Choisir un « pastebin »

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 17 mars 2010. Dernière mise à jour le 10 septembre 2010

https://www.bortzmeyer.org/pastebin.html

Il y a davantage de « "*pastebins*" » (ces applications Web qui permettent d'enregistrer temporairement une partie d'un programme qu'on veut communiquer facilement mais qui est trop long pour des supports comme Twitter) sur l'Internet que de distributions Linux (ou de VCS). Écrire un "*pastebin*" est même souvent un exercice de choix pour le débutant qui essaie un nouvel environnement de programmation Web. Résultat, je ne sais plus lequel choisir lorsque je dois parler d'un petit bout de code sur IRC ou XMPP.

Alors, j'en ai comparé quelques uns. J'ai fait pour cela quatre petits textes, un dans un langage de programmation très classique, C :

```
CURL *curl;
char errbuf[1024];

curl = curl_easy_init();
if (! curl) {
  fprintf(stderr, "Cannot initialize curl\n");
  return 1;
}
curl_easy_setopt(curl, CURLOPT_ERRORBUFFER, errbuf);
```

un dans un langage nettement moins connu, Haskell :

```
checkDcls :: Program -> Result
checkDcls l =
     let t = Map.fromList (map extract
                                (filter is_declaration l)) in
             foldl combine (True, Nothing) (map (test_declaration t)
                  (filter is_transition l))
```

un exemple d'un fichier de configuration (ici, du Apache) :

```
WSGIDaemonProcess bortzmeyer.org processes=3 threads=10 display-name=%{GROUP}
WSGIProcessGroup bortzmeyer.org
<Directory /var/www/www.bortzmeyer.org/wsgis>
    Options -Indexes -Multiviews +ExecCGI
    Order allow,deny
    Allow from all
</Directory>
```

et enfin un exemple d'exécution d'une commande Unix :

```
% git log
commit 4d10b9c920c4925d93df760a042395a23261be0e
Author: Stephane Bortzmeyer <stephane+github@bortzmeyer.org>
Date:   Sat Mar 13 13:15:13 2010 +0100

    Better description of the reflector
```

Je savais qu'il y avait un choix pour les langages de programmation <https://www.bortzmeyer.org/choix-langage-prog.html>, les éditeurs, les VCS, les partis politiques (sauf si on se limite au PS, au Modem et à l'UMP) et les boissons servies au café. Dans tous les cas, je vois à peu près les différences. Et je peux donc faire un choix. Mais pour les "*pastebin*" ? Ils sont vraiment si différents que cela ? Essayons successivement ces extraits de codes sur plusieurs "*pastebins*" (en sachant qu'on n'arrivera jamais à les utiliser tous <http://del.icio.us/bortzmeyer/pastebin>).

Commençons par Copy Paste Code <http://www.copypastecode.com/>. On peut l'utiliser sans avoir de compte mais en avoir un permet de gérer ses extraits, les supprimer, etc (je n'ai pas testé). Copy Paste Code permet de colorier le code selon le langage (qu'il faut indiquer explicitement, le choix est vaste mais Haskell n'y figure pas, ni les langages de configuration comme celui d'Apache). Le résultat, pour C, est en effet agréable (le code Haskell peut être enregistré mais il n'est pas colorié). Copy Paste Code ne met pas automatiquement les numéros de ligne, ce qui serait pratique.

J'essaie ensuite <http://paste.factorcode.org/>. Le choix des langages connus est **beaucoup** plus vaste et inclus cette fois Haskell, ainsi que la configuration Apache. La colorisation pour C est différente de celle de Copy Paste Code mais tout aussi bonne. Celle d'Haskell est très limitée mais c'est cohérent avec le faible nombre de mots-clés de ce langage. Il ne semble pas y avoir d'option « non formaté » (par exemple pour ma commande Unix et son résultat) et le formatage par défaut, pour Factor, donne des mauvais résultats. Enfin, pour écrire un extrait, le CAPTCHA est très pénible. Et il ne semble pas possible de se créer un compte pour pouvoir en être dispensé. Bref, c'est un de ceux que j'ai le moins aimé.

Et le site de référence, <http://pastebin.com/> ? Il souffre de publicités tapageuses. Par contre, son choix de langages est le plus vaste. Le coloriage d'Haskell tient compte des fonctions qui, sans être des mots-clés, sont tellement répandues qu'elles méritent une mise en valeur (comme `fold` ou `map`, que `http://paste.factorcode.org/` ignore). Comme rien n'est parfait, le langage de configuration d'Apache n'est cette fois pas connu, j'ai utilisé l'option « Pas de formatage ». Et j'apprécie beaucoup la numérotation automatique des lignes, très pratique lorsqu'on demande de l'aide dans une pièce XMPP « Quelqu'un voit pourquoi la ligne 4 est refusée ? ». D'autre part, `pastebin.com` permet facilement d'indiquer un extrait de code comme privé (il n'apparait pas dans la liste mais ceux à qui on donne l'URL peuvent le voir). On peut en outre sans formalité créer un espace à soi, accessible par `http://LE-NOM-QUE-JE-CHOISIS.pastebin.com/`. Et `pastebin.com` a une API <http://pastebin.com/api.php>, ce qui est rare (regardez, par exemple, ce code Python pour y accéder

<http://breakingcode.wordpress.com/2010/03/06/using-the-pastebin-api-with-python/>).

<http://paste.lisp.org> permet d'afficher l'URL directement dans un canal IRC sur Freenode ou via Twitter. Sa liste de langages reconnus est très courte (mais inclus Haskell). Il a la possibilité (ce n'est pas fait par défaut, sans doute parce que, pour copier-coller le code, cela peut être un problème) d'indiquer les numéros de ligne. Il a lui aussi un CAPTCHA mais, contrairement à `http://paste.factorcode.org/`, il n'apparait que la première fois. Il a aussi une API, via XML-RPC <http://common-lisp.net/project/lisppaste/xml-rpc.html> (je ne l'ai pas testée).

Place maintenant à SourcePod.com <http://fr.sourcepod.com/> qui a l'avantage d'avoir une interface en français. Sa liste de langages reconnus est vaste mais la coloration syntaxique pour Haskell est complètement ratée (l'argument de `map` mis en rouge mais pas celui de `foldl`). De même, l'option « Pas de coloration syntaxique » colorie quand même ma configuration Apache ou mon exemple de commande Unix, et de manière plutôt bizarre.

Original pour sa mise en œuvre, <http://vpaste.net/> se sert de vim pour faire la coloration. Cela lui permet d'afficher une longue liste de langages reconnus (mais la coloration pour Apache a des bogues). Il se veut très simple et n'offre donc pas plein de gadgets. Mais il a une API REST.

Et, puisque le texte tapé dans les "*pastebin*" est souvent du code, pourquoi ne pas en profiter pour tester la compilation de ce code ? C'est ce que fait IDEone <http://ideone.com/> qui sert à la fois de "*pastebin*" traditionnel et de compilateur de test, pour un grand nombre de langages.

Sinon, on trouve des "*pastebin*" à des tas d'endroits. Par exemple, Github <https://www.bortzmeyer.org/github.html> en a un, nommé Gist <https://gist.github.com/>. Il faut un compte Github mais j'en ai un et que j'utilise donc, pour moi, ce n'est pas trop contraignant. Tous les extraits de code sont des dépôts git et sont donc versionnés. Regardez par exemple <http://gist.github.com/335753>. La liste des langages reconnus est très longue et c'est l'une des deux seules où j'ai trouvé le très récent Go (mais avec une coloration pas encore parfaite <http://gist.github.com/335765>).

Enfin, je n'ai pas eu le temps de tester <http://pastie.org/>.

Lequel faut-il utiliser ? C'est évidemment largement une question de goût. Personnellement, je me sers de Gist qui a toutes les fonctions dont j'ai besoin. Mais c'est aussi, en bonne partie, parce que j'utilise Github régulièrement. Vos goûts peuvent être différents.

Tiens, j'ai oublié de regarder lesquels distribuaient leur code source. Apparemment, `vpaste.net` le fait.

Merci à Samuel Tardieu, Ollivier Robert, lord et Mathieu Arnold pour leurs conseils et suggestions.