Université Paris Diderot
L2 Informatique et Math-Info

Analyse de Données Structurées
Année 2016-2017

Projet

Un interpréteur pour l'édition de texte en HTML

1 But du projet

Le but du projet est de permettre la traduction d'un texte écrit dans le langage `SimpleText` (inspiré du LaTeX) en HTML.

2 Travail demandé

Écrivez un analyseur lexical et syntaxique afin de construire l'arbre de syntaxe abstraite d'un document décrit par la grammaire ci-dessous. Ensuite, écrivez un interpréteur pour l'arbre construit qui va produire la traduction du document lu vers HTML. Il est conseillé d'écrire le programme en plusieurs étapes de difficulté croissante.

HTML Si vous ne connaissez pas ou vous ne vous souvenez pas du langage HTML, vous pouvez consulter les transparents du cours : "Internet et Outils" sur Moodle (répertoire `Science/UFR d'informatique/L1/IO2`). Pour ce projet il n'est pas nécessaire d'utiliser CSS, mais vous êtes libre de le faire si vous le trouvez utile.

Option debug de jflex Pour vous aider à debuguer la partie d'analyse lexicale, il existe une option `%debug` (à mettre au même endroit que `%standalone`) qui crée un `main` qui vous indiquera les tokens trouvés et d'autres informations. (cf. manuel de JFlex)

Gestion des erreurs Votre programme doit traiter proprement toutes les erreurs qui peuvent se produire pendant les analyses lexicales et syntaxiques ainsi que lors de l'interprétation, c'est-à-dire afficher un message explicite, détaillant l'erreur rencontrée et arrêter le processus en cours.

3 Rendu

Il est conseillé de faire le projet en binôme (mais pas de trinômes !). Le projet est à rendre avant le lundi 13 mai, 20h00, sur Moodle. Vous devrez rendre une archive contenant tous les fichiers de votre projet ainsi qu'un court fichier texte : celui-ci indiquera les parties qui ont été faites, et parmi celles-ci ce qui ne fonctionne pas et contiendra une explication indiquant comment compiler et exécuter votre projet. Les modalités exactes du rendu sur Moodle vous seront communiquées ultérieurement.

Il est bien entendu que votre projet doit être fait intégralement par vous et que vous devez être capable d'expliquer non seulement la partie de code que vous avez écrite mais aussi celle écrite par votre binôme. Tout plagiat sera lourdement sanctionné.

Il vous est vivement conseillé de sauvegarder fréquemment et sur plusieurs supports les différentes versions de votre programme et, plus particulièrement, toute version même incomplète qui fonctionne.

4 Première étape

La grammaire La grammaire du langage est la suivante, les terminaux sont en **gras**, les terminaux en MAJUSCULES, les accolades sont des terminaux. Les backslash font partie des terminaux `\begindoc`, `\enddoc`, etc.

DOCUMENT	→	CORPS
CORPS	→	`\begindoc` SUITE_ELEMENTS `\enddoc`
SUITE_ELEMENTS	→	ELEMENT SUITE_ELEMENTS | ε
ELEMENT	→	MOT
		| `\linebreak`
		| `\bf{` SUITE_ELEMENTS `}`
		| `\it{` SUITE_ELEMENTS `}`
		| ENUMERATION
ENUMERATION	→	`\beginenum` SUITE_ITEMS `\endenum`
SUITE_ITEMS	→	ITEM SUITE_ITEMS | ε
ITEM	→	`\item` SUITE_ELEMENTS

Le non-terminal MOT correspond à toute suite de symboles alphanumériques, ponctuation classique comprise ; pour des raisons évidentes le caractère backslash ne pourra pas être utilisé. Il n'est pas nécessaire de prendre en compte les caractères accentués. Les éléments d'une suite d'éléments peuvent être séparés par des espaces.

Sémantique du langage Le corps du document correspond au texte proprement dit :
- les mots sont les mots du texte.
- `\linebreak` correspond à un passage à la ligne.
- `\bf{` *suite_elements* `}` veut dire que le texte entre accolades sera en gras (bold face).
- `\it{` *suite_elements* `}` veut dire que le texte entre accolades sera en italique.

Les ITEM d'une énumération sont écrits en retrait et numéroté (cf. exemple ci-dessous).

4.1 Exemple :

Le code de la figure 1 devra, une fois traduit en HTML et lu sur un navigateur, avoir un rendu similaire à celui de la figure 2, un code HTML possible peut être celui de la figure 3. (les passages à la lignes ne servant qu'à une meilleure compréhension du code).

5 Deuxième étape : la couleur

On veut ajouter de la couleur au texte ; pour faire cela, on va ajouter un certain nombre de règles.

```
\begindoc
\bf{Systeme RGB}
\linebreak
Le systeme \bf{RGB} utilise en informatique distingue 3 couleurs \it{primaires}:
\beginenum
\item \it{\bf{rouge}} code en hexadecimal \bf{FF0000};
\item \it{\bf{vert}} code \bf{00FF00};
\item \it{\bf{bleu}} code \bf{0000FF};
\endenum
Toute autre couleur etant codee par une combinaison de ces trois couleurs.
\linebreak
\linebreak
\bf{Utilisation en Java}
\enddoc
```

FIGURE 1 – code en SimpleText

Le systeme **RGB** utilise en informatique distingue 3 couleurs *primaires*:

1. ***rouge*** code en hexadecimal **FF0000**;
2. ***vert*** code **00FF00**;
3. ***bleu*** code **0000FF**;

Toute autre couleur etant codee par une combinaison de ces trois couleurs.

Utilisation en Java

FIGURE 2 – rendu de l'exemple

$$
\begin{aligned}
\text{DOCUMENT} &\rightarrow \text{DECLARATIONS CORPS} \\
\text{DECLARATIONS} &\rightarrow \texttt{\textbackslash set\{ } \text{ID} \texttt{ \} \{ } \text{VALEUR} \texttt{ \} } \text{DECLARATIONS} \mid \varepsilon \\
\text{ELEMENT} &\rightarrow \mid \texttt{\textbackslash couleur\{ } \text{VAL\_COL} \texttt{ \} \{} \text{SUITE\_ ELEMENTS} \texttt{\}} \\
\text{VAL\_COL} &\rightarrow \texttt{\textbackslash constante\_couleur} \mid \text{ID}
\end{aligned}
$$

La nouvelle règle pour ELEMENT montrée au-dessus s'ajoute aux règles pour ELEMENTS donnée à l'étape 1. Le non-terminal CONSTANTE_COULEUR correspond à une couleur donnée par 6 chiffres hexadécimaux, FF34A1 par exemple.

La partie DECLARATIONS permet de définir des constantes de couleurs.

`\couleur{` x `}` `{` SUITE_ ELEMENTS `}` veut dire que le texte entre accolades prendra la couleur x.

```
<!DOCTYPE html>
<html>
<body>
Le systeme <b>RGB</b> utilise en informatique distingue 3 couleurs <i>primaires</i>:
<ol>
<li> <i><b>rouge</b></i> code en hexadecimal <b>FF0000</b>;</li>
<li> <i><b>vert</b></i> code <b>00FF00</b>;</li>
<li> <i><b>bleu</b></i> code <b>0000FF</b>;</li>
</ol>
Toute autre couleur etant codee par une combinaison de ces trois couleurs.
<br><br>
<b>Utilisation en Java</b>
</body>
</html>
```

FIGURE 3 – code html de l'exemple

6 Troisième étape : les abréviations

On veut maintenant ajouter des "abréviations" : par exemple, au lieu d'écrire **Organisation des Nations Unies** en gras, on peut avoir envie d'écrire `\onu`. Pour faire ça, on va mettre dans la partie DECLARATIONS `\abb{\onu}{\bf{`Organisation des Nations Unies`}}`.

Modifier la grammaire pour que ce soit possible et ajouter cette fonctionnalité à votre programme.

7 Étape bonus

S'il vous reste du temps, vous pouvez ajouter des boucles qui permettront de répéter plusieurs fois la même partie de texte. Vous pouvez aussi ajouter des instructions conditionnelles qui, par exemple, testeront la valeur d'une constante booléenne déclarée dans la partie déclaration. Vous êtes aussi libre de choisir d'autres types d'extensions.