de ce document est situé dans le répertoire doc/bocal-tut.boc. Il peut être intéressant à consulter, en particulier pour l'écriture de la première page, qui n'est pas pour le moment écrite sous forme de macros.