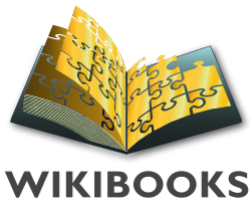
Un livre de Wikilivres.

Mkd (Extracteur de documents)

https://fr.wikibooks.org/wiki/Mkd(Extracteur_de_documents)



Éditeurs européens de logiciels libres

mkd, commande UNIX
Extracteur de documents

Une version à jour et éditable de ce livre est disponible sur Wikilivres,
une bibliothèque de livres pédagogiques, à l'URL :
http://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)

Vous avez la permission de copier, distribuer et/ou modifier ce
document selon les termes de la Licence de documentation libre
GNU, version 1.2 ou plus récente publiée par la Free Software
Foundation ; sans sections inaltérables, sans texte de première
page de couverture et sans Texte de dernière page de couverture.
Une copie de cette licence est inclue dans l'annexe nommée
« Licence de documentation libre GNU ».

Introduction

mkd est une commande Unix / Linux utilisée dans les fichiers de commandes et dans les Makefile pour maintenir à jour la documentation logicielle.

Des versions sont disponibles pour MASM / MS-DOS et pour Visual studio / MS-Windows.

Très simple et très pratique d'utilisation, mkd peut être proposé aux les travaux pratiques de l'informatique pour l'étude du *cycle en V*.

Sommaire


```
Cette version imprimable comprend les chapitres suivants :

  LA COMMANDE ET LES EXERCICES (Version imprimable) (Version pdf) 0
8/2016
    Première partie : DESCRIPTION
      La commande mkd
    Deuxième partie : EXERCICES
      Introduction aux exercices
      Comprendre les options
      Comprendre les fichiers de projets

  MAINTENANCES, MISES À JOUR, ÉVOLUTIONS
    Troisième partie : INTERNATIONALISATIONS
      Internationalisation des manuels
      Internationalisation des messages
      Quatrième partie : CODER, TESTER, INTEGRER, DOCUMENTER
      Ajouter des modules
    Cinquième partie : CONSTRUCTION D'UN PAQUET DEBIAN
      Exercice de Construction d'un Paquet 'debian' pour la mainten
ance logicielle
      Dépôt parrainé debian-ubuntu du paquet original mkd-131215
      Mkd (Extracteur de documents)/Dépôt parrainé du paquet ''orig
inal'' mkd-131215/Mise à jour de paquets
    Sixième partie : CONSTRUCTION D'UN PAQUET RED-HAT
      Paquets 'RPM' Red Hat Package Manager
  NOTES ET ANNEXES
    1.mkd-Manuel (fr)
    2.mkddocu-Manuel (fr)
    3.Compilations de wiki-livres
    4.QR pour pages imprimées
    5.Notes Icônes des fichiers et applications héritées
    6.Licences
```

Mkd (Extracteur de documents)
Première partie :
LA COMMANDE MKD

La commande mkd

Générer la documentation logicielle

Ce livre doit être rangé sur l'étagère Wikilivres Informatique
Systèmes d'exploitation, langages informatiques, logiciels CDU/6/68/681/681.3/681.3.0

Introduction

mkd est une commande destinée à extraire la **documentation** pour la **maintenance et l'exploitation des logiciels** en même temps que la **compilation**, selon les normes ISO[1] [2].

Disponibilité

Les **sources** compilables sont disponibles pour **Windows**, **Unix**, **Linux**.
Les **binaires** sont disponibles pour:

Windows[3] équipés de processeurs 32 et 64 bits.
Unix/Linux équipés de processeurs 32 bits x86 (i386), et 64 bits (amd64).

Les **paquets** d'installation sont disponibles pour les sytèmes à base **Debian**, **Ubuntu**, à télécharger[4].
Les **paquets RPM** pour les systèmes **RED-HAT** sont disponibles en téléchargement sur le serveur de l'**EELL**[5]
Les paquets Debian peuvent être convertis pour d'autres systèmes avec **Alien**[6]

Syntaxe

mkd [-ABCFPSafjlnpstvw] codes chemin_source [chemin_cible]

Description

`mkd`

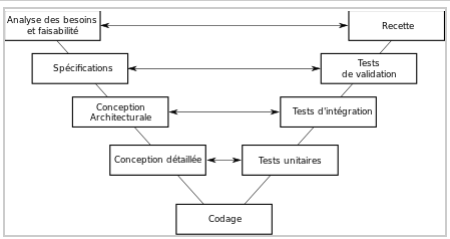
Les commentaires sélectionnés, dans le pseudo-code, ou tout commentaire, d'un fichier informatisé sont extraits et copiés dans un fichier cible « document » afin de produire la documentation désirée.

On peut extraire en standard tout document texte de n'importe quelle langue enregistrée au format UTF-8.

Habituellement les fichiers documents sont:

- Pour les fichiers de programmation (Assembleur, Basic, C, etc.);

Les fichiers documents sont des Organigrammes, des Structures de blocs de codes, des commentaires pour Programmeurs (parfois nombreux), des points de Tests (ou blocs de tests) et 'warnings' avec les numéros de lignes .... Les codes de sélection peuvent être respectivement 'O', 'S', 'P', 'T' et "www" juste après le caractère début de commentaire.



Cycle de développement (logiciel)

- Des fichiers spéciaux. Les options de compilation standards permettent de nombreuses possibilités. Elles permettent d'extraire des documents balisés avec des caractères spéciaux (" *texte* ") (% *texte* Fin de ligne) etc. Voir les options « -l ligne » et « -p page ».

Fichiers inhabituels:

Ajouter un fichier de décodage spécifique à un balisage (html, xml, etc).

codes

Ce sont des caractères alphanumériques. Les commentaires commençant par ces caractères sont extraits du fichier source et sont ajoutés au fichier « document ». Tous les caractères ASCII peuvent servir a coder les commentaires; toutefois on utilise ordinairement des caractères en majuscule, sauf w pour 'warning'. Avec deux étoiles “**”, mkd extrait tous les commentaires. (Voir aussi option t et les exemples).

chemin_source

C'est le hemin du fichier source (ou fichier de projet: avec option -j)

[chemin_cible]

C'est le chemin du fichier cible « document ». Par défaut chemin_cible est une copie du chemin_source auquel on remplace le suffixe par le suffixe ´.doc´ jusqu'à la version mkd_120508.

Options

Voir **Shell** Interface système.
Voir aussi les **Notes** pour les styles de langages.

```
Options [-ABCFPSafjlnpstvw]

       Les options en majuscules restreignent l'extraction des comm
entaires à
       un style de langage:

       -A     extrait le style Assembleur (   ;    -> fin de ligne)

       -B     style Basic                (REM ou ' -> fin de ligne)

       -C     style C++                  (   //  et  /*  ->  */   )
```

-F style Fortran (c,C ou * -> fin de ligne)

-P style Pascal ({ ... } et (* -> *))

-S style Shell ou Ratfor (# -> fin de ligne)

Les options en minuscules agissent sur la manière d'effectuer le décodage:

-a (append) Ajoute à la suite du fichier documentaire ´chemin_cible´.

-f (find) Trouve le langage du fichier source par analyse du suffixe. (s´utilise généralement avec un fichier projet)

-j (project) S´utilise avec un fichier projet composé de sources en langages différents.

-l (lignes) Extrait les lignes commençant par les caractères CD1 ou CD2 ou CD3 et suivis par un des caractères codes, le commentaire se termine par la lecture du caractère ´New Line´. CD1 et CD2 doivent être placés en 1ère colonne, alors que CD3 peut être placé en milieu de ligne. CD1, CD2, CD3, sont des options de compilation dans le fichier version.h de la distribution source de mkd. On peut connaître ces options compilées par la commande mkd \? sous Unix/Linux ou mkd ? sous DOS ou Terminal Widows.

-n insère un numéro de ligne

-p (page) Extrait le texte débutant par le caractère CD4 suivi d´un des caractères codes, l'extraction du commentaire se termine avec la lecture du caractère CD5. CD4 et CD5 sont des options de compilation dans le fichier version.h de la distribution source de mkd. On peut connaître ces options compilées par la commande

```
              mkd \? sous Unix/Linux ou mkd ? sous DOS ou Terminal
Widows.

       -s     (screen : add to)  Duplique  les  commentaires  extra
its  sur la
              sortie standard. Permet les redirection shell et  les
  filtrages
              >, >> , || grep , etc

       -t     (text) Ne copie que le commentaire.

       -v     (verbose) Bavard. Ajoute des détails de décodage dans
 le fichier
              cible

       -w     (overwrite) Crée ou écrase le fichier cible « documen
t ».
```

Options compilées et de compilation

Pour **visualiser les options compilées** il faut interroger la commande mkd:

mkd \? sous Linux ou mkd ? sous Windows.
On obtient les options qui ont été compilée avec la version courante de mkd, à la **ligne -l et -p**.

```
     -l et -p:   ligne;  (compil.: % ou - en première colonne ou #
 dans la ligne)
                 page:  (compil.: commence avec " et se termine av
ec ")
```

Ici, l'option -l permet:

avec # (typé shell) : d'extraire les conditions de compilation (#define, #ifdef, etc)

Cette option permet de se substituer à l'option impérative -S et d'utiliser l'option -f

avec % : typé PostScript
avec - : peut-être ADA ? commentaire ligne que l'on extrait aussi avec avec **cat** et **grep** (Voir aussi le manuel de grep)

```
grep -n -e --[Code] <Fichier_source> [> Fichier cible]
```

Autres options possibles avec -l : \' typé Basic, ; typé Assembleur, etc.

Ici, l'option -p permet d'extraire toutes les chaînes de caractères avec les codes '**'

Exemples

Exemple de fichiers sources pour générer la documentation

Quand c'est possible on écrit chaque fonction d'une application dans un fichier séparé, afin de produire une documentation dans un ordre alphabétique.
Quand les fonctions sont regroupées dans un même fichier, la documentation des fonctions apparaîtra dans le même ordre que dans le fichier.

Dans le fichier de la fonction on précise sa syntaxe d'utilisation, le fichier d'entête (header), ainsi que son usage.

Exemple pour la fonction cpp_ : du fichier cpp.c copié du manuel Linux "man 3 cpp"
Notez que les logiciels destinés à la traduction doivent être écrits en anglais

```
/*P
     File cpp.c
     Programmers directives (in V cycle)
     Last Date and Programmer name
*/

/*D
        function cpp_
 ------------------------------------------------------------------
-----------
NAME
       cpp_ - Extract coded informations from C, c++, php source fi
les.

SYNOPSIS
       #include  "/usr/include/mkd/version.h" // IMPORTANT: Compila
tion
          directives

       #include "/usr/include/mkd/cpp.h"

       int cpp_(FILE * pfdoc, FILE * pnfile);

DESCRIPTION
       The cpp_ function reads the source file (pnfile) transmitted

       from  the calling  function,  and  decodes  the  comments
       pre-encoded in lines or blocks of style C, c++ or php files,
 and
       writing  this  comments  in  a target file (pfdoc).

       Pre-coded characters are defined in a external global table
       'Codes':

              The golbal external variables are 'Codes' and 'Option
s':
```

```
       extern char codes[];
              They  must  be  défined in the calling function; char
              codes[5] =
              {0,0,0,0,0}; table of 5 ASCII characters on 8 bits.
              The comment are  extracted  if the comment begin with
              one of these characters.

       extern unsigned char n,s,t;
              They must be defined in  the  calling  function;
              unsigned char n=0,s=0,t=0;

       -n     The transcript is preceded by line number. This allow
s to
              easily reach the commented line.

       -s     Add the comment to the terminal or Konsole to  use  s
hell
              redirections > , >> , or || etc.

       -t     With  the  t  option  only  the commented text is cop
ied.
              Without the t option the entire line or block is  cop
ied.
              The  t option permit to generate directly exploitable
 and
              publishable documents.

RETURN VALUE
       Return 0 in case of success in  konsole  version,  nothing
with
       gtkmm.

CONFORMING TO
       All UNIX / LINUX systems with gcc

AUTHORS
       Contact : ..................

RESOURCES
       If  your product is designed to work with any graphical appl
ica-
       tion, then the file cpp.c++ is ready to  use  gtkmm.  (See
also
       mkdcppw)

FILES
       cpp.c, cpp.h, version.h

       /usr/include/mkd/ : mkd header files

       /usr/src/mkd/ : mkd source files

NOTES
```

```
       Not ready for read and decode files including sources files

BUGS
       Bugs Report: http://edeulo.free.fr/phpBB3/index.php

SEE ALSO
       Documents generator mkd(1) "man mkd"

COPYRIGHT: (Specified in version.h) :
*/

/*H
        // cpp.c:
        extern int cpp_ (FILE *pfdoc, FILE *pnfile);
*/

     int cpp_ (FILE * pfdoc, FILE * pnfile)
     { //S Function Begin
       ... // Code
     } //S Function End
```

Exemple de fichiers cibles obtenus

- **Exemple de fichier d'entête obtenu par la commande:**
  mkd -Ctw H cpp.c cpp.h

  Fichier cpp.h:

```
        // cpp.c:
        extern int cpp_ (FILE * pfdoc, FILE * pnfile);
```

- **Exemple de fichier pseudo code obtenu par la commande:**
  mkd -Ctnv O cpp.c cpp.txt

  Ci-Dessous: Notez que la première ligne est dûe à l'option -v, les lignes sont numérotée (option -n), seul le texte est copié (option -t)
  Le « code » d'extration est O comme Organigramme.
  Fichier cpp.txt

```
(fichier cpp.c :)Options a=0 f=0 j=0 l=0 n=1 p=0 s=0 t=1 w=1
  138    While next character is not End Of File
  142      If it is the first char then  c1=LF
  144      Else c1 is the next char
  146      If the char is New Line
  147        then mark the followed position in memory (long int 'n
l')
  156        If the char is '\t' tabulation, then tab++
  158        Else if the char is '/'
  160        Then:
  164          If the char is followed by c2 = '*' or '/' and if op
```

```
tions[0]=0
  165           -- or if it is followed by user char 'code[]'
  174           Then:
  176             If c2 = '/' set the Boolean 'ligne' to true (=1)
  178             Recognise the comment position
  182             If option n is true then insert the line number
  188             If option t is not true,
  189             Then:
  192               Positioning at the beginning line
  195               Copy the line to comment in the doc file
  203             Else: (option t)
  206               Copy the tabulations as much as there is in the
source file
  212               Copy spaces up to comment and:
  219                 If the line boolean 'ligne' is true (=1)
  221                 Then:
  222                   Copy the comment to End Of Line (EOL) or (EO
F)
  231                 Else while does not find the char * followed b
y / :
  232                   Copy the comment
  244                     If option n is true and char c1=NL
  245                     Then: insert the line number following NL
  271                 BEGIN FULL_LINE *** Compil. option
  272                 If t option is false,
  273                 Then:
  276                   Copy the comment to End Of Line or EOF
  277                   -- including '\r' (CR/LF) under DOS
  284                 Else (option t true)
  287                 END FULL_LINE *** Compil. option
  289                 t is true by default if FULL_LINE is not denin
ed at the
  290                 -- compilation
  292                   Go to EOL without copying, except the carria
ge return
  302                 And next copy New Lines under this form: \n
  307                 And backward to NL
  311               Else: (it is not a comment)
  314                 Back behind two characters
  321    Return of a character back to avoid the End Of File EOF !
```

Exemple de fichiers de commandes Shell. *Exemple pour générer les manuels et les mettre à jour*

Le fichier **Update_mkd_en.1.txt** doit être imposé au format UTF-8 avec un éditeur comme gedit ou BlueFish, pour être traduisibles dans toutes les langues. Vérifiez votre syntaxe avec GmanEdit (Éditeur Gnome)[7]
Note: Ceci est un exemple; Voir aussi: Installation du manuel français

```
#!/bin/bash
# File: install_manuals,
```

```
# command syntax: sudo install_manuals
# only for update see install
clear # or cls on windows
echo "install or update mkd manual mkd (1)"

# Autoriser la lecture et l'écriture des fichiers « chmod 666 »:
# chmod: Attention crée une erreur si les fichiers n'existent pas.
chmod 666 mkd_en.1 mkd_en.1.gz
# Créer le fichier du manuel en anglais (mkd_en.1):
mkd -Ctw 1 Update_mkd_en.1.txt mkd_en.1
# Compresser le fichier au format du manuel unix:
gzip -f mkd_en.1
# Autoriser le fichier en lecture seule:
chmod 444 mkd_en.1.gz

chmod 666 mkd_fr.1 mkd_fr.1.gz
mkd -Ctw 1 Update_mkd_fr.1.txt mkd_fr.1
gzip -f mkd_fr.1
chmod 444 mkd_fr.1.gz

# Installer le manuel standard en anglais:
cp -f mkd_en.1.gz /usr/share/man/man1/mkd.1.gz ;

# Si le répertoire fr.UTF-8/man1 existe dans le répertoire des manu
els,
# Alors: copier le manuel français dans ce répertoire (UTF-8),
# Sinon: le copier dans le répertoire standard des manuels en franç
ais (ISO).
if [ -d "/usr/share/man/fr.UTF-8/man1" ]; \
        then sudo cp -f mkd_fr.1.gz /usr/share/man/fr.UTF-8/man1/mk
d.1.gz ; \
        elif [ -d "/usr/share/man/fr/" ]; \
                then sudo cp -f mkd_fr.1.gz /usr/share/man/fr/man1/
mkd.1.gz ; \
fi
```

Exemple de Makefile pour générer la documentation

Toutes les chemins des fichiers de l'application sont regroupés dans un fichier de projet dans l'ordre alphabétique.

> Exemple : "**ls -1 *.c > app.prj**" qui va contenir le nom de tous les fichiers à examiner pour générer la documentation. ***Attention***, *ls -1 (chiffre un) et non -l (lettre 'l')*

La ligne de commande "**mkd -Cjt H app.prj app.h**" génère le fichier d'entête de toutes les fonctions de l'application.

La ligne de commande "**mkd -Cjt D app.prj app_functions.documentation**" génère le fichier de la documentation des fonctions de l'application. Si le fichier est composé de sources en langages différents il faut remplacer l'impératif -C par -f (find) trouver le langage.

Exemple de lignes dans un Makefile.

Dans cet exemple le Makefile se trouve dans le répertoire des fichiers sources du projet.

```
# APP is any "macro". Warning: no space after the macro and the cha
r '\'
APP = MyProgram

# Unconditional directive:
Create_header_and_functions-doc:
     # Warning: the first char is a tabulation and not spaces
     if [ -e "/usr/bin/mkd" ]; \
     then \
          # first : create or overwrite new project file \
          ls -1 *.cpp > $(APP).prj; \
          # create or overwrite header file \
          mkd -Ctw H $(APP).prj $(APP).h: \
          # create or overwrite functions documentation \
          mkd -Ctw D $(APP).prj $(APP).txt: \
          # option w : create or overwrite warnings documentation \
          mkd -Cwn w $(APP).prj $(APP).wars; \
     else \
          @echo "The mkd command is not found in bin directory"; \
     fi
```

Valeurs retournées et messages

mkd retourne la valeur 0 (zéro) en cas de succès, une valeur non nulle en cas d'erreur, et affiche:

```
syntaxe: mkd [-ABCFPSafjlnpstvw] codes chemin_source [chemin_cible]
   ou: mkd \? .Voir aussi le manuel: 'man mkd'
```

Ressources

Compilations avec:

- **gcc** sous unix / linux, et bibliothèques standards.
- **Microsoft visual studio** sous MS-Windows (Une version est prête à l'emploi, compilée avec Microsoft Visual Studio 2010[8])

Fichiers

Traductions

Il existe des traductions sous unix / linux: Voir la page Traductions Internationales

- Les langues anglais français sont intégrées dans la distribution 2012 et sont prévues pour les langues espagnol allemand italien

(Les fichiers de traduction .pot et .po sont à votre disposition) ainsi que sur le site des mainteneurs.

- Les **manuels** formatés **nroff** existent pour anglais allemand français et sont prévus pour espagnol italien
- Voir aussi la documentation de gettext[9] pour les traductions.

Notes

- Il appartient aux programmeurs de veiller à refermer les commentaires, blocs et lignes, par les codes de fermeture appropriés dans les fichiers sources.

  Attention: La fermeture d'un commentaire ligne est un retour chariot (NL, LF, CR/LF) selon les cas. Cette remarque implique un retour chariot en fin de commentaire ligne. Dans ce cas vous devez avoir une ligne vide à la fin du fichier source.

- Jusqu'à la version 2012, mkd ne lit et ne décode pas de façon récursive les fichier inclus par '#include' dans les fichiers sources.

Droits de copie

© EELL, Éditeurs Européens de Logiciels Libres, EUPL 2007.

Vous êtes libre de copier reproduire et distribuer le livre selon les termes de la Licence Wikipédia Creative Commons Paternité-Partage des Conditions Initiales à l'Identique 3.0 non transposé (CC-BY-SA 3.0) et licence de documentation libre européenne EUPL[10] *European Union Public Licence* totalement compatible avec GNU

Droit de copie des fichiers sources

```
 DROIT DE COPIE DES FICHIERS SOURCES:
  © EELL, Éditeurs Européens de Logiciels Libres, EUPL 2007.
  Association à but non lucratif selon l'Article 11 de la convention
  européenne des droits de l'homme.
  Concédée sous licence EUPL, version 1.1 ou – dès leur approbation pa
r la
  Commission européenne - versions ultérieures de l’EUPL (la «Licence»
).
  Vous ne pouvez utiliser la présente œuvre que conformément à la Lice
nce.
  Vous pouvez obtenir une copie de la Licence à l’adresse suivante:
    https://joinup.ec.europa.eu/system/files/FR/EUPL%20v.1.1%20-%20Lic
ence.pdf
  Sauf obligation légale ou contractuelle écrite, le logiciel distribu
é sous la
  Licence est distribué «en l’état», SANS GARANTIES OU CONDITIONS QUEL
LES
  QU’ELLES SOIENT, expresses ou implicites.
  Consultez la Licence pour les autorisations et les restrictions ling
uistiques
  spécifiques relevant de la Licence.
```

```
  Concernant la documentation : la Licence est totalement compatible a
vec la
  licence de documentation libre GNU.
```

Bugs

Mise à jour des bugs. (http://edeulo.free.fr/phpBB3/viewforum.php?f=14)

Voir aussi

anglais Comparison of documentation generators
(http://en.wikipedia.org/wiki/Comparison_of_documentation_generators)

mkdcppw (https://launchpad.net/~mkdcppw/+archive/mkdcppw), mkdcpp en mode graphique.

Histoire

La commande **mkd** pour MS-DOS et nommée **mkdoc** pour Unix, a été créée au Centre d'Electronique de Montpellier (CEM) en 1986 pour générer la documentation logicielle des interfaces logiciel-matériel.
Il n'existait alors pas de logiciel connu pour générer cette documentation aux multiples langages (Assembleur, C, Fortran, Pascal, etc.) Les bibliothèques au format COFF (Common Object File Format) permettaient d'obtenir des bibliothèques communes.

mkdoc a peu évolué jusqu'en 2007. Il a été compilé pour MS-DOS, Unix BSD Sun, Linux RedHat, Windows 98, Windows 2000. Les commentaires des fichiers de l'application ainsi que les messages sont restés au format ASCII.

En **2007** le nom ***mkdoc est abandonné*** et retrouve son premier nom **mkd** avec la version 10.01 compilée avec Visual C++ au format de texte ISO-8859-1

En **2012** mkd 12.03 s'adapte au format de texte international **UTF-8** pour son internationalisation.
Dans la foulée mkdcppw est écrit pour générer la documentation des fichiers de langages style C, (c++, php, etc.) en mode graphique (fenêtres).

Notes et références

- Notes Style de langage[11]:

Définition de NL: *New Line*, fin de ligne.

Les styles impératifs (ABCFPSlp) s'excluent les uns des autres et ne permet pas la recherche de style par l'option '-f' (trouver le style)

- Style A (Commentaires style Assembleur)

; *Commentaire* NL

- Style B (Commentaires style Basic)

  REM (après un marqueur ' : ' dans QBasic) *Commentaire* NL
  ' *Commentaire* NL

- Styles C

  /* *Commentaire* */ : C, C++, C#, Java, Javascript, PHP, VHDL 2008, ...
  // *Commentaire* NL : C, C++, C#, CSS, D, Delphi, Java, Javascript, PHP, Scilab ...

- Style F (Commentaires style Fortran 77) voir Style l pour le Fortran 90 et plus récents.

  C ou c ou * *Commentaire* NL ; C,c ou * en première colonne.

- Style P (Commentaires style Pascal):

  (* *Commentaire* *) : AppleScript, Caml, Modula-2, Modula-3, Oberon, Pascal, ...
  { *Commentaire* } Delphi (Pascal objet)

- Style S (Commentaires style Shell): Csh, Bash, sh, et autres shells, Makefile, Perl, PHP, Python, Ratfor, Ruby, Tcl, ...

  # *Commentaire* NL

- Option l (selon option compilée) ; décodage des lignes. Remplace une commande impérative (ABCFPS) et les trois extractions peuvent être associée simultanément. CD_1 et CD_2 en première colonne, CD_3 n'importe où dans la ligne.

  avec CD_3 = ' ! ' Fortran 90
  avec CD_{123} = ' % ' Latex, Matlab, Postscript, Tex
  avec CD_{123} = ' # ' comme style S

- Option p (selon option compilée) ; décodage des blocs:

  Avec CD_4 = ' (' et CD_5 = ') ' : (*Commentaire*)
  avec $CD_4 = CD_5$ = ' " ' Chaînes de caractères " *Commentaire* "
  avec $CD_4 = CD_5$ = ' ' ' Chaînes de caractères ' *Commentaire* '
  avec CD_4 = ' < ' et CD_5 = ' > ' tous les tags

- Remarques: Il est toujours possible d'inclure un style de commentaire dans le commentaire d'un autre style !

- Références

1. ISO/IEC 26514:2008 (http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=43073)
2. ISO/IEC 26514:2008 prévisualisation (http://webstore.iec.ch/preview/info_isoiec26514%7Bed1.0%7Den.pdf)
3. mkd pour **Windows** (http://edeulo.free.fr/wiki/index.php/Projet_mkd/MS-Windows)
4. Launchpad (https://launchpad.net/~jean-paul-louyot/+archive/mkd/+copy-packages)

5. Téléchargement RPM sur le serveur de l'EELL (ftp://62.147.143.241/t%E9l%E9chargements/mkd-Fedora/)
6. Site officiel du convertisseur de paquets Alien (http://joeyh.name/code/alien/)
7. GmanEdit (Gnome manpage editor) (http://gmanedit.sourceforge.net/)
8. Package redistribuable Visual C++ 2010 (x86) (http://www.microsoft.com/fr-fr/download/details.aspx?id=5555)
9. gettext
10. Licence Publique de l'Union Européenne (fichier pdf) (http://edeulo.free.fr/documents/EUPL%28licence%29FR.pdf)
11. https://fr.wikipedia.org/wiki/Commentaire_(informatique)

Mkd (Extracteur de documents)
Deuxième partie :
EXERCICES

Exercices

Make documents

Introduction

Pour ce faire il faut évidemment disposer du matériel (Extrateur de documents, fichiers encodés, etc.) Nous proposons ici des fichiers déjà disponibles, mais vous pouvez essayer avec vos propres fichiers dans la langue et le langage qui vous convient.

Télécharger mkd

- Version Windows:

  Binaire 32 bits mkd.10.03.zip (http://edeulo.free.fr/telechargements/console-ms/x86-32/mkd.10.03.zip) *(Compilé avec Visual VC10)*.
  sources-mkd.10.03.beta.zip (http://edeulo.free.fr/telechargements/sources/sources-mkd.10.03.beta.zip) *Pour MSDEV, Visual VC10, UNIX / LINUX*.

- Versions Unix / Linux: (Paquets sécurisés sur Launchpad)

  Version Debian / Ubuntu sur Launchpad:
  Binaires, et ***sources*** *sous réserve d'adapter les chemins des répertoires man et catman.*:

  Paquets pour Ubuntu / Debian (https://launchpad.net/~jean-paul-louyot/+archive/mkd/+copy-packages)

  Version RPM (Fedora): (Paquets sécurisés sur le serveur eell.fr)

  Paquets RPM (ftp://62.147.143.241/t%E9l%E9chargements/mkd-Fedora/)

  Autres systèmes linux : Convertissez les paquets debian avec la commande *alien*

- Vous pouvez tenter de compiler mkd sur Mac:

  Exercices en langage C - GCC standard sur Mac

Sources pour les exercices

Les fichiers ci après sont universels (pour tous les exercices et tous systèmes d'exploitation.)

cpp.c pour exercices

asm.c pour exercices

Autres formats de fichiers

Exercices élémentaires pour se familiariser avec la commande mkd

VOTRE ATTENTION: A l'impression les solutions n'apparaissent pas dans le texte. Les solutions aux exercices se trouvent dans la section suivante, dans le chapitre nommé *Comprendre les options de la commande mkd*.

- Vérifiez votre commande mkd. Tapez la commande *mkd*. L'erreur de syntaxe devrait s'afficher:

```
 mkd PC version, Release 10.03, USAGE:
 syntax: mkd [-ABCFPSafjlnpstvw] char_codes path_source [path_targe
t]
     or: mkd ? .See also mkd.doc or manual.
```

- Inspirez-vous du **manuel en français ou en anglais**

- Observez l'effets des options a,s,t,v,w de la commande mkd
- Observez ensuite l'action des *codes* de sélection de commentaires D, H, O, P, S, T, w (5 maximun)

- *Rappel de la commande:* ***mkd [-ABCFPSafjlnpstvw] codes chemin_source [chemin_cible]***

Dans le répertoire des fonctions, tapez les commandes ci dessous dans un terminal:

Commande: mkd -Cwsv H cpp.c

Testez **mkd -Cwsv H cpp.c** (pour décoder le **H**eader de style **C**, **w** pour écraser un fichier existant, **s et v** pour afficher à l'écran en mode bavard) :

Comprendre les options : aperçu 1.1

Commande: mkd -Cws H cpp.c

```
Quel est l'effet de l'option -v de la commande précédente ?
```

Comprendre les options : aperçu 1.2

Commande: mkd -Cwts H cpp.c

```
Quel est l'effet de l'option -t sans l'option -v ?
```

Comprendre les options : aperçus 1.3

Commande: mkd -Cwt H cpp.c *.h

```
Comparez ce résulat avec la commande précédente.
```

Comprendre les options : aperçus 1.3

Résumé des observations sur les options s, t, v

```
Faites un résumé de l'utilisation des options n,s,t.v,w.
```

Essayez aussi l'option -n

```
Quelle syntaxe proposez-vous pour la génération du fichier d'entête
 cpp.h ?
```

Comprendre les options : Résumé des observations sur les options s, t, v

Créez ou écrasez le fichier cible de l'organigramme cpp.organigramme

- Organigramme simple avec numérotation des lignes: code 'O'
- Organigramme avec structure avec numérotation des lignes: codes "OS"
- Organigramme complet avec les tests warnings et numérotation des lignes: Codes "POSTw"

Comprendre les options : Solution 1.5

Crééz ou écrasez le fichier de documentation logicielle html du fichier cpp.c

- Le but de l'exercice est de se familiariser avec l'option -a.
- Le **Fichier cible: aura pour nom "cpp-documentation_logicielle_codage.html"**.
- Créez *la tête* **du fichier html avec le code 'h' options -Cwt**

  Exemple: mkd -Ctw h cpp.c cpp-documentation_logicielle_codage.html

- Ajoutez *le corps* **avec le code 'd' et les options -Cat (Option -a comme append, ajouter au fichier cible)**
- Ajoutez *le pied* **du fichier html avec le code 'f' et les options -Cat**

- Visualisez le fichier cpp-documentation_logicielle_codage.html avec votre navigateur

- Quelle meilleure solution proposez-vous pour créer cpp-documentation_logicielle_codage.html ?

Comprendre les options : *Solution 1.6*

Exercices pour se familiariser avec les fichiers de projets

ci-après: fichier *est un nom quelconque sensé représenter votre projet dans le répertoire des fichiers sources.*

Utiliser l'option -f **(finder)**:
Pour valoriser cette option il serait utile d'ajouter un fichier d'un autre langage de programmation à votre convenance *(Trouvez sur wikibooks ou wikipedia)*. A défaut recopiez et collez le texte ci-dessous dans un fichier nommé putchar.asm *sous Windows* ou putchar.s *sous Linux*, à placer dans le répertoire du projet, avec cpp.c et asm.c
Trouvez l'exemple en assembleur Microsoft (MASM) sur wikipédia[1]:

putchar.asm

Ce court fichier est utilisé à titre d'exemple pour donner une solution réaliste aux exercices et peut être remplacé par tout autre fichier en assembleur.

```
;P Fichier puchar, macro en assembleur pour MS-DOS
;P ATTENTION: N'oubliez pas la dernière ligne vide
;P Macro trouvée sur wikipedia:fr:Assembleur#Macro-assembleur
;P Pour l'exercice suivant:
;P    wikibooks:fr:Mkd_(Extracteur_de_documents)/Exercices#Fichiers_de
_projet
;
;D Fichier putchar.asm sous DOS/Windows
;D Fichier putchar.s sous Unix/Linux
;D macro putchar
;D ------------------------------------------------------------------
---------
;D putchar est une macro MASM qui affiche un caractère sous MS-DOS.
;D On l'utilisera par exemple ainsi
;D        putchar "X"
;D Et cela générera :
;D        mov    dl,"X"
;D        mov    ah,2
;D        int    21h
;D
;
;H   // puchar.asm:
;H   // Pas de pototype
;
;O macro.asm
 putchar Macro   car          ;O Prototype de la macro
         ifdef   car          ;O si car est défini
         mov     dl,car       ;O   le mettre dans dl
         endif
         mov     ah,2         ;O ah=2 : fonction "putchar" en DOS
         int     21h          ;O appel au DOS
         endm                 ;O fin macro
```

Aperçus et solutions des exercices pour comprendre les "Fichiers de projets"

Créer un fichier de projet manuellement

Projet exercice.prj ; à créer manuellement avec les fichiers disponibles asm.c cpp.c putchar.s

Comprendre les fichiers de projets : Aperçu

Créer un fichier de projet avec les commandes système

Projet exercice.prj ; à créer avec les commandes de votre système d'exploitation; ls, dir.

Comprendre les fichiers de projets : Solution

Créer un fichier de commandes pour :

*Vous aurez copié les fichiers à éprouver dans le répertoire des essais . exécutez la commande **mkd \?** pour visualiser les extensions reconnues par votre mkd. Ces extensions dépendent des options de compilation.*

Les fichiers sont écrits dans des langages de programmation différents, le but est de créer des documentations globales avec ou sans le *finder* (option -f).

Comprendre les fichiers de projets : Aperçus et solutions des "Fichiers de commandes"

Mettre à jour le fichier d'entête. header global

(header global, exercice.h)

Comprendre les fichiers de projets : Solutionl

Mettre à jour l'organigramme de chaque fichier de programme

Pseudo-code; Code de repérage 'O', asm.org, cpp.org, putchar.org

Comprendre les fichiers de projets : Solution et aperçu avec commentaires

Mettre à jour la structure de chaque fichier

avec le pseudo code. Code 'S' et 'O' et 'w', *fichier*.struct
Fichier:s asm.struct, putchar.struct.

Comprendre les fichiers de projets : Aperçus

Mettre à jour la documentation globale pour les programmeurs

avec les dates de mise à jour etc. Code 'P' Fichier: exercice.info_programmeurs

Comprendre les fichiers de projets : Aperçu

Mettre à jour *de* la documentation logicielle globale

(Conception détaillée, Tests unitaires, du cycle en V) Code 'D' Fichier: exercice.docu

Comprendre les fichiers de projets : Aperçu

Notes sur le *finder*

Le *finder* (trouveur) cherche le style du langage par examen de l'extension du nom de fichier, l'option -f **est rarement utilisée**.

Extensions spécifique à Windows et MS-DOS: Assembleur, Basic, C, Fortran, Pascal, C ou Prolog.

.ASM .BAS .C .FOR .PAS .PRO extensions en majuscules ou minuscules.

Extensions spécifique à unix / linux: assembleur, C, Fortran, Ratfor, Pascal, Shell, Cshell

.s .S .c .h .i .f .F .r .p .sh .csh

Notez que les extensions de fichiers .c .h et .prj sont toujours reconnus alors que .c++, .cpp, .hpp ne sont pas reconnus par le *finder* tout au moins jusqu'à la version 2012. Il est nécessaire de préciser l'option 'C' dans la ligne de commande.

REMARQUES:

c++, cpp, hpp seront probablement reconnus dans les prochaines mises à jour.
Il est relativement facile de l'implémenter dans le fichier "find.inc.c" ou find.i

Les commentaires en style ADA commencent par deux tirets et ne sont pas reconnus en standard par le *finder* (trouveur) de mkd, cependant si il est facile de l'implémenter dans le fichier "find.i" il faudra aussi créer un fichier spécifique à ADA (ada.c ou ada.i)

Fichiers de tests unitaires

- Inspirez-vous de la documentation de la fonction cpp_() *(fichier cpp.c)* pour créer un fichier de commandes qui met la fonction cpp_() à l'épreuve.
- **Conditions des tests:** tous les tests se feront avec l'option impérative -C
- Vérifiez les options de la fonction: n, s, t, séparément.

  Vérifiez que la numérotation des lignes est correcte dans tous les cas de décodage (lignes et blocs). n, ns. nt, nst.
  Vérifiez que le texte reste à la bonne place dans tous les cas de décodage: toute la ligne (dans la cas de la directive de compilation FULL_LINE), ou texte seul.

  - Vérifiez que les tabulations sont bien prises en charge dans tous les cas, décodage des lignes et des blocs
  - Vérifiez de même pour les espaces.

  Vérifiez que le caractère de fin de fichier n'apparaît pas en fin de texte lorsque le commentaire ne se termine pas par une fin de ligne.

- Documentation:

  Ajouter des modules : Documentation pour éprouver les modules.

  Ajouter des modules : Une solution à cet exercice

Notes et références

Philosophie de la méthode en génie logiciel avec ADA

1. wikipedia:Assembleur#Macro-assembleur

Comprendre les options de la commande mkd

Make documents

Généralités

Syntaxe de la ligne de commande
mkd [-ABCFPSafjlnpstvw] codes chemin_source [chemin_cible]
Distinguer les options impératives
Les options de langages en majuscules ABCFPS permettent d'extraire les commentaires codés dans un style de langage générique.

Le style C par exemples peut décoder le C, c++, c--, c#, CSS, java, javascript, php

Les options de langage en minuscules l et p

Ces options dépendent de la compilation CD1, CD2, CD3, CD4, CD5. On peut connaître ces options par la commande mkd \?; options l et p
Les options -l d'extraction des lignes peuvent être utilisées simultanément;
les options -l; CD1 et CD2 :'%' '<' et CD3 '!' on peut simultanément extraire le PostsScript, les tags html, les commentaires Fortran 90
L'option -p d'extraction de blocs permet d'extraire des blocs de chaînes de caractères avec les options CD4=CD5= '\"'. CD1 '(' avec CD5 ')' permet d'extraire des blocs entre parenthèses

Les options d'extraction et d'ouverture des fichiers anstvw
L'ouverture des fichiers peut être la création du fichier cible avec **l'option -w** ou l'ajout avec **l'option -a**
Les options nstv sont des options d'extration et d'affichage

n permet de connaître le numéro de la ligne extraite
s permet une copie vers la sortie standard (un écran, un fichier, une imprimante, etc.
t permet d'éliminer les codes d'extraction et les caractères de début et de fin de commentaire
v mode bavard qui permet d'afficher les conditions d'extraction au terminal ou à la sortie standard

Exercices élémentaires pour se familiariser avec la commande mkd

Commande: mkd -Cwsv H cpp.c

Aperçu 1.1

en mode bavard (-v), vous devez obtenir à la console:

```
Options a=0 f=0 j=0 l=0 n=0 p=0 s=1 t=0 w=1
fichier doc : 'cpp.doc'

fichier pour doc: 'cpp.c'

(fichier cpp.c :)
```

```
/*H
   // Files: cpp.c (cpp.inc.c++ for mkdcppw and gtkmm)
   extern int cpp_ (FILE * pfdoc, FILE * pnfile);
*/

DOC cpp.doc FIN
```

Le fichier cpp.doc contient:

```
(fichier cpp.c :)Options a=0 f=0 j=0 l=0 n=0 p=0 s=1 t=0 w=1

/*H
   // Files: cpp.c (cpp.inc.c++ for mkdcppw and gtkmm)
   extern int cpp_ (FILE * pfdoc, FILE * pnfile);
*/
```

Notez qu'avec l'option -v on ne peut pas obtenir un fichier d'entête cpp.h car cette option crée une première ligne avec le nom du fichier.

Commande: mkd -Cws H cpp.c

Aperçu 1.2

Sans l'option -v on obtient au terminal:

```
(fichier cpp.c :)
/*H
   // Files: cpp.c (cpp.inc.c++ for mkdcppw and gtkmm)
   extern int cpp_ (FILE * pfdoc, FILE * pnfile);
*/
```

Le fichier cpp.doc contient:

```
/*H
   // Files: cpp.c (cpp.inc.c++ for mkdcppw and gtkmm)
   extern int cpp_ (FILE * pfdoc, FILE * pnfile);
*/
```

Les options Cw ou Cws ne suffisent pas pas pour créer le fichier d'entête cpp.h car les caractères de commentaires sont recopiés.

Commande: mkd -Cwts H cpp.c

Aperçu 1.3

Avec l'option -t et sans l'option -v on obtient au terminal:

```
(fichier cpp.c :)
   // Files: cpp.c (cpp.inc.c++ for mkdcppw and gtkmm)
   extern int cpp_ (FILE * pfdoc, FILE * pnfile);
```

Le fichier cpp.doc contient:

```
   // Files: cpp.c (cpp.inc.c++ for mkdcppw and gtkmm)
   extern int cpp_ (FILE * pfdoc, FILE * pnfile);
```

Commande: mkd -Cwt H cpp.c *.h

Cette fois le fichier d'entête cpp.h peut être créé correctement et cette syntaxe doit être employée dans les fichiers de commandes, ainsi que dans les Makefiles

Résumé des observations sur les options s, t, v

Résumé des observations

1. L'option -v, mode bavard, est inappropriée pour générer des documents imprimables. Elle sert pour la mise au point des fichiers de commandes avec l'option -s
2. L'option -s permet la mise au point des fichiers de commandes sans nécessité d'éditer le fichier cible.
3. L'option -t est nécessaire pour obtenir des document présentables et imprimables.

Créez ou écrasez le fichier cible de l'organigramme cpp.organigramme

Solution 1.5

- Organigramme simple avec numérotation des lignes:

mkd -Cnstw O cpp.c cpp.organigramme

- Organigramme avec structure avec numérotation des lignes:

mkd -Cnstw OS cpp.c cpp.organigramme

- Organigramme complet avec les tests warnings et numérotation des lignes:

mkd -Cnstw POSTw cpp.c cpp.organigramme

Crééz ou écrasez le fichier de documentation logicielle html du fichier cpp.c

Solutions 1.6

- Familiarisation avec l'option -a

mkd -Ctw h cpp.c cpp-documentation_logicielle_codage.html

mkd -Cat d cpp.c cpp-documentation_logicielle_codage.html

mkd -Cat f cpp.c cpp-documentation_logicielle_codage.html

- Solution globale pour un seul fichier:

mkd -Cat hdf cpp,c cpp-documentation_logicielle_codage.html

Notes et références

Notez que:

Les options impératives -ABCFPS sont incompatibles entre elles et avec l'option -f *(find - trouver le style du langage de programmation).*
Les options -a (append - ajouter) et -w (overwrite - écraser) sont difficilement compatibles ...

Comprendre les fichiers de projets pour mkd

Make documents

Exercices pour se familiariser avec les fichiers de projets

Les fichiers de projets d'extension .prj ou .pj sont des fichiers qui contiennent la liste des fichiers *(sources)* pour générer les documents *(cibles)*.

Les fichiers de la liste ont parfois des extensions différentes.

Si les fichier sont tous de même style de commentaires on peut imposer une des options impératives pour générer les documents, mais pour utiliser un fichier de projet il est nécessaire de préciser que le fichier source est un fichier de projet avec l'option -j souvent associé à l'option -f *(find)*. Dans le cas où *find* ne trouve pas le type de fichier, les styles compilés sont proposés à la sortie standard *(Terminal en général)*
L'option -j est compatible avec les options impératives.

Créer un fichier de projet manuellement

Projet **exercice.prj** ; à créer manuellement avec les fichiers disponibles asm.c cpp.c putchar.s

Aperçu

Manuellement on est maître de la liste des fichier et de l'ordre nécessaire au décodage.

Nom du fichier de projet **exercice.prj** ; dans l'ordre alphabétique:

```
asm.c
cpp.c
putchar.s
```

Créer un fichier de projet avec les commandes système

Projet **exercice.prj** ; à créer avec les commandes de votre système d'exploitation; **ls, dir**.

Solution

- **Avec terminal linux:** ***ls***

```
ls -1 *.s *.c > exercice.prj
```

Fichier exercice.prj : *La mise en ordre alphabétique est automatique.*

```
asm.c
cpp.c
putchar.s
```

- **Avec terminal cmd** *Windows 7,* ***dir***: option /B: noms courts, et option /ON: tri alphabétique.
  Notez que le terminal PowerShell ne convient pas.

```
dir /B /ON *.asm *.c > exercice.prj
```

Fichier exercice.prj :

```
asm.c
cpp.c
putchar.asm
```

Créer un fichier de commandes pour :

Vous aurez copié les fichiers à éprouver dans le répertoire des essais . exécutez la commande ***mkd \?*** *pour visualiser les extensions reconnues par votre mkd. Ces extensions dépendent des options de compilation.*

Les fichiers sont écrits dans des langages de programmation différents, le but est de créer des documentations globales avec ou sans le *finder* (option -f).

Mettre à jour le fichier d'entête. header global

exercice.h

Solution

La commande doit trouver le type de fichier du projet exercice.prj avec l'option -f

```
mkd -jfstw H exercice.prj exercice.h
```

Fichier exercice.h:

```
   // asm.c:
   extern int asm_(FILE *pfdoc, FILE *pnfile);

   // Files: cpp.c (cpp.inc.c++ for mkdcppw and gtkmm)
   extern int cpp_ (FILE * pfdoc, FILE * pnfile);
```

```
// puchar.asm:
// Pas de pototype
```

Mettre à jour l'organigramme de chaque fichier de programme

Pseudo-code; Code de repérage 'O',
asm.org, cpp.org, putchar.org

Solution et aperçu avec commentaires

1. **mkd -Cstw O asm.c asm.org**
2. **mkd -Cstw O cpp.c cpp.org**
3. **mkd -Astw O putchar.asm putchar.org**

(L'exemple putchar avec MASM est forcément sous MS-DOS)

Fichier putchar.org:

```
macro.asm
 putchar Macro   car           O Prototype de la macro
         ifdef   car           O si car est défini
         mov     dl,car        O   le mettre dans dl
         mov     ah,2          O ah=2 : fonction "putchar" en DOS
         int     21h           O appel au DOS
         endm                  O fin macro
```

Notez que la première ligne est correcte: *macro.asm*, alors que les commentaires qui suivent sont précédés du caractère O du fait que toute la ligne est recopiée sauf le caractère ; (Ceci est dû à l'option de compilation FULL LINE): La copie de la ligne est entière le caractère de commentaire est remplacé par un espace ***ligne 256 du fichier asm.c***. (**Voir asm.struct** de l'exercice suivant "Mettre à jour la structure de chaque fichier")
Pour éviter ce qui pourrait être un inconvénient il suffit de recompiler mkd sans l'option FULL_LINE.
Pour une bonne présentation on a toujours intérêt à **écrire le pseudo-code sur des lignes séparées avec le caractère ; en première colonne**.

Bug: Un des bugs connus est la fin de ligne qui se termine par EOF à la place de EOL (NL retour chariot)

Mettre à jour la structure de chaque fichier

avec le pseudo code. Code 'S' et 'O' et 'w',
asm.struct

Aperçu de asm.struct et commentaire

mkd -Cswn OSw asm.c asm.struct

Fichier asm.struct *(édité avec Bluefish)*

```
(fichier asm.c :)
   98    { /*S asm */
  109 /*O   tant que pas fin de fichier source (c1, c2, c3 différents
de EOF)*/
  111       { /*S tq !EOF */
  112 /*O      si début de texte faire c1=LF */
  114 /*O      sinon prendre pour c1 le char suivant */
  116 /*O      si le char est NL repérer la position qui suit 'NL' (re
pérer dans
  117           -- l'entier long nl) */
  119           { /*S reperage debut de ligne */
  123           } /*S reperage debut de ligne */
  125 /*O----- si le char est NL ----------------------------------
--------------*/
  127           { /*S ; en colonne 1 */
  130 /*O         si c1 est suivi par c2 = ';' et
  131               -- si codes[0]=0 ou si suivi par un des 5 char code
d'utilisateur */
  141 /*O         alors copier les caractères dans le fichier doc, et
si option s
  142               -- ajouter au terminal */
  143               { /*S copier */
  144 /*O            si option n insérer le numéro de ligne et si opti
on s l'ajouter au
  145                  -- terminal */
  147                  { /*S n */
  150                  } /*S n */
  151 /*O            si l'option t est fausse (pas de texte seul) */
  153                  { /*S !t */
  154 /*O               copier le début de ligne */
  156 /*O               si option s copier le commentaire au terminal
*/
  158                     { /*S si opt s */
  161                     } /*S si opt s */
  162                  } /*S !t */
  163 /*O            sinon si il ne faut pas copier tout le commentair
e:
  164                  -- reculer de 1 caractère */
  166 /*O            tant que l'on a pas trouvé le caractère fin de li
gne NL ('\n')*/
  168                  { /*S w copy line */
  169 /*O               copier les caractères de la ligne */
  172                  } /*S w copy line */
  173 /*O            copier aussi le caractère de fin de ligne et, ave
c l'option s
  174                  -- envoyer le retour-chariot au terminal */
  177 /*O            revenir sur LF du fichier source */
  179               } /*S copier */
  180 /*O         sinon restituer les deux derniers caractères */
  182               { /*S sinon */
  185               } /*S sinon */
```

```
  186          } /*S ; en colonne 1 */
  188 /*O--------- sinon si c1 = ';' est dans la ligne ---------------
-------------*/
  190          { /*S else */
  191 /*O         si le caractère c1 est une tabulation incrémenter ta
b */
  193 /*O         sinon si c1 = ; */
  195 /*O         alors: */
  196             { /*S char = ; */
  198 /*O            si codes[0]=0 ou si suivi par c2 = char code util
isateur */
  206 /*O            alors: */
  207               { /*S commentaire page */
  208 /*O               repérer la position de début de commentaire */
  211 /*O               si l'option booléenne t est fausse (pas de tex
te seul) */
  213                  { /*S !t */
  214 /*O                  se positionner en début de ligne */
  217 /*O                  si l'option n est vraie, insérer le numéro
de ligne */
  219                     { /*S n (numéro de ligne) */
  221 /*O                     si option s vraie, ajouter le numéro de
ligne à l'écran */
  223                     } /*S n */
  224 /*O                  copier toute la ligne tq pas NL (CR/LF), da
ns le fichier doc */
  226                     { /*S w */
  229                     } /*S w*/
  230                  } /*S !t */
  231 /*O               sinon: (option t) */
  233                  { /*S option t */
  234 /*O                  se positionner en dedut de ligne */
  237 /*O                  si option n vraie, insérer le numéro de lig
ne */
  239                     { /*S n (numero de ligne) */
  241 /*O                     si s vrai, ajouter le numéro de ligne à
l'écran */
  243                     } /*S n */
  246                     //P { /*S tabs */
  249                     //P } /*S tabs */
  250 /*O                  copier toute la ligne jusqu'à la position c
ommentaire */
  252                     { /*S w */
  255                     } /*S w*/
  256 /*O                  ajouter un espace pour remplacer le caractè
re ';' */
  258                     { /*S espaces */
  261                     } /*S espaces */
  262 /*O                  puis copier le commentaire tant que NL n'es
t pas trouvé */
  264                     { /*S copier commentaire */
  267                     } /*S copier commentaire */
  268                  } /*S option t */
  269                  putc('\n',pfdoc);if(s)putch('\n');   /*O copie
r NL */
  270                  ungetc(c1,pnfile);                   /*O reven
```

```
ir sur NL */
  271                } /*S commentaire page */
  272 /*O                              sinon: */
  274                { /*S*/
  275 /*O                revenir un caractère en arrière */
  277                } /*S*/
  278             } /*S char = ; */
  279          } /*S else */
  280       } /*S tq !EOF */
  282    } /*S asm */ÿ
```

Notez la fin de fichier avec le caractère ÿ qui témoigne d'une fin incorrecte sur EOF

putchar.struct

Aperçu de putchar.struct et commentaire

mkd -Aswn OSw putchar.asm putchar.struct

```
(fichier putchar.s :)
   19 ;O macro.asm
   20   putchar Macro   car          ;O Prototype de la macro
   21           ifdef   car          ;O si car est d?fini
   22           mov     dl,car       ;O   le mettre dans dl
   24           mov     ah,2         ;O ah=2 : fonction "putchar" en D
OS
   25           int     21h          ;O appel au DOS
   26           endm                 ;O fin macro
```

Notez la fin de fichier correcte qui témoigne d'une fin correcte sur EOL (New Line)
Notez aussi qu'un des caractères n'est pas au format UTF-8 *(le é de défini)*

Mettre à jour la documentation globale pour les programmeurs

avec les dates de mise à jour etc. Code 'P'
exercice.info_programmeurs

Aperçu

mkd -fjstw P exercice.prj *.info_programmeurs

```
    Fonction asm_(); Fichier asm.c

    Documentation simplifiée.
    -------------------------
    La fonction  asm_() doit décoder des commentaires, pré-codés avec
un carac-
```

```
    tère ASCII  qui suit le début de commentaire.
    Exemple ;A en 'début' ou 'dans' la ligne.
    ... et bla bla bla ...(raccourci pour les exercices)
    Tests unitaires:
    ----------------
    La fonction doit être éprouvée de telle sorte qu'un commentaire qu
i se
    termine par le caractère de fin de fichier soit entièrement copié
dans
    le fichier cible.
    Le caractère de fin de fichier décodé avant le caractère de fin de
 ligne
    doit être remplacé par un caractère de fin de ligne.
    Le caractère de fin de fichier (EOF) ne doit jamais apparaître dan
s texte
    du fichier cible.


 ------------------------------------------------------------------------
--------
© Contact: http://edeulo.free.fr/contacts/formmail.php
© EELL, Editeurs Européens de Logiciels Libres, EUPL 2007
Association à but non lucratif selon l'Article 11 de la convention eur
opéenne
des droits de l'homme.
 ------------------------------------------------------------------------
--------

NOM DU FICHIER: asm.inc.c asm.c

PROJET: mkd
Générer la documentation pré-écrite dans les fichiers de sources multi
ples.
PROJET INITIAL: mkdoc 1989 pour MS-DOS et UNIX, devenu obsolète.
DOSSIER: extractdoc 04/12/2009

MODIFICATIONS:
asm.inc.c
....
asm.c
Le: ../../2010 par JPL Transformation into independent function for mk
dasm
                Move the name asm.inc.c to asm.c
Le: 10/01/2010 par Clara objet de la modification contrôle de l'accent
uation
                des commentaires en ISO-8859-1
Le: 11/02/2010 par JPL Update for MS Visual C10 beta 2
Le: 10/03/2010 par JPL objet de la modification simplification de l'en
tête
Le: 17/03/2012 par JPL objet de la modification vérification des comme
ntaires
                au format UTF-8

                                        rappels: options booléennes à d
éfinir
      -- dans main() ou winmain()
```

```
                    copier autant de tabulations qu'il y en a dans le
 source */


    Fonction cpp_(); Fichier cpp.c

    Documentation simplifiée.
    -------------------------
    La fonction  doit décoder des commentaires, pré-codés avec un cara
ctère
    ASCII qui suit le début de commentaire. exemple //A ou /*A
    ... et bla bla bla ...(raccourci pour les exercices)
    Tests unitaires:
    ----------------
    Vérifiez les options n, s et t séparément:
    Vérifiez que la numérotation des lignes est correcte dans tous les
 cas de
    décodage (lignes et blocs). n, ns. nt, nst.
    Vérifiez que le texte reste à la bonne place dans tous les cas de
décodage:
    toute la ligne (dans la cas de la directive de compilation FULL_LI
NE), ou
    texte seul.
    -- Vérifiez que les tabulations sont bien prises en charge dans to
us les
       cas, décodage des lignes et des blocs.
    -- Vérifiez de même pour les espaces.
    La fonction doit être éprouvée de telle sorte qu'un commentaire qu
i se
    termine par le caractère de fin de fichier soit entièrement copié
dans le
    fichier cible.
    Le caractère de fin de fichier décodé avant le(les) caractère(s) d
e fin de
    commentaire doit être remplacé par un caractère de fin de ligne.

    Le caractère de fin de fichier (EOF) NE DOIT JAMAIS APPARAÎTRE DAN
S LE
    TEXTE DU FICHIER CIBLE.
    Ceci provoquait un bug dans la version Alpha de l'application fenê
trée
    mkdcppw à l'étape du test d'intégration.

 ------------------------------------------------------------------------
--------
```
© Contact: http://edeulo.free.fr/contacts/formmail.php
© EELL, Éditeurs Européens de Logiciels Libres, 2007
Association à but non lucratif selon l'Article 11 de la convention eur
opéenne
des droits de l'homme.
```
 ------------------------------------------------------------------------
--------

FILE NAMES: cpp.inc.c cpp.c cpp.inc.c++

PROJECT: mkd
```

```
mkd is a UNIX command to extract pre-encoded comments lines to generat
e the
software documentation according to ISO standards.
mkd is the abbreviation of make documentation. This command was origin
ally
known under the name mkdoc (make documentation).
This command is not integrated into the standard distributions of Unix
 / Linux

INITIAL PROJECT:
mkdoc 1989 for MS-DOS and UNIX now obsolete. CEM - University of Montp
ellier II

ADMINISTRATIVE RECORD:
extractdoc 04/12/2009

FILE UPDATE:
cpp.inc
-------
Date: ../../1986 by JPL Initial programming for MSDOS and UNIX on SUN
Date: ../../1991 by JPL mkdoc 3.12 for PC and UNIX
Date: ../../1995 by JPL mkdoc 3.22 for RED-HAT on LINUX
Date: ../../2004 by JPL mkdoc 4.00 for LINUX ubuntu
Date: 11/02/2010 by JPL Update for MS Visual C10 beta 2
cpp.c
-----
Date: ../../.... by JPL Transformation into independent function for m
kdcpp
                 Move the name cpp.inc.c to cpp.c
cpp.inc.c++ and cpp.c
---------------------
Date: ../03/2012 by CLARA Update the comments to UTF-8 format and C++
for
                 mkdcppw. Version name for mkdcppw: cpp.inc.c++
Date: 02/05/2012 by CLARA Special adaptation for mkdcppw with gtkmm
Date: 10/03/2013 by JPL Rewrite comments in english (en) to facilitate
 the
                 translations

                     w appearance of codes in lines if codes[] flags a
re false
                                                          w depends of c
odes[]
                                     CR/LF under DOS
                                  (c1='*' voir c2)
                                     c1 = always '*'

 Fichier puchar, macro en assembleur pour MS-DOS
 Macro trouvée sur wikipedia:fr:Assembleur#Macro-assembleur
 Pour l'exercice suivant:
    wikibooks:fr:Mkd_(Extracteur_de_documents)/Exercices#Fichiers_de_p
rojet
```

Mettre à jour de la documentation logicielle globale

(Conception détaillée, Tests unitaires, du cycle en V) Code 'D'
exercice.docu

Aperçu

mkd -fjst D exercice.prj *.docu

```
exercice.docu

    Fonction asm_(); Fichier asm.c


        fonction asm_
 ------------------------------------------------------------------------
--------
 NOM DU FICHIER; asm.c

 ACTION:
      La fonction asm_ lit le fichier écrit en assembleur et extrait l
es
      renseignements de structure, l'organigramme, la documentation de
stinée
      aux programmeurs, au fichier d'entête (.h) ou à la documentation

      destinée à l'utilisateur etc.
      On utilise ici les codes d'identification des commentaires suiva
nts :
      D: pour la documentation générale sur les fonctions (listing)
      H: pour générer le fichier d'entête (header, .h ou .hpp)
      O: pour l'organigramme
      S: pour le contrôle de la structure du programme
      T: pour les points de tests
      U: pour la documentation utilisateur
      Options :
      n : numéro de ligne (ajoute le numéro de ligne)
      s : écran (ajoute le commentaire à l'écran)
      t : texte seul (ne copie pas les caractères de repérage)

 SYNTAXE:
      #include "version.h"
      #include "asm.h"
      int asm_(FILE *pfdoc, FILE *pnfile);

 PORTABILITE:
      Win32     Win64   UNIX/Linux x86 et amd64

 DESCRIPTION:
      FILE *pfdoc : pointeur sur le flux du fichier de destination
      File *pnfile : pointeur sur le flux du fichier source

 VALEUR RETOURNEE:
      Ne retourne rien
```

```
 DROIT DE COPIE:
      © mkd, EUPL 2007, précisée dans version.h



    Fonction cpp_(); Fichier cpp.c



   function cpp_
 ------------------------------------------------------------------------
--------
 FILE NAMES: cpp.c and cpp.inc.c++ with gtkmm for mkdcppw
   ( Comments in UTF-8 with Bluefish text editor )

 ACTION:
   The cpp_ function reads the source file (pnfile) transmitted from t
he
   calling function, and d□codes the comments pre-encoded in lines or

        blocks of style c++. and then writing this comments in a targe
t file
        (pfdoc).
        Pre-coded characters are defined in a external global table 'C
odes';

   The golbal variables are 'Codes' and 'Options'.
   The 'Codes': table of 5 characters:
      extern char codes[];
      They must be defined in the calling function:
      char codes[5] = {0,0,0,0,0};
   The 'Options': n,s,t,v.
      extern unsigned char n,s,t;
      They must be defined in the calling function:
      unsigned char n=0,s=0,t=0;
   With the options:
   n: The transcript is preceded by line number. This allows to easily

      reach the commented line.
   s: Add the comment to the screen to use shell redirections > , >> ,

      or || etc.
   t: With the t option only the commented text is copied.
      Without the t option the entire line or block is copied.
      The t option permit to generate directly exploitable and publish
able
      documents.

   Remark:
        If the decoded comment begins with the characters "/ *", the
comment
        is copied until find the characters "* /", whatever included a
ny
        comment-line starting with "//".
        If the decoded comment begins with the characters "//", the li
ne is
```

```
        copied until find the end-of-line or new-line 'NL' character o
r
        end-of-file 'EOF'.
        This provisions facilitate the automatic generation of header
files
        (file.h ; .hpp ; etc.) and documentation of functions.

 SYNTAX:
   In UNIX / LINUX environnement:
   #include "/usr/include/mkd/version.h" // IMPORTANT: Compilation dir
ectives
   #include "/usr/include/mkd/cpp.h"
   int cpp_(FILE * pfdoc, FILE * pnfile);

 PORTABILITY:
   Microsoft Visual studio under Windows : x86(Win32) x64(Win32 and WI
N64)
   gcc under Unix / Linux.

 DESCRIPTION:
   cpp_ fonction
   FILE * pfdoc: pointer of the target file opened by the calling func
tion
   FILE * pnfile: pointer of the source file opened by the calling fun
ction

 RETURN VALUE:
   Return 0 in case of success in konsole version, nothing with gtkmm.

 COPYRIGHT: (Specified in version.h)

 SEE ALSO MANUAL:
   Man(3) attached in English.
   Command line : man 3 cpp_



 Fichier putchar.asm sous DOS/Windows
 Fichier putchar.s sous Unix/Linux
 macro putchar
 ------------------------------------------------------------------------
-------
 putchar est une macro MASM qui affiche un caract?re sous MS-DOS.
 On l'utilisera par exemple ainsi
        putchar "X"
 Et cela génèrera :
        mov    dl,"X"
        mov    ah,2
        int    21h
```

Notes et références

Mkd (Extracteur de documents)

MAINTENANCES, MISES À JOUR, ÉVOLUTIONS

Troisième partie :

INTERNATIONALISATIONS

MANUELS ET MESSAGES

Internationalisation des manuels

Maintenance

Cette page est susceptible de subir des modifications occasionnelles

Make documents

Internationalisation des manuels

Les manuels pour UNIX sont écrits en standard au format nroff
Les manuels traduits doivent être compressés mkd.1.gz avant d'être transférés dans les répertoires des langues:
Les manuels de la version de mai 2012 existent en allemand, anglais et français. Vous pouvez envisager de les traduire dans d'autres langues à partir des références ci-dessous.

Exemple d'installation des manuels (Fichier de commandes)

Pour d'autres langues que que anglais et français voir aussi le chapitre #Autres_langues

Installation des manuels

```
#!/bin/bash
# File: install_manuals,
# command syntax: sudo (or su) install_manuals
echo "install or update mkd manual mkd (1)"
chmod 444 mkd_en.1.gz
chmod 444 mkd_fr.1.gz
# standard (en) US
cp -f mkd_en.1.gz /usr/share/man/man1/mkd.1.gz ;
#
# (en) GB
# if [ -d "/usr/share/man/en_GB/man1" ]; \
#       then sudo cp -f mkd_en_GB.1.gz /usr/share/man/en_GB/man1/mkd.1
.gz ; \
# fi
#
# (fr)
if [ -d "/usr/share/man/fr.UTF-8/man1" ]; \
        then sudo cp -f mkd_fr.1.gz /usr/share/man/fr.UTF-8/man1/mkd.1
.gz ; \
elif [ -d "/usr/share/man/fr/" ]; \
        then sudo cp -f mkd_fr.1.gz /usr/share/man/fr/man1/mkd.1.gz ;
\
fi
#
# (de)(es)(it)
# see maintainers web site
```

Manuels de référence

- (anglais) Mkd-Manual (english reference manual)
- (français) Mkd, manuel de référence en français

Manuels traduits

- allemand Mkd-Handbuch

Manuels souhaités (à traduire)

- espagnol Mkd-Manual en Español Votre participation est souhaitée.
- italien Mkd-Manuale italiano Votre participation est souhaitée.

Mises à jour des manuels internationalisés

- (man1) :

  - Téléchargez la mise à jour des manuels[1]
  - Download the latest multi-lang Installer:[2]

- **Attention :** Vérifiez les dates de publication de ces pages de manuels en bas de page.

1. Manuels (http://edeulo.free.fr/wiki/index.php/Shell_command_projects/mkd#Manuels_.2F_Manuals) anglais allemand français
2. Translations (http://edeulo.free.fr/pub/mkd/) anglais allemand français

Internationalisation des messages

> **Maintenance**
>
> Cette page est susceptible de subir des modifications occasionnelles

Internationalisation des messages

Make documents

Les traductions automatiques des messages des applications dépendent directement de la langue que vous avez adoptée à votre connexion. Voyez la ligne LANG votre environnement (Commande env dans un terminal sous Unix/Linux).

Si vous avez adopté la langue française vous devez trouver LANG=fr_FR.UTF-8 sous Linux.
Si les fichiers de langue ont été installés pour chacune des applications, dont la langue française que vous utilisez, vous devez avoir les messages en français, sinon, ils doivent apparaître dans la langue universelle en informatique; l'anglais.
Les messages de la version de mai 2012 existent en allemand, anglais et français. Vous pouvez envisager de les traduire dans d'autres langues à partir des références ci-dessous.

Pour arriver à ce résultat on balise les messages des fichiers sources avec gettext exemple:

```
printf(gettext("My name is %s.\n"), my_name);
```

Les messages seront enregistrés dans un fichier .pot avec une première ligne qui est le message d'origine et une deuxième ligne qui servira à la traduction dans un fichier file.po dans la langue voulue:
(file est un nom que vous aurez choisi, sans doute le nom de votre application.)
Pour plus de détails voyez **gettext** [1] [2]

file.pot sert de référence pour créer les fichiers .po dans différentes langues.

```
#: src/name.c:36
msgid "My name is %s.\n"
msgstr ""
```

Pour effectuer la traduction on utilise l'éditeur **poedit** pour obtenir le fichier file.po. Lorsque l'on effectue plusieurs traductions il faut les différencier dans des répertoires différents ou avec des noms différenst comme file_en.po, file_fr.po.

file_fr.po pour la traduction en français:

```
#: src/name.c:36
msgid "My name is %s.\n"
msgstr "Je m'appelle %s.\n"
```

Mise à jour et installation des traductions pour mkd 2012

Notez que la version 2012 n'est disponible que pour UNIX ou LINUX, et que toutes les commandes citées sont disponibles ou ont des équivalents pour les sytèmes d'exploitation courants.

Principe d'installation pour tous les systèmes d'exploitation:

- Télécharger[3] poedit pour Windows ou Mac ou Linux, selon votre système d'exploitation.
- Avec le fichier .pot (Portable object template) créer le fichier .po (Portable object) avec la commande:

  fr est à remplacer par une autre langue maternelle.
  Commande Linux : **msginit -l fr -o mkd_fr.po -i mkd.pot**

- Convertir le fichier .po au format UTF-8:

  Commande Linux : **iconv -f ISO-8859-1 -t UTF-8 filename.txt > UTF8.filename**
  et dans l'entête du fichier .po, modifier « ASCII » pour « UTF-8 » ou un format ISO[4].

- Editer le fichier .po avec poedit et faites la traduction dans votre langue maternelle. Poedit créera le fichier .mo
- Installer le fichier .mo dans le répertoire de la langue correspondante.

  Exemple pour la langue française : (Remplacer « fr » par la langue traduite)
  sudo ou **su** *selon la version de Linux*
  Commande Linux : **sudo cp mkd_fr.mo /usr/share/locale/fr/LC_MESSAGES/mkd.mo**
  Si vous désirez remplacer un fichier .mo existant la commande doit être cp -f

- *Les fichiers .mo sont lisibles avec* ***Virtaal****.*

Fichier Pot (Potfile)

mkd.pot Référence pour la version mkd-0~120508

```
# SOME DESCRIPTIVE TITLE.
# Copyright (C) YEAR THE PACKAGE'S COPYRIGHT HOLDER
# This file is distributed under the same license as the PACKAGE packa
ge.
# FIRST AUTHOR <EMAIL@ADDRESS>, YEAR.
#
#, fuzzy
msgid ""
msgstr ""
"Project-Id-Version: mkd 12.03\n"
```

Exemple de traduction en français

ATTENTION Cette traduction en français peut avoir été modifiée et n'est peut-être plus la traduction officielle (http://edeulo.free.fr/telechargements/console-ubuntu/po/mkd_fr.po).

Vous pouvez copier la traduction ci-dessous pour créer une version personnelle, ludique par exemple.

```
# French translations for mkd package.
# Copyright (C) 2012 THE mkd'S COPYRIGHT HOLDER
# This file is distributed under the same license as the mkd packag
e.
# Blabla <blabla at free.fr>, 2012.
#
msgid ""
msgstr ""
"Project-Id-Version: mkd 12.03\n"
"Report-Msgid-Bugs-To: \n"
"POT-Creation-Date: 2012-03-31 16:31+0200\n"
"PO-Revision-Date: 2012-03-31 18:00+0100\n"
"Last-Translator: Blabla <blabla at free.fr>\n"
"Language-Team: French\n"
"Language: fr\n"
"MIME-Version: 1.0\n"
"Content-Type: text/plain; charset=UTF-8\n"
"Content-Transfer-Encoding: 8bit\n"
"Plural-Forms: nplurals=2; plural=(n > 1);\n"

#: ../mkd.c:132
#, c-format
msgid "mkd PC version, Release %s, USAGE:\n"
msgstr "mkd version PC, mise à jour %s, USAGE:\n"

#: ../mkd.c:133
#: ../mkd.c:139
#, c-format
msgid "syntax: mkd [-ABCFPSafjlnpstvw] char_codes path_source [path
_target]\n"
msgstr "syntaxe: mkd [-ABCFPSafjlnpstvw] codes chemin_source [chemi
n_cible]\n"

#: ../mkd.c:134
msgid "    or: mkd ? .See also mkd.doc or manual."
msgstr "    ou: mkd ? .Voir aussi mkd.doc ou le manuel."

#: ../mkd.c:138
#, c-format
msgid "mkd UNIX version, Release %s, USAGE:\n"
msgstr "mkd version UNIX, mise à jour %s, USAGE:\n"

#: ../mkd.c:140
```

```
msgid "    or: mkd \\? .See also nanual: 'man mkd'"
msgstr "    ou: mkd \\? .Voir aussi le manuel: 'man mkd'"

#: ../mkd.c:145
#, c-format
msgid " --> options:\n"
msgstr " --> options:\n"

#: ../mkd.c:146
#, c-format
msgid "      -A decode Assembler comment style only\n"
msgstr "      -A décode uniquement les commentaires de style Assemb
leur\n"

#: ../mkd.c:148
#, c-format
msgid "      -B        Basic style\n"
msgstr "      -B        style Basic\n"

#: ../mkd.c:150
#, c-format
msgid "      -C        C++ style\n"
msgstr "      -C        style C: C, c++, php ...\n"

#: ../mkd.c:151
#, c-format
msgid "      -F        Fortran style\n"
msgstr "      -F        style Fortran\n"

#: ../mkd.c:152
#, c-format
msgid "      -P        Pascal style\n"
msgstr "      -P        style Pascal\n"

#: ../mkd.c:154
#, c-format
msgid "      -S        Shell style\n"
msgstr "      -S        style Shell\n"

#: ../mkd.c:156
#, c-format
msgid "      -a append target file\n"
msgstr "      -a ajoute au fichier cible\n"

#: ../mkd.c:157
#, c-format
msgid "      -f:\n"
msgstr "      -f:\n"

#: ../mkd.c:160
#, c-format
msgid "          find language ( .ASM .BAS .C .FOR .PAS .PRO )\n"
msgstr "          trouve le langage ( .ASM .BAS .C .FOR .PAS .PRO )
```

```
\n"

#: ../mkd.c:164
#, c-format
msgid "          find language ( .s .S .c .h .i .f .F .r .p .sh .cs
h )\n"
msgstr "          trouve le langage ( .s .S .c .h .i .f .F .r .p .s
h .csh )\n"

#: ../mkd.c:167
#, c-format
msgid "      -j use only with project sources file.\n"
msgstr "      -j précise que le fichier source est un fichier de pr
ojet.\n"

#: ../mkd.c:168
#, c-format
msgid "      -l and p;   line:  (compil.: %c or %c in first column
or %c in line)\n"
msgstr "      -l et -p:   ligne;  (compil.: %c ou %c en première co
lonne ou %c dans la ligne)\n"

#: ../mkd.c:169
#, c-format
msgid "                  page:  (compil.: begin with %c and end wit
h %c)\n"
msgstr "                   page:  (compil.: commence avec %c et se t
ermine avec %c)\n"

#: ../mkd.c:170
#, c-format
msgid "      -n insert line number\n"
msgstr "      -n insère un numéro de ligne\n"

#: ../mkd.c:171
#, c-format
msgid "      -s copy and add to screen\n"
msgstr "      -s copie dans le fichier cible et à l'écran\n"

#: ../mkd.c:172
#, c-format
msgid "      -t copy the comment only\n"
msgstr "      -t ne copie que le commentaire\n"

#: ../mkd.c:173
#, c-format
msgid "      -v verbose\n"
msgstr "      -v mode bavard\n"

#: ../mkd.c:174
#, c-format
msgid "      -w overwrite (default option: switch off)\n"
msgstr "      -w écrase le fichier cible (par défaut cette option e
```

```
st invalidée)\n"

#: ../mkd.c:175
#, c-format
msgid " --> char_codes: all ASCII "
msgstr " --> codes: tout caractère ASCII "

#: ../mkd.c:178
#, c-format
msgid "except space:' '  (5 char max)\n"
msgstr "exeptÃ© le carctère espace:' '  (5 caractères max)\n"

#: ../mkd.c:179
#, c-format
msgid "                 example codes = UM or *OPTw or *HOS or ** f
or all\n"
msgstr "                  exemple de codes = UM, ou *OPTw, ou *HOS,
ou ** pour tout\n"

#: ../mkd.c:183
#, c-format
msgid " (5 char max)\n"
msgstr " (5 caractères au maximum)\n"

#: ../mkd.c:184
#, c-format
msgid "                 example codes = UM or \\*OPTw or '* HOS' or
 '**' for all\n"
msgstr "                  exemple de codes = UM,ou \\*OPTw, ou '* HO
S', ou '**' pour tout\n"

#: ../mkd.c:187
#, c-format
msgid " --> path_source: source file (option j: if it is a project
file) \n"
msgstr " --> chemin_source: fichier source (option -j: si le fichie
r source est un fichier de projet) \n"

#: ../mkd.c:188
#, c-format
msgid " --> path_target: target file\n"
msgstr " --> chemin_cible: chemin du fichier cible\n"

#: ../mkd.c:191
#, c-format
msgid "Example: mkd -Csnv '*S' file.c \\*.verif_struct\n"
msgstr "Exemple: mkd -Csnv '*S' fichier.c \\*.vérif_structure\n"

#: ../mkd.c:195
#, c-format
msgid "Example: mkd -Cs *Snv file.c *.str\n"
msgstr "Exemple: mkd -Cs *Snv fichier.c *.str\n"
```

```
#: ../mkd.c:197
#: ../mkd.c:642
#, c-format
msgid ".Exit 2\n"
msgstr ".Sortie 2\n"

#: ../mkd.c:259
#, c-format
msgid "Syntax error: Too many 'Language' options:\n"
msgstr "Erreur de syntaxe: Trop d'options de 'Langage':\n"

#: ../mkd.c:260
#, c-format
msgid ""
" A or B or C or...please.\n"
".Exit 2\n"
msgstr ""
" A ou B ou C ou...s'il vous plaît.\n"
".Sortie 2\n"

#: ../mkd.c:274
#, c-format
msgid "Options a or w please... "
msgstr "Options -a ou -w s'il vous plaît... "

#: ../mkd.c:286
#, c-format
msgid "Force language comment, style "
msgstr "Force le décodage des commentaires dans le style "

#: ../mkd.c:287
#, c-format
msgid "ASSEMBLER\n"
msgstr "ASSEMBLEUR\n"

#: ../mkd.c:288
#, c-format
msgid "BASIC\n"
msgstr "BASIC\n"

#: ../mkd.c:289
#, c-format
msgid "C\n"
msgstr "C\n"

#: ../mkd.c:290
#, c-format
msgid "FORTRAN\n"
msgstr "FORTRAN\n"

#: ../mkd.c:291
#, c-format
msgid "PASCAL\n"
```

```
msgstr "PASCAL\n"

#: ../mkd.c:292
#, c-format
msgid "SHELL\n"
msgstr "SHELL\n"

#: ../mkd.c:295
#, c-format
msgid "Options a=%d f=%d j=%d l=%d n=%d p=%d s=%d t=%d w=%d\n"
msgstr "Options a=%d f=%d j=%d l=%d n=%d p=%d s=%d t=%d w=%d\n"

#: ../mkd.c:343
#: ../mkd.c:428
#, c-format
msgid ""
"Syntax error '*' in path_source.\n"
".Exit 2\n"
msgstr ""
"Erreur de syntaxe '*' dans le chemin_source.\n"
".Sortie 2\n"

#: ../mkd.c:352
#, c-format
msgid ""
"Arg. syntax error '\\:'\n"
".Exit 2 \n"
msgstr ""
"Erreur de syntaxe dans l'argument '\\:'\n"
".Sortie 2 \n"

#: ../mkd.c:381
#, c-format
msgid ""
"Syntax error: path_source, suffixe '.doc'\n"
".Exit 2\n"
msgstr ""
"Erreur de syntaxe dans le chemin_source: suffixe '.doc'\n"
".Sortie 2\n"

#: ../mkd.c:439
#, c-format
msgid ""
"Arg. syntax error '\\:'\n"
".Exit 2\n"
msgstr ""
"Erreur de syntaxe dans la ligne de commande '\\:'\n"
".Sortie 2\n"

#: ../mkd.c:466
#, c-format
msgid ""
"Path_target: syntax error at or near '\\'\n"
```

```
".Exit 2\n"
msgstr ""
"Chemin_cible: erreur de syntaxe à ou après '\\'\n"
".Sortie 2\n"

#: ../mkd.c:483
#, c-format
msgid ""
"Path_target: syntax error at or near '/'\n"
".Exit 2\n"
msgstr ""
"Chemin cible: erreur de syntaxe à ou après '/'\n"
".Sortie 2\n"

#: ../mkd.c:533
#: ../mkd.c:578
#, c-format
msgid "Path target: suffixe not determined.\n"
msgstr "Chemin cible: le suffixe est indéterminé.\n"

#: ../mkd.c:605
#, c-format
msgid "Syntax error: (path_source : '%s') == (path_target = '%s')\n
"
msgstr "Erreur de syntaxe: (chemin_source : '%s') == (chemin_cible
= '%s')\n"

#: ../mkd.c:618
#, c-format
msgid "doc file : '%s' \n"
msgstr "fichier doc : '%s' \n"

#: ../mkd.c:629
#, c-format
msgid ""
"\n"
" OVERWRIT '%s' ...  y?\n"
msgstr ""
"\n"
" ECRASER '%s' ...  y?\n"

#: ../mkd.c:633
#, c-format
msgid ".Exit 1\n"
msgstr ".Sortie 1\n"

#: ../mkd.c:641
#, c-format
msgid "%s is write protected.\n"
msgstr "%s est protégé en écriture.\n"

#: ../mkd.c:660
#, c-format
```

```
msgid ""
"fopen file '%s' error for append\n"
".Exit 2\n"
msgstr ""
"erreur d'ouverture du fichier '%s' pour ajouter\n"
".Sortie 2\n"

#: ../mkd.c:674
#, c-format
msgid ""
"fopen file '%s' error for overwrit\n"
".Exit 2\n"
msgstr ""
"erreur d'ouverture du fichier '%s' pour écraser\n"
".Sortie 2\n"

#: ../mkd.c:681
#, c-format
msgid "error option -a or -w not found\n"
msgstr "erreur option -a ou -w pas trouvé\n"

#: ../mkd.c:689
#, c-format
msgid "project file : '%s' \n"
msgstr "fichier de projet : '%s' \n"

#: ../mkd.c:697
#, c-format
msgid ""
"project file '%s' not found\n"
".Exit 2\n"
msgstr ""
"fichier de projet '%s' pas trouvé\n"
".Sortie 2\n"

#: ../mkd.c:715
#, c-format
msgid ""
"\n"
"\n"
"file for doc: '%s'\n"
msgstr ""
"\n"
"\n"
"fichier pour doc: '%s'\n"

#: ../mkd.c:742
#, c-format
msgid "file '%s' not found for read\n"
msgstr "fichier '%s' par trouvé en lecture\n"

#: ../mkd.c:749
#: ../mkd.c:802
```

```
#, c-format
msgid ""
"%s\n"
"(file %s :)"
msgstr ""
"%s\n"
"(fichier %s :)"

#: ../mkd.c:750
#: ../mkd.c:803
#, c-format
msgid ""
"\n"
"\n"
"(file %s :)\n"
msgstr ""
"\n"
"\n"
"(fichier %s :)\n"

#: ../mkd.c:751
#: ../mkd.c:804
#, c-format
msgid "Options a=%d f=%d j=%d l=%d n=%d p=%d s=%d t=%d w=%d"
msgstr "Options a=%d f=%d j=%d l=%d n=%d p=%d s=%d t=%d w=%d"

#: ../mkd.c:761
#, c-format
msgid "Error, language not determined !.\n"
msgstr "Erreur. Le language n'a pas été trouvé !.\n"

#: ../mkd.c:765
#: ../mkd.c:775
#, c-format
msgid ""
"Closure error: '%s'\n"
".Exit 2\n"
msgstr ""
"Erreur de fermeture: '%s'\n"
".Sortie 2\n"

#: ../mkd.c:784
#, c-format
msgid ""
"\n"
"\n"
"file for doc: '%s' \n"
msgstr ""
"\n"
"\n"
"fichier pour doc: '%s' \n"

#: ../mkd.c:793
```

```
#, c-format
msgid ""
"file '%s' not found for read\n"
".Exit 2\n"
msgstr ""
"fichier '%s' pas trouvé en lecture\n"
".Sortie 2\n"

#: ../mkd.c:816
#, c-format
msgid ""
"Error, programming language not found !.\n"
".Exit 2\n"
msgstr ""
"Erreur. Le langage n'a pas été trouvé !.\n"
".Sortie 2\n"

#: ../mkd.c:822
#, c-format
msgid ""
"Closure error: '%s'\n"
".Exit 2 \n"
msgstr ""
"Erreur de fermeture: '%s'\n"
".Sortie 2 \n"

#: ../mkd.c:830
#, c-format
msgid ""
"Closure '%s' error\n"
".Exit 2\n"
msgstr ""
"Erreur de fermeture '%s'\n"
".Sortie 2\n"

#: ../mkd.c:834
#, c-format
msgid ""
"\n"
"\n"
"DOC %s END\n"
msgstr ""
"\n"
"\n"
"DOC %s FIN\n"
```

Autres langues

Traductions effectuées:

- allemand mkd_de.po
- (anglais) mkd_en.po *Page originale*

- (français) mkd_fr.po *Cette page*

Traductions souhaitées:

- espagnol mkd_es.po Votre participation est souhaitée.
- italien mkd_it.po Votre participation est souhaitée.

Notes et références

1. GNU Operating System (http://www.gnu.org/software/gettext/)
2. gettext pour Windows 32 (http://gnuwin32.sourceforge.net/packages/gettext.htm)
3. http://www.poedit.net/download.php
4. http://www.gnu.org/software/gettext/manual/gettext.html#Creating

Mkd (Extracteur de documents)

MAINTENANCES, MISES À JOUR, ÉVOLUTIONS

Quatrième partie :
CODER
TESTER
INTEGRER
DOCUMENTER

Exemple de programme pour le cycle en V

Ajouter des modules

Make documents

Maintenance

Cette page est susceptible de subir des modifications occasionnelles

Pourquoi ajouter des modules ?

Il est aisé d'extraire des fichiers, des lignes complètes, contenant une chaîne déterminée de caractères. Voir la commande **Grep** et **Awk**.

Les fichiers de textes sont de plus en plus compliqués, il est difficile d'extraire des blocs de commentaires débarrassés des signes de balisage; d'où l'intérêt de créer des modules adaptés aux langages de programmation ou d'éditions utilisant des balises; **Postscript**, **SGML** et autres méta langages; **Xml, Html,** ... .

Comment ajouter un module ?

Quel langage de programmation ?

Tous les langages de programmation peuvent convenir à condition qu'ils soient compilable avec gcc pour produire des fichiers objets liables entre eux (linkables)

Définir le module

Il faut déterminer quel est le style de commentaire à extraire. Repérer éventuellement un fichier similaire existant. Il existe en standard des modules d' *extraction de lignes* et des modules d' *extraction de blocs*.

Certains fichiers ont des commentaires plus compliqués comme : <!-- texte --> avec quatre caractères voire plus. Dans des cas plus complexes il pourrait être préférable de repérer des chaînes de caractères plutôt que des caractères ASCII. Ceci aurait l'avantage d'être adapté aux caractères asiatiques en UTF 8, 16, ou 32 avec des chaînes de caractères données paramètres d'entrée.

Créer le pseudo-code

Le **pseudo code** que nous utilisons est un commentaire qui va décrire ce que l'on fait et ce que l'on doit obtenir, ce qu'il faut faire dans certaines conditions comme créer une boucle jusqu'à ce que l'on obtienne ce que l'on cherche.
Il y a de nombreux ouvrages qui décrivent les manières de programmer. Le plus célèbre est sans aucun doute la méthode **Merise** pour la **programmation impérative**.

Il est préférable d'écrire le pseudo code avant d'écrire les lignes de codes.

```
 //0 Ceci est un pseudo code style C++
 std::cout << "Unable to open file "<< fprj << std::endl; // messag
e to stdout
```

Examinez les fichiers à disposition pour avoir une meilleure idée de l'utilisation du pseudo code.

Intégration des lignes de codes

Langue de programmation et internationalisation

Nous utilisons **gettext** pour adapter les messages aux différentes langues parlées.

Pour que l'on puisse facilement traduire les messages il convient de les écrire en anglais US (ASCII). Les divers caractères accentués des langues européennes ne sont pas si faciles à traduire.

Compilation de l'unité de test

Les options du fichier version.h

Le fichier d'entête version.h est utilisé pour compiler la série d'applications dérivées de la commande mkd.
La vérification des options est indispensable avant toute compilation.

version.h for mkd* series

```
/*
* © mkd, version 2009.12 et ultérieures
*
* Concédée sous licence EUPL, version 1.1 ou – dès leur approbation pa
r la
* Commission européenne - versions ultérieures de l'EUPL (la «Licence»
).
* Vous ne pouvez utiliser la présente œuvre que conformément à la Lice
nce.
* Vous pouvez obtenir une copie de la Licence à l'adresse suivante:
*
* http://ec.europa.eu/idabc/eupl5
*
* Sauf obligation légale ou contractuelle écrite, le logiciel distribu
é sous
* la Licence est distribué «en l'état»,
* SANS GARANTIES OU CONDITIONS QUELLES QU'ELLES SOIENT, expresses ou i
mplicites.
* Consultez la Licence pour les autorisations et les restrictions ling
uistiques
* spécifiques relevant de la Licence.
*/

/*P
        NOM DU FICHIER: version.h
        PROJET INITIAL: mkd console linux & MSDOS
        DOSSIER INITIAL : mkdoc3.12 Centre d'Electronique de Montpelli
er
        PROGRAMMEUR: JPL
        DATE: 18/12/2009
        MODIFICATIONS:
        le: 09/01/2011 par JPL objet: Ajouté l'option de compilation p
our MSDEV41
        le: 10/01/2012 par JPL objet: Ajouté la définition std_C pour
compilation UNIX STD
        le: 25/03/2012 par Clara objet: internationalisation
        le: 27/03/2012 par jpl objet: internationalisation en utf-8
        le: 10/04/2013 par Cardabela: Modification mineure. Alignement
 des commentaires.
*/
//      le: ../../.. par .............. objet de la modification ....

/**# MKD.C ############################*/ /*H*/
/*U            UTILISATEURS            */ /*H*/
```

```
/*O             ORGANIGRAMME             */ /*H*/
/*S               STRUCTURE              */ /*H*/
/*T                 TESTS                */ /*H*/
/*P              PROGRAMMEUR             */ /*H*/
/*www     remarques particulieres       */ /*H*/
/**************************************/ /*H*/

// define VERSION line 173

#define UNIX            /*O UNIX version */ /* Delete this line for PC
 version */
//#define GTKMM         /*O Version fenêtrée avec gtkmm pour mkdcppw u
niquement */

#ifndef UNIX            /*H*/
#define PC              /*O version de compilation PC   */  /*H*/
/*#define PC_ONLY*/     /*H find_ ne reconnaît pas: .s .f .p .sh .csh
 */
#define NL "\r\n"       /*H*/
//#define VC10                  /*O Microsoft Visual C 10.- */
#define MSDEV41
#endif                  /*H*/
                        /*H*/
#ifdef UNIX             /*H*/
#define UNIX_ONLY       /*H find_ ne reconnaît pas: .BAS .PAS .FOR ...
 */
#define NL "\n"         /*H*/
#endif                  /*H*/
                        /*H*/

/*H #D ***************************************************************
**********************/
/*H *** #Define OPTIONS DE COMPILATION:  options par défaut, redéfinis
sables: **************
          (#D pour l'extraction des options de compilation en ligne av
ec l'option CD3='#') */
/*H****************** #Define 'l' (ligne) commence par #define en 1ère
 colonne : ***********/
#define CD1 '%'   /*H #Define Option CD1 en 1ère colonne  prend fin av
ec NL                */
#define CD2 '-'   /*H #Define Option CD2 en 1ère colonne  prend fin av
ec NL                */
#define CD3 '#'   /*H #Define Option CD3 dans la ligne    prend fin av
ec NL                */
                  /*H #Define Option '%' commentaire postcript
                     */
                  /*H #Define Option '#' commentaire shell ou pour Mak
efile voir option S
                      #D            ( le commentaire se termine avec n
ew_line )            */
                  /*H #Define Option '\'' commentaires Basic
                     */
                  /*H #Define Option ';'  commentaires assembleur
                     */
/*H ***************** #Define Option 'p' (dans la page) #define en 1èr
e colonne : **********/
```

```
#define CD4 '\"'  /*H #Define Option CD4 = "  debut de commentaire
                      */
#define CD5 '\"'  /*H #Define Option CD5 = "  fin de commentaire
                      */
                  /*H #Define ************************************
**********************/
#define FULL_LINE /*H #Define Option de compil. copier toute la ligne
                      */
/*H #D ************************************************************
**********************/


/*U*** MKD ********************************************************
*********
*M name: mkd  (make documentation)
*U (programme extracteur de documents, Manuel, Organigramme, Structure
,
*U warnings, points de Test, notes pour la maintenance du Programmeur,
 etc... )
*M origine: Centre d'Electronique de Montpellier
*M          Univ. Sciences & Techniques du Languedoc
*M          34060 MONTPELLIER CEDEX
*P programmeur R3.12: JPLouyot, adr électronique: louyot@obelix.cnusc.
fr
*P programme R3.12 pour systèmes UNIX mkdoc, et version PC IBM mkd.exe
*U
*U Ce programme range dans un fichier.doc par défaut, ou dans tout fic
hier
*U  nommé en dernier paramètre:
*U avec les options l et p:( CD1,CD2,CD3,CD4,CD5, sont définis dans ve
rsion.h )
*U -- la ligne COMMENÇANT par les caractères CD1 ou CD2 ou CD3 et suiv
is par
*U    un des 5 caractères donnes en paramètre char_codes (avec option
l).
*U -- la ligne CONTENANT le caractère défini par CD3 suivi par un des
5
*U    caractères donnes en paramètre char_codes (avec option l).
*U -- les lignes CONTENANT et SUIVANT le caractère défini par CD4 et s
uivi par
*U    un des 5 caractères donnes en paramètre char_codes (avec option
p)
*U    JUSQU'À la ligne qui contient le caractère CD5. Attention, si le
 caractère-
*U    caractère CD5 est NL la ligne suivante sera ignorée.
*U
*U avec les options: A, B, C, F, P, et S:
*U    le tri est effectue conformément a la définition des commentaire
s de
*U    chaque langage, A:Assenbleur, B:Basic, c:C, F:Fortran, P:Pascal,
 S:Shell
*U
*U avec char_code = '**' tous les commentaires correspondants aux lang
ages ou
*U    aux options l et p sélectionnés sont copies.
*U
```

```
*U syntaxe de la ligne de commande:
*U syntaxe pour pc IBM:
*U   mkd [-ABCFPSajlnpstvw] char_codes [drive:]path_source [[drive:]pa
th_target]
*M syntax for UNIX version:
*M   mkdoc [-ABCFPSajlnpstvw] char_codes path_source [path_target]
*M
*M   -- path_source : path source file (or project file: option j )
*M   -- char_codes  : all ASCII
*M   -- options     : -A Assembler source file or similar   (  ;  ->
NL   )
*M                    -B Basic source file or similar       (REM or '
-> NL)
*M                    -C C source file, or similar (PROLOG) ( / *  ->
 * / )
*M                    -F Fortran source file or similar     (c,C or *
-> NL)
*M                    -P Pascal source file                 ( {to}, (*
to*) )
*M                    -S Shell files or similar             (   # -> N
L  ' )
*M                    -a append target file
*M                    -f find langage source file (useful with project
 file)
*M                    -j if path_source is a project file only
*M                    -l line read line only
*M                    -n insert line number
*M                    -p page read text
*M                    -s screen view
*M                    -t comment only
*M                    -v verbose mode
*M                    -w overwrite target file
*M                     ( default option: overwrite switch off)
*M   -- path_target : path target file
*M
*M examples for UNIX version:
*M ++++++++++++++++++++++++++
*M mkdoc -Csn '*S' mkdoc3.c verif.struct
*M mkdoc -Fst UM version.h /usr/local/docu/mkdoc.docu
*M mkdoc -Cst M version.h /usr/local/man/manl/mkdoc.l
*M mkdoc -lt '*UM' mkdoc3.c \*.manual   with CD1='*' or CD2='*'
*M mkdoc -jv '**' mkd_docu.prj  (for all comments, sources in prj file
)
*M mkdoc -pjnv '**' mkd_docu.prj (for all strings, sources in prj file
,
*M                                with CD4=CD5='\"')
*U exemples pour version PC:
*U ++++++++++++++++++++++++++
*U mkd -Csn *SOn \mkdoc3.c vérif.str (char espace:' ' n'est pas valabl
e)
*U mkd -Fst UM a:version.h c:\man\mkd.doc
*U mkd -Fst M a:version.h d:mkd.man
*U mkd -lt *UM mkdoc.c *.MAN (pour utilisateurs et manuel, with CD1='*
', CD2='*')
*U mkd -jv ** file.prj   (pour tous commentaires, sources dans le fich
ier projet)
```

```
*U or mkd -pjnv '**' mkd_docu.prj  (for all strings, sources in .prj f
ile,
*U                                  with CD4=CD5='"')
*U
*U Pour l'ASSEMBLEUR utiliser ; avec l'option A
*U                              (ou l'option l avec CD3=';')
*U Pour le BASIC (ou BATCH) utiliser ' ou REM avec l'option B
*U                              (ou option l avec CD3='\'')
*U Pour le C et PROLOG utiliser / *  et  * / avec l'option C
*U Pour le FORTRAN utiliser C, c, ou * en 1ère colonne avec l'option F
*U                              ( ou option l avec CD1='c' CD2='C' CD3='*
' )
*U Pour le PASCAL utiliser les accolades { et } ou (* et *)
*U                              ( ou option p avec CD4='{' et CD5='}' )
*U Pour le CSHELL utiliser # avec l'option S
*U                             ( ou option l avec CD3='#' cas fin de ligne
 = NL )
*U Pour les fichiers projet avec des sources UNIX: .s .c .i .f .r .p m
élangés,
*U      ou .ASM .C .FOR .PAS etc... sources PC, utiliser l'option f (f
ind).
*U Pour les commentaires de fichiers POSCRIPT compiler avec CD3='%'
*************************************************************************
********/
                                    /*H*/
#define VERSION "12.03"             /*H*/
                                    /*H*/
#define MAX     85                  /*H longueur des buffers des path sour
ce et doc */
#define STX     0x2                 /*H Start Text */
#include <stdio.h>                  /*H*/
                                    /*H*/
#ifdef PC                           /*H*/
#include <conio.h>                  /*H*/
#include <process.h>                /*H*/
#include <io.h>                     /*H*/
#endif                              /*H*/
#ifdef VC10                         /*H Microsoft Visual C 10.- */
        #define putch _putch        /*H ISO C++ */
        #define getch _getch        /*H ISO C++ */
        #define access _access      /*H ISO C++ */
#endif                              /*H*/
                                    /*H*/
#ifdef MSDEV41                      /*H*/
        //#define STD_C             /*H SVr4, 4.3BSD, C89, C99.*/
        #include <string.h>         /*H*/
#endif                              /*H*/
                                    /*H*/
#ifdef UNIX                         /*H*/
        #define bool int
        //#define STD_C             /*H SVr4, 4.3BSD, C89, C99.*/
        /*#define getch() getchar()*/
        #define putch putchar       /*H*/
#include <signal.h>                 /*H*/
#include <sys/file.h>               /*H*/
```

```
#include <string.h>                /*H*/
#endif                             /*H*/
```

La fonction maîtresse - main()

La fonction main doit ouvrir les fichiers source et cible et transmettre l'adresse de ces fichiers ouverts à la fonction à éprouver (tester).

Elle doit en retour, s'il y a lieu, d'afficher la valeur de retour de fonction à défaut de d'afficher un message correspondant à cette valeur de retour.

Il n'est pas interdit d'ouvrir "en dur" toujours les mêmes fichiers source et cible, cela dépend des tests demandés.
L'essentiel pour les tests unitaires est de répondre aux directives. Il peut cependant être nécessaire d'entrer des arguments comme n, s, t par exemple pour l'épreuve de la fonction cpp_() que nous prenons souvent en exemple.

```
#include "version.h"
// #include "" // entête du fichier à tester app.h si la fonction e
st indépendante comme cpp.c
// #include "" // fichier de la fonction app.i si la fonction doit
être incluse
// Variables globales:
     unsigned char A,B,C,F,P,S,a,f,j,l,n,p,s,t,v,w;
     char codes[5] = {0,0,0,0,0};
//*0 Début de la fontion main():
     int main(int argc, char **argv)
     {
        //0 Déclarations
        int ret=0;
        char fsrc[MAX]; /* Nom du fichier source  */
        char fdoc[MAX]; /* Nom du fichier cible   */
        FILE *pnfile;   /* FICHIER A EXTRAIRE   */
        FILE *pfdoc;    /* FICHIER A DOCUMENTER */

        //0 Pseudo code des instructions
        .....
        return ret;
      }
//*0 Fin de la fontion main()
```

Exemple complet avec mkdcpp (Éditeur ISO-8859-1)

mkdcpp.c

```
//0 Programme de tests de la fonction cpp_()
#include "version.h"
#include "cpp.h"
```

Le Makefile pour la compilation de l'unité

Le makefile suivant, mkdcpp.mak complet pour une installation standard a été bridé pour les tests unitaires.

mkdcpp.mak

```
# file: mkdcpp.mak (Makefile pour version LINUX UNBUNTU 10.11) mise à
jour septembre 2012
# syntaxe: make -f mkdcpp.mak -d (voir install...)
# gcc et mkdoc doivent être installés, si mkdoc n'est pas installé on
peut utiliser mkdcpp
# avec la ligne de commande : mkdcpp --t -$(LANG) manuel | mv manuel.t
xt $(PGM).1
#       avec cette commande il faut hélas penser à faire un retour cha
riot à la fin de la commande
#
# Attention: avant de compiler, il faut valider les options de compila
tion
# dans version.h (notament la langue et la version UNIX qui invalide l
a version PC)
#
# Use BINDIR /bin or /usr/bin or /usr/local/bin
# Use CATMANDIR = /usr/share/man/man1 or /usr/local/share/man
# LANG F for Fr or M for Eng

 ####################
 ### UNIX version ###
 ####################

CC      = gcc
MKDOC = mkd

PGM     = mkdcpp
BINDIR  = #/usr/bin
BINMODE = 755

MAN     = mkdcpp.1.gz
LANG    = F
#CATMANDIR = /var/cache/man/cat1
#MANDIR  = /usr/share/man/man1
MANDIR  = /usr/local/man
MANMODE = 644

SRCS    = mkdcpp.c cpp.c
HDRS    = version.h cpp.h
OBJS    = cpp.o mkdcpp.o
LIBS    =
CFLAGS  =  -O
LDFLAGS =

SPLINTFLAG = -weak -nestcomment
```

Épreuve unitaire

Générer la documentation unitaire

- La documentation unitaire doit (devrait ?) être intégrée dans le fichier source afin d'être certain que cette documentation concerne CE fichier source.
- On extrait cette documentation par copier-coller ou avec mkd.
- Pour une production importante il faut se référer à la **norme ISO**[1][2]..
- Pour une petite production, cette documentation unitaire doit fournir les indications suivantes:

1. **FILE NAMES**: Nom du fichier et nom de la fonction.
2. **PROJECT**: Nom du projet.
3. INITIAL PROJECT: Projet initial si le projet a changé de nom.
4. ADMINISTRATIVE RECORD: Nom d'enregistrement dans les archives avec la date, le lieu etc.
5. **UPDATES**: Date, auteur, nature de la modification.
6. **RECOMMANDED TESTS**: Description des épreuves pour valider la fonction.
7. Description de la fonction: (Utile pour le manuel man 3)
    1. **NAME**: Nom de la fonction (Suivi, éventuellement du nom du fichier source, Voir FILES ci après)
    2. **SYNOPSIS**: ou **SYNTAX**: fichiers d'entête (#include) return-value function (parameters);
    3. **DESCRIPTION**: ou **ACTION**: action détaillée et sa description.
    4. **RETURN VALUE**: valeurs de retour de la fonction.
    5. **CONFORMING TO**: ou **STANDARDS**: POSIX, ANSI C, BSD, ISO/IEC 9899:2011, gcc, MS-VC10, UTF-8, etc.
    6. **PORTABILITY**: LINUX-Debian-systems, LINUX-Red-Hat, Windows NT, etc.
    7. COPYRIGHT: Droit de copie s'il y a lieu.
    8. AUTHORS: Concepteur, traducteurs, etc. (Éventuellement)
    9. RESSOURCES: Si le projet nécessite des ressources particulières (gtkmm, etc.)
    10. **FILES**: Fichiers concernés Fichiers d'entête (.h), sources, etc. (où on les trouve)
    11. NOTES: ou recommandations.
    12. BUGS: Erreurs constatées ou erreurs courantes.
    13. SEE ALSO: Voir aussi, au cas où ...

Éprouver le module (Test unitaire)

Le répertoire des tests unitaire se compose:

1. Un ou plusieurs fichiers sources. le fichier source peut simplement s'appeler "Source" mais on préfère lui donner un nom qui le distinguera des autre modules lors des tests d'intégration; par exemple "Test_cpp_U.cc"
2. Un fichier de commande des test que l'on peut appeler "Tests" ou "Test_cpp_U" que l'on rendra exécutable sous linux, sous Windows on utilise l'extension ".bat"
3. Un fichier de nettoyage "Clean" ou Test_cpp_U.clean. On le rendra exécutable (Même remarque que précédemment)
4. Un fichier qui donne le résultat de l'analyse "Analysis" en anglais.

Exemple pour le Styles C

- /* ... */ : C, C++, C#, PHP, Java, Javascript, CSS, ...
- // ... NL : C++, C#, PHP, Java, Javascript

Fichiers de tests unitaires de la fonction cpp_() dans mkd

Tests function files: Test_cpp_U.cc, Test_cpp_U.c
Bash commands files: Tests, Tests_Clean

Unit testing of the cpp_() function in mkd

```
//T File Test_cpp_U.cc to test function cpp_()
/*P
    Fonction cpp_(); Fichier cpp.c

    Directives de tests unitaires version alpha 2013:
    -------------------------------------------------
    Vérifiez les options n, s et t séparément:
    Vérifiez que la numérotation des lignes est correcte dans tous les
 cas de
    décodage (lignes et blocs). n, ns. nt, nst.
    Vérifiez que le texte reste à la bonne place dans tous les cas de
décodage:
    toute la ligne (dans la cas de la directive de compilation FULL_LI
NE), ou
    texte seul.
    -- Vérifiez que les tabulations sont bien prises en charge dans to
us les
       cas, décodage des lignes et des blocs.
    -- Vérifiez de même pour les espaces.
    La fonction doit être éprouvée de telle sorte qu'un commentaire qu
i se
    termine par le caractère de fin de fichier soit entièrement copié
dans le
    fichier cible.
    Le caractère de fin de fichier décodé avant le(les) caractère(s) d
e fin de
    commentaire doit être remplacé par un caractère de fin de ligne.

    Le caractère de fin de fichier (EOF) NE DOIT JAMAIS APPARAÎTRE DAN
S LE
    TEXTE DU FICHIER CIBLE.
    Ceci provoquait un bug dans la version Alpha de l'application fenê
trée
    mkdcppw à l'étape du test d'intégration.
*/

/*T
12345678901234567890123456789012345678901234567890123456789012345678
1234567890 */
#define TEST //T The macro becomes active
#ifdef TEST  //T Do something (Faire quelque chose)
```

```
#undef test  //T The macro becomes inactive

/*T
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890 */
                //T test in ligne with 2 tabulations
     //T test in ligne with 5 spaces

/*T test bloc in first line end bloc*/

/*T test bloc in first line
                with 2 tabs
     with 5 spaces end bloc */

/*T
12345678901234567890123456789012345678901234567890123456789012345678901234567890
1234567890 */
                /*T test bloc in line with 2 tabs end bloc*/
     /*T test bloc in line
                2 tabs
     with 5 spaces
     end bloc */
//T see also "/*T" with out end of block comment and EOL in file Test_
cpp_U.c
//T test inline with out end of line (EOL)(NL)(CR/LF)
```

File Test_cpp_U.c

```
//T File Test_cpp_U.c
/*T with out end of block comment and EOL (NL) in file Test_cpp_U.c
```

File Tests

```
#!/bin/bash
#O mkd tests under Linux.
#O Epreuve de vérification de la fonction cpp_() avec mkd

#O    Copy Test_cpp_U.cc in target created file
      cat Test_cpp_U.cc > Test_cpp_U.tstcpp

#O    Test options -nstv with source "Test_cpp_U.cc" and target "*.tst
cpp"
#O    - and screen redirection to tstcpp.screen

#O 1: Test de l'option -s append target
      mkd -savC T Test_cpp_U.cc *.tstcpp > tstcpp.screen

#O 1: Test option -t
      mkd -tavC T Test_cpp_U.cc *.tstcpp # >> tstcpp.screen

#O 1: Test option -n, with append files target and screen
      mkd -navC T Test_cpp_U.cc *.tstcpp # >> tstcpp.screen
```

```
#O 2: Test options -ns, with append files target and screen
      mkd -nsavC T Test_cpp_U.cc *.tstcpp >> tstcpp.screen

#O 2: Test options -nt, with append files target and screen
      mkd -ntavC T Test_cpp_U.cc *.tstcpp >> tstcpp.screen

#O 2: Test options -nst, with append files target and screen
      mkd -nstavC T Test_cpp_U.cc *.tstcpp >> tstcpp.screen


#O 5: Independant test with -nstv "Test_cpp_U.cc" and target "*.tstcpp
lastline1"
      mkd -nstvwC T Test_cpp_U.cc *.tstcpplastline1 > tstcpplastline.s
creen1

#O 5: Independant test with -nstv "Test_cpp_U.c" and target file "*.ts
tcpplastline2"
#w bug      mkd -nstvwC T Test_cpp_U.c *.tstcpplastline2 > tstcpplastl
ine.screen2

#O    Break with gedit "Test_cpp_U.tstcpp", "tstcpp.screen"
      gedit Test_cpp_U.tstcpp
      gedit tstcpp.screen
      gedit Test_cpp_U.tstcpplastline1
#w bug      gedit Test_cpp_U.cc.tstcpplastline2

#O    Erase Test_cpp_U.tstcpp
```

File Tests_Clean

```
#!/bin/bash
#O File Tests_Clean
#O Force delete files created with mkd "tests" under Linux.
      rm -f Test_cpp_U.tstcpp
      rm -f tstcpp.screen
      rm -f Test_cpp_U.tstcpplastline1
      rm -f tstcpplastline.screen1
      rm -f Test_cpp_U.tstcpplastline2
      rm -f tstcpplastline.screen2
      rm -f *.doc
```

Analyse des tests unitaires

Éprouver le module unitaire nécessite de répondre aux recommandations des tests contenus dans la documentation.

Exemple de fichier Analysis

Analysis

Intégration du module

L'intégration du module se fait par la compilation d'une application qui utilise ce "module", cette fonction: mkd, mkdcpp, mkdcppw par exemple. Le fichier d'entête version.h doit être modifié en fonction des compilateurs et options de compilation.

Ce module peut être intégré par inclusion (#include) avec l'option de compilation gtkmm pour mkdcppw.

Il peut être compilé séparément sous forme d'objet éventuellement intégrable dans une librairie.
Le module objet sera lié (Link) à l'application par la compilation.
Attention : Une fonction intégrée dans une librairie est figée. Il faut alors veiller à proposer un fichier d'entête correspondant à l'objet; on ne peut plus utiliser version.h.

> Pour la fonction cpp() par exemple, il faudra proposer un fichier d'entête cpp.h correspondant à la compilation cpp.o

Épreuve d'intégration

- Exemple:

1. Constater que le module se compile bien dans l'application.
2. Vérifier que le module répond bien aux paramètres d'entrée et qu'il prend en compte les variables globales. Vérifier les valeurs de retour de fonction et l'affichage des messages correspondants.
3. Reprendre tous les tests unitaires des fonctions avec les fonctions intégrées dans l'application.

Générer la documentation

- Déterminer à qui s'adresse cette documentation, par exemple:

1. Utilisateur final
   Manuels
2. Concepteurs
   Suivi de la programmation selon les étapes du cycle en V
3. Programmeurs
   Documentation des fonctions et librairies de fonctions
   Tests unitaires, d'intégration, de validation

Notes et références

1. ISO/IEC 26514:2008 (http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=43073)
2. ISO/IEC 26514:2008 prévisualisation (http://webstore.iec.ch/preview/info_isoiec26514%7Bed1.0%7Den.pdf)

Mkd (Extracteur de documents)

MAINTENANCES, MISES À JOUR, ÉVOLUTIONS

Cinquième partie :

CONSTRUCTION D'UN PAQUET DEBIAN

Exercice de Construction d'un Paquet 'debian'

Dépôt parrainé debian-ubuntu du paquet original mkd-131215

Exercice de Construction d'un Paquet 'debian' pour la maintenance logicielle

Make documents

Exercice de construction d'un paquet debian.

Cet exercice a pour but de se familiariser avec la construction d'un paquet debian.
Ce paquet n'est pas censé persister dans la structure linux. *Paquet officiel : mkddocu*[1].
Voyez la construction d'un paquet mkd officiel (version 131215)

Cette page a été optimisée pour "Version imprimable". Si vous désirez imprimer cette page assurez-vous que les textes déroulants ne sont pas masqués.

Les paquets d'installation des logiciels pour les systèmes d'exploitation linux permettent de sécuriser les systèmes grâce à des clés numériques. Ils permettent aussi de faciliter l'installation des applications.

La description qui suit ne conduit pas à la construction d'un paquet sécurisé

Créer un paquet debian

La lecture de la documentation sur le sujet est si obscure qu'il est bien difficile de créer son premier paquet.
L'objet de cet article est de voir, au travers d'un exemple simple, comment sortir de l'obscurité.

Le paquet debian que nous allons créer ne sera pas sécurisé et ne pourra pas être installé par le gestionnaire de la logithèque. Il sera simplement installé et désinstallé par des commandes shell.

Le paquet

1. L'idée de la démonstration est de créer un utilitaire très simple à l'usage des programmeurs avec l'aide de la commande **mkd**.
2. Nous nous sommes inspirés de l'article *myecho*[2] pour vous proposer cette commande que nous appellerons ***mkmaintainerdocu*** *(Make maintainer documentation)*
3. Cette commande n'a pas pour ambition de persister dans les répertoires des commandes binaires (/bin, /usr/bin, etc.)

Que doit-il faire ?

1. créer un fichier de projet pour une application écrite en langage C
2. générer le fichier d'entête de l'application "<projet>.h
3. générer la documentation des fonctions
4. générer la documentation destinée aux programmeurs
5. Si possible afficher les fichier à la fin de leur création ou modification.

Choix de l'exécutable

Le plus simple est de créer une commande shell bash↑, facile à maîtriser et inspirée par *myecho* cité dans les références.

Structure de l'archive projet-mkmaintainerdocu.tar.gz

"./" est le **répertoire conteneur** que nous pouvons appeler **projet-mkmaintainerdocu** Il contient nos utilitaires et les fichiers d'installation. **"./mkmaintainerdocu"** est le **répertoire racine** de notre paquet. Ce qui nous intéresse se trouve DANS ce répertoire.

```
Créer le contenu de l'archive. Les fichiers à créer sont décrits ci-après :
./mkmaintainerdocu/usr
./mkmaintainerdocu/usr/share
./mkmaintainerdocu/usr/share/doc
./mkmaintainerdocu/usr/share/doc/changelog.debian - fichier à créer
./mkmaintainerdocu/usr/share/doc/copyright - fichier à créer
./mkmaintainerdocu/usr/share/doc/changelog - fichier à créer
./mkmaintainerdocu/usr/share/doc/README - fichier à créer
./mkmaintainerdocu/usr/local
./mkmaintainerdocu/usr/local/bin
./mkmaintainerdocu/usr/local/bin/mkmaintainerdocu - fichier à créer
./mkmaintainerdocu/DEBIAN
./mkmaintainerdocu/DEBIAN/postrm - fichier à créer
./mkmaintainerdocu/DEBIAN/control - fichier à créer
./mkmaintainerdocu/DEBIAN/postinst - fichier à créer
On y ajoutera les utilitaires décrits ci-après :
./empaqueter - pour créer le paquet - fichier à créer
./installer - pour installer l'application - fichier à créer
./desinstaller - pour désinstaller le paquet - fichier à créer
./KONSOLE - utilitaire pour créer une console locale - fichier à créer
Fichiers qui seront créés à l'empaquetage :
./mkmaintainerdocu.deb
```

changelog.debian

- fichier nécessaire à la création du paquet, peut être vide.

```
First version : 1.0 (2013-12-15)
```

copyright

- fichier nécessaire à la création du paquet, peut être vide.

```
First version : 1.0 (2013-12-15) (c) Demo <demo@demo.com>
```

changelog

- fichier nécessaire à la création du paquet, peut être vide.

```
First version : 1.0 (2013-12-15)
```

README

- fichier utile à la construction du paquet

```
This demo command is writed to create a self debian package.
```

mkmaintainerdocu

- notre fichier shell qui va exécuter nos commandes de création des documents

```
#!/bin/bash
# File mkmaintainerdocu writed by Clara Jimenez 2013-12-15
# Demo : Create any simple debian package for mkd and C programming
 language

# remove old files :
rm *.c~ # ATTENTION !
# create project file only with C files :
ls -1 *.c > project-prj
# create ./maintainer-documentation directory if not exist :
install -d ./maintainer-documentation

# Make header file project.h and header document; style C, overwrit
e, text only
echo " make header file './maintainer-documentation/project.h' "
mkd -Cjwt H project-prj ./maintainer-documentation/project.h
echo " make fuctions documentation './maintainer-documentation/proj
ect-functions-documentation.txt' "
mkd -Cjwt D project-prj ./maintainer-documentation/project-function
s-documentation.txt

# Make individuals tests directives document; style C, overwrite, a
dd line number, text only
echo " make tests document, directives for all functions './maintai
ner-documentation/\
project-functions-directives.txt' "
mkd -Cjwnt T project-prj ./maintainer-documentation/project-functio
ns-directives.txt

# Make functions updates documentation; style C, overwrite, add lin
e number, text only
echo " make functions maintainers documentation '/maintainer-docume
```

```
ntation/\
project-functions-maintainers-doc.txt' "
mkd -Cjwnt P project-prj ./maintainer-documentation/project-functio
ns-maintainers-doc.txt

# Get all strings and select strings for translations (add 2012-02-
13)
echo " make list of strings for tranlations"
mkd -pnjs '**' project-prj ./maintainer-documentation/strings.txt |
 grep gettext > ./maintainer-documentation/strings-for-translations
.txt

echo " Make mkd tests : "
echo "  See in working directory. "
```

postrm

- shell exécuté après la suppression du paquet (si nécessaire)
- cet exécutable peut exécuter des mises à jour comme 'cat', 'whatis', et d'autres ...

```
#!/bin/bash
echo " If this message appears then 'mkmaintainerdocu' is now remov
ed "
```

control

- fichier nécessaire à la construction du paquet

```
Version: 1.0
Section: devel
Priority: optional
architecture: all
Depends: bash, mkd
Maintainer: Demo <demo@demo.com>
Description: Make maintainer documentation for C files. See mkd man
ual.
Suggests: mkdcppw
Recommends: mkd
```

- Le classement des paquets doit être choisi dans une des sections :
  admin, devel, doc, graphics, libs, misc, net, otherofs, pyton, shells, sound, text, utils, x11.

postinst

- shell exécuté après la création du paquet (si nécessaire)
- cet exécutable peut exécuter des mises à jour comme 'cat', 'whatis', et d'autres ...

```
#!/bin/bash
echo " Use 'mkmaintainerdocu' in your working directory "
echo "  Then type 'mkmaintainerdocu' in your terminal or konsole "
echo "  See manual man1 'mkd' and, or, 'mkmaintainerdocu' "
```

utilitaires

empaqueter

```
#!/bin/bash
fakeroot chmod 755 mkmaintainerdocu/usr/local/bin/mkmaintainerdocu
fakeroot chmod 755 mkmaintainerdocu/DEBIAN/post*
fakeroot chmod 755 mkmaintainerdocu/DEBIAN/pre*
fakeroot rm *.deb
fakeroot dpkg-deb --build mkmaintainerdocu
```

installer

```
#!/bin/bash
sudo dpkg -i mkmaintainerdocu.deb
```

desinstaller

```
#!/bin/bash
sudo dpkg --remove mkmaintainerdocu
ou :
# sudo dpkg --purge mkmaintainerdocu
```

KONSOLE

Cet utilitaire facilite l'ouverture d'une console dans le répertoire courant.

```
#!/bin/bash
echo -e '\E['32';'01'm'"click on New Line to 'start' the Konsole"
# echo -e '\E['31';'01'm' "not 'start in a terminal'"
echo -e '\E['32';'01'm'"click on Ctrl-C to exit"
tput sgr0 # Reset text attributes to normal without clear
read pwd
pwd
echo $pwd
tput sgr0 # Reset text attributes to normal without clear
/usr/bin/konsole background-mode --workdir pwd dir
read
```

Contenu extrait du paquet

- L'extraction avec le *gestionnaire d'archives* recrée les répertoires avec la même structure.

```
./mkmaintainerdocu
./mkmaintainerdocu/usr
./mkmaintainerdocu/usr/share
./mkmaintainerdocu/usr/share/doc
./mkmaintainerdocu/usr/share/doc/changelog.debian
./mkmaintainerdocu/usr/share/doc/copyright
./mkmaintainerdocu/usr/share/doc/changelog
./mkmaintainerdocu/usr/share/doc/README
./mkmaintainerdocu/usr/local
./mkmaintainerdocu/usr/local/bin
./mkmaintainerdocu/usr/local/bin/mkmaintainerdocu
./mkmaintainerdocu/DEBIAN
./mkmaintainerdocu/DEBIAN/postrm
./mkmaintainerdocu/DEBIAN/control
./mkmaintainerdocu/DEBIAN/postinst
```

Tests

Boîte des tests

Fichiers pour les tests

main.c

```
/*P
     FILE NAME : main.c
     PROJECT   : myproject
     UPDATES   :
     2013-12-15 Created by me <myemail@emailserver.com>
     ....-..-.. Updated by ..... for ......

*/
/*T
     FILE NAME : main.c
     UNIT TESTS DIRECTIVES : Blablabla0 ...

*/
/*D
     FILE NAME : main.c
     FUNCTION NAME : Main project ; <myproject>
     SYNOPSYS, SYNTAX :
      #include myproject.h
      void main (void);
      int function0(char* string0);
      char* string0 : <description>
     ACTION, DESCRIPTION ....
     CONFORMING TO : gcc, UTF-8
     PORTABILITY : .....
     RETURN VALUE :
     COPYRIGHT :
     AUTHORS :
     RESSOURCES :
     NOTES :
     BUGS : email <email@mailer>
     SEE ALSO MANUAL : man 1 myprojetc

*/
/*H
     // FILE NAME : main.c
     // function0 (included in main.c)
     int function0(char*);

*/

     int function0(char* string0){
     .....
     }

     void main(void) {
     }
```

fonction1.c

```
/*P
     FILE NAME : function1.c
     PROJECT   : myproject
     UPDATES   :
     2013-12-15 Created by me <myemail@emailserver.com>
     ....-..-.. Updated by ..... for ......

*/
/*T
     FILE NAME : function1.c
     UNIT TESTS DIRECTIVES : Blablabla1 ...

*/
/*D
     FILE NAME : function1.c
     FUNCTION NAME :
     SYNOPSYS, SYNTAX :
      #include myproject.h
     .....

*/
/*H
     // FILE NAME : function1.c
     int funtion1(char*);

*/

     int function1(char* string1) {
     .....
     }
```

fonction2.c

```
/*P
     FILE NAME : function2.c
     PROJECT   : myproject
     UPDATES   :
     2013-12-15 Created by me <myemail@emailserver.com>
     ....-..-.. Updated by ..... for ......

*/
/*T
     FILE NAME : function2.c
     UNIT TESTS DIRECTIVES : Blablabla2 ...

*/
/*D
     FILE NAME : function2.c
     FUNCTION NAME :
     SYNOPSYS, SYNTAX :
      #include myproject.h
    .....

*/
/*H
     // FILE NAME : function2.c
     int function2(char*, int);

*/

     int function2(char* string2, int num) {
     .....
     }
```

Résultat des tests

Liste de l'archive myproject.tar.gz

```
./myproject/
./myproject/main.c
./myproject/project-prj
./myproject/fonction2.c
./myproject/fonction1.c
./myproject/maintainer-documentation
./myproject/maintainer-documentation/project-functions-documentation.t
xt
./myproject/maintainer-documentation/project-functions-directives.txt
./myproject/maintainer-documentation/project.h
./myproject/maintainer-documentation/project-functions-documentation.t
xt~
./myproject/maintainer-documentation/project-functions-maintainers-doc
.txt
```

Fichier project.prj

```
fonction1.c
fonction2.c
main.c
```

Fichier d'entête (header)

project.h:

```
    // FILE NAME : function1.c
    int funtion1(char*);


    // FILE NAME : function2.c
    int funtion2(char*, int);


    // FILE NAME : main.c
    // function0 (included in main.c)
    int function0(char*);
```

Directives de programmation et de tests

- project-functios-directives.txt

```
 9
10       FILE NAME : function.c
11       UNIT TESTS DIRECTIVES : Blablabla1 ...
12
13
 9
10       FILE NAME : function2.c
11       UNIT TESTS DIRECTIVES : Blablabla2 ...
12
13
 9
10       FILE NAME : main.c
11       UNIT TESTS DIRECTIVES : Blablabla0 ...
12
```

Manuel des fonctions

- project-functions-documentation

```
FILE NAME : function1.c
FUNCTION NAME :
SYNOPSYS, SYNTAX :
 #include myproject.h
.....


FILE NAME : function2.c
FUNCTION NAME :
SYNOPSYS, SYNTAX :
 #include myproject.h
.....


FILE NAME : main.c
FUNCTION NAME : Main project ; <myproject>
SYNOPSYS, SYNTAX :
 #include myproject.h
 void main (void);
 int function(char* string0);
 char* string0 : <description>
ACTION, DESCRIPTION ....
CONFORMING TO : gcc, UTF-8
PORTABILITY : .....
RETURN VALUE :
COPYRIGHT :
AUTHORS :
RESSOURCES :
NOTES :
BUGS : email <email@mailer>
SEE ALSO MANUAL : man 1 myprojetc
```

Document à l'usage des mainteneurs

```
1
2       FILE NAME : function1.c
3       PROJECT   : myproject
4       UPDATES   :
5       2013-12-15 Created by me <myemail@emailserver.com>
6       ....-..-.. Updated by ..... for ......
7
8
1
2       FILE NAME : function2.c
3       PROJECT   : myproject
4       UPDATES   :
5       2013-12-15 Created by me <myemail@emailserver.com>
6       ....-..-.. Updated by ..... for ......
7
8
1
2       FILE NAME : main.c
3       PROJECT   : myproject
4       UPDATES   :
5       2013-12-15 Created by me <myemail@emailserver.com>
6       ....-..-.. Updated by ..... for ......
7
```

Sécurisation des paquets

Les paquets non sécurisés sont à proscrire.

Boîte de sécurisation des paquets

Créer un nouveau paquet sécurisé

Make-new-ppa : Pour créer un dépôt ppa.

```
#!/bin/bash
# File : Make-new-ppa
# Notes:
# <archive> a la forme : <exécutable>-<version>.tar.gz

# Nettoyage :
echo " Ne pas lancer en 'sudo', 'su' ou 'root'. Utiliser la 'Konsole'
ou le 'Terminal' "

# Conserver une trace du vieux répertoire DEBIAN, au cas où ...
echo " Déplacement le vieux répertoire DEBIAN si il existe et le mettr
e en mode lecture seule. "
mv <répertoire-origine>/debian ./debian.old
chmod 444 ./debian.old
echo "suppression éventuelle du vieux répertoire DEBIAN "
rm -Rf debian/

# Entrer dans le répertoire-origine. C'est le répertoire de vos source
s.
echo " entrant dans les sources de l'archive : <répertoire-origine>/sr
c/  "
cd <répertoire-origine>/src
# Nettoyage du répertoire des sources avec l'option -d : print debuggi
ng
echo " mettant à jour l'exécutable et le plaçant dans son répertoire .
./usr/bin "
make <exécutable> -d
rm *.o
cd ../../
# créer le fichier orig.tar.gz
echo " sortant de l'archive et créant le fichier compressé de l'archiv
e d'origine "
tar cvzf <répertoire-origine>.tar.gz  <répertoire-origine>
# attention au point qui se transforme en souligné sauf pour l'archive
 native
# 'mv' pas pour le natif.
# Rappel :  <répertoire-origine> = <exécutable>-<version>
# mv <exécutable>-<version>.tar.gz <exécutable>_<version>.orig.tar.gz

# Créer le nouveau répertoire DEBIAN selon la configuration et options
# -c : avec licence de copyright. -e : avec UNE adresse de courriel de
s mainteneurs.
# -s : seulement le binaire. -n : paquet natif. -f : <fichier> où <fic
hier> est le nom
#      de l'archive (compressée .tar.gz) que nous venons de créer ci-d
essus.
#       -s : si il faut compiler le binaire pour chaque architecture,
sinon -i pour 'all arch'.
# -r <format> d'exécution (rules) pour la création du paquet : cdbs, o
u dh7, ou old
cd <archive>
echo "entrant dans le répertoire <répertoire-origine> "
echo " créant les fichiers de base pour la construction du paquet dans
 ./debian/  "
dh_make -c GPL -e myemail@mailserver -s -n -f <archive-origine>.tar.gz
 -r CDBS
```

Il faut noter que toutes les options citées pour l'exécution de dh_make ne sont pas nécessaires. On peut supprimer la plupart des fichiers créés dans debian si on utilise une compilation simple de l'exécutable avec 'make'. Cependant, pour la portabilité sur d'autres machines avec des processeurs différents x86, amd64 etc. il est nécessaire de re-compiler l'exécutable; c'est le fichier exécutable 'rules' qui devra prendre en charge les différentes versions.

Compilation et envoi au dépôt ppa:

- Build-and-put-to-ppa

```
#!/bin/bash

# Création du fichier .changes où <clé> est votre clé publique PGP qui
 permet le cryptage et
#    le dépôt parrainé sécurisé
# L'option -sa précise  que le fichier source d'origine '.tar' est un
répertoire.
# L'option -S pour ne créer qu'un seul binaire.
echo " entrant dans l'archive : compilation cryptée d'un seul paquet s
ource.changes pour le dépôt "
cd <répertoire-origine>
debuild -S -sa -k<clé>
cd ..
echo " sortant de l'archive, envoi parrainé au dépôt ppa "
dput ppa:<your-ppa/<exécutable> <exécutable>_<version+1>_source.change
s

# Création du paquet crypté sans re-signer le fichier '.changes' ou/et
 '.dsc'
# -i (--info) directive pour lintian. Affiche les problèmes et les err
eurs.
# -us et -uc directives pour ne pas modifier les fichiers '.changes' e
t '.dsc'
echo " construction de paquets binaires "
cd <répertoire-origine>
debuild -i -us -uc -b
cd ..
```

Suggestions

- Si vous utilisez souvent les mêmes codes d'extraction, vous pouvez créer un shell pour la maintenance dans le répertoire /usr/local/bin, sans pour cela créer un paquet ....
- Les paquets sont réellement utiles lorsque l'installation est complexe ou pour une meilleure sécurité par un dépôt officiel (Voir Launchpad (https://launchpad.net/~jean-paul-louyot/+archive/exercises))

Liens utiles

Créer un paquet ubuntu (http://doc.ubuntu-fr.org/tutoriel/creer_un_paquet)

Liste 'complète' ? des pages décrivant l'utilisation des paquets sous Ubuntu (http://doc.ubuntu-fr.org/paquet)

Références

1. Dépôt officiel de mkddocu : mkd-doc_140515_all.deb (https://launchpad.net/~jean-paul-louyot/+archive/ubuntu/mkd/+files/mkd-doc_140515_all.deb)
2. http://alp.developpez.com/tutoriels/debian/creer-paquet/

Exemple de dépôt parrainé debian-ubuntu du paquet original mkd-131215

Make documents

Maintenance

Cette page a été optimisée pour "Version imprimable"

Les paquets d'installation des logiciels pour les systèmes d'exploitation linux permettent de sécuriser les systèmes grâce à des clés numériques. Ils permettent aussi de faciliter l'installation des applications.

Un paquet *original* est le premier paquet d'une série. Le paquet *natif* 131215 a été créé pour un nouveau dépôt ppa (Personnal packages)[1] personnalisé par l'équipe de maintenance de l'EELL (Éditeurs européens de logiciels libres)

Créer la hiérarchie des fichiers et répertoires de travail

Répertoires

- ~Packaging/mkd/mkd-131215/ répertoire de travail qui contient :

1. les fichiers de commandes
   Make-new-ppa, Buid-all, Rebuild-all.
2. le répertoire *origine* ./mkd-0~131215
   Contiendra le Makefile de création du paquet et les documents README, NEWS, etc.
3. le répertoire des sources ./mkd-0~131215/src
   Contiendra tous les fichiers nécessaires à la compilations des sources

```
./mkd-0~131215/src/find.inc.c
./mkd-0~131215/src/shell.c
./mkd-0~131215/src/asm.c
./mkd-0~131215/src/cpp.h
./mkd-0~131215/src/version.h
./mkd-0~131215/src/internationalisation.h
./mkd-0~131215/src/basic.h
./mkd-0~131215/src/tri.c
./mkd-0~131215/src/cpp.c
./mkd-0~131215/src/basic.c
```

```
./mkd-0~131215/src/fortran.h
./mkd-0~131215/src/fortran.c
./mkd-0~131215/src/mkd.h
./mkd-0~131215/src/shell.h
./mkd-0~131215/src/pascal.h
./mkd-0~131215/src/find.inc.h
./mkd-0~131215/src/mkd.c
./mkd-0~131215/src/asm.h
./mkd-0~131215/src/tri.h
./mkd-0~131215/src/pascal.c
```

4. le répertoire des binaires et des données ./mkd-0~131215/usr
On pourra prévoir de les créer avec un Makefile ou les placer dores et déjà dans la hiérarchie 'usr' lorsque ces fichiers ne nécessitent pas de nouvelle compilation : les manuels, langues, icônes etc. :

```
./mkd-0~131215/usr/share/mime/packages/mkd.xml
./mkd-0~131215/usr/share/doc
./mkd-0~131215/usr/share/man/fr.UTF-8/man1/mkd.1.gz
./mkd-0~131215/usr/share/man/fr
./mkd-0~131215/usr/share/man/de/man1/mkd.1.gz
./mkd-0~131215/usr/share/man/man1/mkd.1.gz
./mkd-0~131215/usr/share/icons/hicolor/256x256/apps/mkd.pngps
./mkd-0~131215/usr/share/icons/hicolor/16x16/apps/mkd.png
./mkd-0~131215/usr/share/icons/hicolor/32x32/apps/mkd.png
./mkd-0~131215/usr/share/icons/hicolor/48x48/apps/mkd.png
./mkd-0~131215/usr/share/locale/fr/LC_MESSAGES/mkd.mo
./mkd-0~131215/usr/share/locale/en/LC_MESSAGES/mkd.mo
./mkd-0~131215/usr/share/locale/de/LC_MESSAGES/mkd.mo
./mkd-0~131215/usr/bin
```

Notez que :

- le seul fichier qui nécessite d'être mis à jour et placé dans le répertoire usr/bin/, est le binaire *mkd* adapté au processeur. (i386, amd64, ...)

Création du sous répertoire debian

Le répertoire ./mkd-0~131215/debian est créé à l'aide du fichier de commande 'Make-new-ppa' que nous avons placé au dessus du répertoire *origine* pour simplifier la création des divers paquets.

```
#!/bin/bash
# File Make new ppa
# ne pas lancer en sudo, et utiliser la Konsole

# créer le fichier orig.tar.gz
tar cvzf mkd-0~131215.tar.gz mkd-0~131215
# attention au souligné
# pas pour le natif : mv mkd-0~131015.tar.gz mkd_0~131015.orig.tar.
```

```
gz
cd mkd-0~131215
dh_make -c GPL -e edeulo@free.fr -s -n -f mkd-0~131215.tar.gz -r CD
BS
echo "Lorsque le répertoire debian a été créé, mettre les fichiers
à jour:"
echo "Fichiers à mettre à jour:"
echo "changelog remplacer unstable par la version (precise ou saucy
)"
echo "control, copyright, rules, watch"
echo "Il faudra ensuite lancer Build-all"
```

Options dh_make

-c GPL; general public licence (gpl2, gpl3, lgpl, lgpl2, lgpl3, artistic, apache, bsd ou mit)
-e adresse électronique des mainteneurs
-s seulement le binaire
-n natif, ne pas générer d'archive .orig
-f mkd-0~131215.tar.gz Ce sera l'archive originale
-r CDBS directive pour créer le fichier de commande 'rules'

Fichiers nécessitant une explication

debian/changelog

```
mkd (0~131215) saucy; urgency=low

  * Initial Release.

 -- Clara <email@mailserver>  Thu, 19 Dec 2013 17:44:21 +0100
```

Notez que :

- à la création, l'option -n a initialisé la version "* Initial Release"
- nous avons modifié la première ligne pour préciser la version saucy (ubuntu 13.10)
- ici, dans ce cadre, l'adresse de courriel est masquée pour éviter les spams

debian/control

```
Source: mkd
Section: unknown
Priority: extra
Maintainer: Clara <email@emailserver>
Build-Depends: cdbs, debhelper (>= 8.0.0)
Standards-Version: 3.9.4
Homepage: <insert the upstream URL, if relevant>
#Vcs-Git: git://git.debian.org/collab-maint/mkd.git
#Vcs-Browser: http://git.debian.org/?p=collab-maint/mkd.git;a=summa
ry
```

```
Package: mkd
Architecture: any
Depends: ${shlibs:Depends}, ${misc:Depends}
Description: Application for software maintenance.
 mkd (commande unix), extracteur de documents.
 .
 mkd extrait la documenation logicielle incluse dans les fichiers s
ources.
 .
 mkd extract comments in source files and product maintainers docum
entation.
```

Notez que :

- le créateur de fichier a bien fait son travail ...
- nous avons ajouté la 'Description' qui sera affichée à l'installation par le gestionnaire d'archives
- d'autres champs peuvent être indispensables[2]
- si le copyright n'apparaît pas à l'installation avec le gestionnaire d'archives il est possible, si c'est indispensable, de l'ajouter à la fin la description longue.

debian/copyright

```
Format: http://www.debian.org/doc/packaging-manuals/copyright-forma
t/1.0/
Upstream-Name: mkd
Source: <http://edeulo.free.fr>

Files: *
Copyright: 1989-2001 Centre d'électronique de Montpellier <jpl@cem2
.univ-montp2.fr>
           1989-2013 Jean-Paul Louyot <emailjpl@mailserver>
License: GPL-3.0+

Files: debian/*
Copyright: 2013 Clara <emailclara@mailserver>
License: GPL-3.0+

License: GPL-3.0+
 This program is free software: you can redistribute it and/or modi
fy
 it under the terms of the GNU General Public License as published
by
 the Free Software Foundation, either version 3 of the License, or
 (at your option) any later version.
 .
 This package is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 GNU General Public License for more details.
 .
 You should have received a copy of the GNU General Public License
 along with this program. If not, see <http://www.gnu.org/licenses/
>.
 .
 On Debian systems, the complete text of the GNU General
 Public License version 3 can be found in "/usr/share/common-licens
es/GPL-3".
# Please also look if there are files or directories which have a
# different copyright/license attached and list them here.
# Please avoid to pick license terms that are more restrictive than
 the
# packaged work, as it may make Debian's contributions unacceptable
 upstream.
```

Notez que :

- l'option -c GPL a automatiquement créé la licence GPL-3+
- nous avons ajouté les droits de copie et les licences depuis l'origine du fichier
- ce droit de copie va apparaître dans les docs '/usr/share/doc/mkd/. ' et n'est pas vraiment nécessaire si on propose les fichiers HISTORY ou/et COPYING ou/et AUTHORS avec les droits détaillés.

debian/rules

```
#!/usr/bin/make -f

include /usr/share/cdbs/1/rules/debhelper.mk
include /usr/share/cdbs/1/class/makefile.mk


# Add here any variable or target overrides you need.

# Cette classe est destinée à ceux qui ont uniquement un fichier Ma
kefile (aucun autotools disponible) pour construire le programme. V
ous avez uniquement besoin de quatre règles dans le Makefile :
# - une pour nettoyer le répertoire de construction (c'est-à-dire «
 mrproper »)
# - une pour construire votre logiciel (c'est-à-dire « myprog »)
# - une pour vérifier que votre logiciel fonctionne correctement (c
'est-à-dire « check »)
# - une pour installer votre logiciel (c'est-à-dire « install »)
# Pour être honnête, la présence des règles d'installation n'est pa
s essentielle, mais cela aide toujours beaucoup lorsque vous les av
ez.

# The first operation, is to write the debian/rules. First, we add
the include lines:

# Add here any variable or target overrides you need.
# if you detect authors's loss of sanity, tell CDBS not to try runn
ing the nonexistent
# clean rule, and do the job yourself in debian/rules
DEB_MAKE_CLEAN_TARGET:= mrproper
DEB_MAKE_BUILD_TARGET:= mkd
DEB_MAKE_INSTALL_TARGET:= install
# no check for this software
##DEB_MAKE_CHECK_TARGET :=

# allow changing the makefile filename in case of emergency exotic
practices
##DEB_MAKE_MAKEFILE:= MaKeFiLe
# example when changing environnement variables is necessary :
##DEB_MAKE_ENVVARS:= CFLAGS="-fomit-frame-pointer"
##DEB_BUILD_OPTIONS is checked for the following options :
##- noopt: utilise -O0 au lieu de -O2
##- nocheck: skip the check rule
# If your Makefile doesn't support the DESTDIR variable, take a loo
k in it and find the variable responsible for setting installation
directory. If you don't find some variable to do this, you'll have
to patch the file...
# C'est tout :)
```

Notez que :

- le fichier a été créé par l'option '-r CDBS'
- nous avons ajouté les commentaires de la notice
- notre nettoyeur s'appele 'mrproper' dans le Makefile d'empaquetage. on aurait pu utiliser le traditionnel 'clean'
- mkd est l'exécutable qu'il faut générer pour chaque configuration de processeur
- install va copier les fichiers ./mkd-0~131215/usr/* dans le répertoire ./mkd-0~131215/debian/mkd/usr/* .

A titre indicatif, il est possible d'utiliser une variante install DESTDIR=debian/mkd à condition de le prévoir; cette solution permet de créer un fichier temporaire de test avec install DESTDIR=debian/tmp/ (Pas vraiment utile !)

debian/doc

```
NEWS
README
AUTHORS
COPYING
HISTORY
INSTALL
```

les fichiers NEWS à INSTALL seront copiés dans le répertoire ./mkd-0~131215/debian/mkd/usr/share/doc/mkd/ avant d'être recopié dans /usr/share/doc/mkd/. à l'installation du paquet.

```
./mkd-0~131215/debian/mkd/usr/share/doc/mkd/HISTORY
./mkd-0~131215/debian/mkd/usr/share/doc/mkd/NEWS.gz
./mkd-0~131215/debian/mkd/usr/share/doc/mkd/INSTALL
./mkd-0~131215/debian/mkd/usr/share/doc/mkd/AUTHORS
./mkd-0~131215/debian/mkd/usr/share/doc/mkd/changelog.gz
./mkd-0~131215/debian/mkd/usr/share/doc/mkd/COPYING
./mkd-0~131215/debian/mkd/usr/share/doc/mkd/copyright
./mkd-0~131215/debian/mkd/usr/share/doc/mkd/README
```

Notez que :

- ces fichier sont dans le répertoire *origine*
- ces fichiers ne sont pas tous indispensables. Les fichiers intéressant sont le plus souvent

    NEWS qui sont les nouveautés par rapport aux versions précédentes
    README qui sont les commentaires qui concernent ce paquet.

Vous pouvez ajouter :

- AUTHORS avec le droit de copie spécial à chaque contributeur
- COPYING même chose que Copyright, mais peut spécifier des particularités notamment si une partie du logiciel contient une autre source.
- HISTORY décrit l'historique et les péripéties du logiciel si il y a lieu
- INSTALL est généralement créé par un 'autotool'. Dans ce cas in est inutile de l'intégrer dans les docs, Dans notre cas on peut y préciser une autre manière d'installer le logiciel sans le paquet (qui ne sera donc pas automatiquement mis à jour)

On peut ajouter d'autres fichiers tels que :

- SURVEY souvent pour obtenir un soutien financier, mais ce peut être aussi un appel à contribution technique
- TRANSLATIONS demande d'aide pour traduire les messages et les manuels.
- ...etc.

debian/postinst

Ce fichier sera exécuté après l'installation du paquet et doit mettre les bases de données à jour.

```
#!/bin/sh -e
# file postinst for mkd debian installation
#

#
# POST INSTALL
#
echo " postinst : update icon cache "
        gtk-update-icon-cache -t /usr/share/icons/hicolor
echo " postinst : install and update MIME database "
        update-mime-database /usr/share/mime
        # OPTIONS :
        # update catman only if selected manuals exist in manpath :
        # if [ -e $(MAN)"/man1/mkd.1.gz" ]; then catman -M $(MAN) 1
 3; fi ;
        # if [ -e $(MAN)"/de.UTF-8/man1/mkd.1.gz" ]; then catman -M
 $(MAN)/de.UTF-8 1; fi ;
        # if [ -e $(MAN)"/de/man1/mkd.1.gz" ]; then catman -M $(MAN
)/de 1 ; fi ;
        # if [ -e $(MAN)"/fr.UTF-8/man1/mkd.1.gz" ]; then catman -M
 $(MAN)/fr.UTF-8 1; fi ;
        # if [ -e $(MAN)"/fr/man1/mkd.1.gz" ]; then catman -M $(MAN
)/fr 1 ; fi ;
        #
        # END POST INSTALL
        #
```

Notez que :

- Si le manuel man1 du logiciel a été installé on peut aussi ajouter, en cas de nécessité, la mise à jour de la base de données 'whatis'. C'est en principe automatique ....

debian/postrm

Ce fichier sera exécuté après la désinstallation de l'application pour mettre les bases de données à jour.

```
#!/bin/sh -e
# file postrm for mkd debian installation
#
# POST RM
#
echo " postrm : update icon cache "
        gtk-update-icon-cache -t /usr/share/icons/hicolor
echo " postrm : update MIME database "
        update-mime-database /usr/share/mime
```

```
#
# END POST RM
#
```

Makefile d'empaquetage

- Le fichier le plus important du répertoire *origine* est le Makefile

```
# file: Makefile pour version LINUX UNBUNTU 13.10 reprise de la ver
sion de janvier 2012 par Clara

 ###################################################
 ### UBUNTU version for mkd_R_13.12 debian package ###
 ###################################################

PGM     = mkd
SRC     = src
BINDIR  = usr/bin
BINMODE = 755

SRCS    = $(SRC)/mkd.c $(SRC)/asm.c $(SRC)/basic.c $(SRC)/cpp.c $(S
RC)/fortran.c $(SRC)/pascal.c $(SRC)/shell.c $(SRC)/tri.c
HDRS    = $(SRC)/version.h $(SRC)/mkd.h $(SRC)/asm.h $(SRC)/basic.h
 $(SRC)/cpp.h $(SRC)/fortran.h $(SRC)/pascal.h $(SRC)/shell.c $(SRC
)/find.inc.c $(SRC)/find.inc.h $(SRC)/tri.h
OBJS    = mkd.o asm.o basic.o cpp.o fortran.o pascal.o shell.o tri.
o
LIBS    =
CFLAGS  =  -O
LDFLAGS =
SPLINTFLAG = -weak


$(PGM): $(SRCS) $(HDRS)
        -@echo "*********** update $(PGM) ***********"
        $(CC) $(CFLAGS) -o $(PGM) $(SRCS)
        @#$(CC) $(LDFLAGS) -o $(PGM) $(OBJS) $(LIBS)
        -@echo " -- update $(BINDIR)/$(PGM):"
        -@if [ -d $(BINDIR) ]; then \
                mv -f $(PGM) $(BINDIR)/$(PGM);\
                chmod $(BINMODE) $(BINDIR)/$(PGM);\
        else \
                echo "couldn't find $(BINDIR)";\
        fi

install:
        mkdir -p debian/mkd/usr/bin
        cp -rf usr debian/mkd/.

mrproper:
        rm -rf *~
        rm -f $(PGM) $(BINDIR)/$(PGM)
        rm -f $(OBJS)
```

L'ordre d'exécution du Makefile est défini lors de la création du paquet, notamment par le fichier 'rules" que nous verrons ci_après.

1. nettoyage mrpropper
2. création de l'exécutable mkd
3. transfert des données et du binaire dans le répertoire debian/mkd/usr

- Les lettres majuscules définissent les macros habituelles dans les Makefiles
- La première directive $(PGM): $(SRCS) $(HDRS) va recompiler le binaire 'mkd' en fonction des processeurs : i386, amd64, etc. puis déplacer cet exécutable du répertoire courant vers le répertoire usr/bin
- La seconde directive va copier toute la hiérarchie du répertoire courant usr vers le répertoire debian/mkd/
- La troisième directive sert au nettoyage avant la création du paquet.

Création du paquet (debuild)

```
#!/bin/bash
# File Buid-all-clara

echo " compilation "
cd mkd-0~131215
# clé de clara <emailclara@mailserver> sur le pc GE60 expire le 20
dec 2014 autorisée à déposer sur le ppa de JPL
debuild -S -sa -k49E6B949
cd ..

echo " put to distri server "
dput ppa:jean-paul-louyot/mkd mkd_0~131215-1_source.changes
echo " Attendre le résultat envoyé par courrier électronique ... "

echo " make local debian package "
cd mkd-0~131215
debuild -i -us -uc -b
cd ..
```

Notez que :

- ceci est la phase finale

  debuild -S -sa -k49E6B949 va créer les fichiers nécessaires à la recompilation des paquets sur le dépôt.
  dput ppa:jean-paul-louyot/mkd mkd_0~131215-1_source.changes. Il faut attendre quelque temps avant que le résultat nous parvienne en positif ou échec
  debuild -i -us -uc -b va créer un paquet local .deb

Échecs et mise à jour de paquets

Créer et administrer un dépôt sur launchpad[3]

Éviter les échecs

ATTENTION : Le dépôt va utiliser l' *archive d'origine* dans toute la suite des procédures de mises à jour. Si vous avez modifié le Makefile par exemple, il est nécessaire de tout reprendre ou de faire un *patch*, car l'*archive d'origine* du dépôt ne va pas contenir le même Makefile, donc pas la même **somme de contrôle**.
Il est aussi possible de conserver, dans une *archive native*, le *répertoire d'origine* avec son *sous répertoire debian*; cela permet de conserver une mémoire des mises à jour .

Il est à noter que

- les *archives d'origine de série* ne contiennent pas le répertoire debian
- le répertoire debian est contenu dans une archive séparée *.debian.tar.gz*

Exemple de noms d'archives

- Archive du *répertoire d'origine* : <répertoire>-<version>.tar.gz
- Archive d'origine native : <répertoire>_<version>.tar.gz
- Archive d'origine de série : <répertoire>_<version>.orig.tar.gz
- Archive d'origine de série : <répertoire>_<version>-<n>.orig.tar.gz où <n> est une suite successive de série 1, 2, 3, etc.

Pour faire les essais il est préférable de lancer la commande 'Rebuild-all' *(Clara pour le chiffrage du transfert)*

```
#!/bin/bash
# File Rebuid-all-clara

echo " rm old files "
# rm old files
rm -f *.dsc
rm -f *.changes
rm -f *.build
rm -f *.upload
rm -f *.deb
rm -f mkd-0~131215/debian/*.ex
rm -f mkd-0~131215/debian/*.EX

echo " sauvegarde du répertoire debian"
echo " note d'option -n :  les fichiers ne sont pas recopiés si ils
 existent"
# n'oubliez pas le ' .' après debian. choisir l' option -rf pour re
créer les fichiers
cp -rn mkd-0~131215/debian .

# attention à la commande qui suit ...
# cette archive native de référence ne doit plus être récréée lors
des mises à jour
# ---------------------------------
echo " re-créer le fichier natif  .tar.gz pour tenir compte des mod
ifications "
tar cvzf mkd-0~131215.tar.gz mkd-0~131215
#
# pour une mise à jour :
# ---------------------------------
# - attention au caractère souligné 'mkd_0'
# - mv mkd-0~131015.tar.gz mkd est l'original natif
# - mkd_0~131015.orig.tar.gz ne doit plus être changé
# echo " renommer le fichier original natif .tar.gz en .orig.tar.gz
 et ne plus le modifier "
# mv mkd-0~131015.tar.gz mkd_0~131015.orig.tar.gz
#
echo " dh_make va créer le répertoire debian avec de nouveaux fichi
ers si il a été supprimé "
echo " il ne sera pas recréé avec de nouveaux fichiers si il existe
 "
cd mkd-0~131215
dh_make -c GPL -e edeulo@free.fr -s -n -f mkd-0~131215.tar.gz -r CD
```

```
BS
# cd..

echo " re-debuild -S -sa -k49E6B949 (Clara) "
# cd mkd-0~131215
debuild -S -sa -k49E6B949
# cd ..

echo " re-debuild -i -us -uc -b création du paquet local "
# cd mkd-0~131215
debuild -i -us -uc -b
cd ..

echo " Si tout s'est bien passé on peut envisager le dépôt "
echo " Entrer la commande 'Buid-all-clara' "
```

Pour éviter les échecs : Vérifier si tout se passe bien avant d'envoyer le fichier au dépôt.

Échecs

Le transfert au dépôt ne se passe pas toujours très bien, il est rejeté. Il est nécessaire de corriger la construction du paquet.

Les rejets sont souvent dus aux commandes incluses dans le fichier *Makefile*.
Évitez d'utiliser des commandes susceptibles de renvoyer une valeur différente de zéro.

1. Ecrivez *rm -f <fichier>* plutôt que *rm <fichier>*. *rm -f* renvoie zéro même si *<fichier>* n'existe pas.
2. Supprimez le binaire dans la dépendance **clean**, faute de quoi le binaire ne sera pas mis à jour pour un autre processeur bien qu'il n'y ait aucune erreur de syntaxe.
3. Vérifiez la validité des dépendances. N'écrivez pas *clean : <fichier>* ça n'a pas de sens.
4. Veillez à ce que le contenu du répertoire soit propre, identique après compilation :
    - De telle sorte que la somme de contrôle du répertoire soit la même après compilation; que tous les fichiers créés soient effacés.
    - Effacer tous les fichiers '.o' (rm -f *.o) par exemple, ainsi que le nouvel exécutable qui a déjà été copié dans *debian/mkd/usr/bin/*

Launchpad renvoie des informations à l'auteur du transfert (Erreurs communes, la raison de l'échec, etc.)

> Rejected: Unable to find distroseries: unstable
>
> > Dans debian/changelog il faut remplacer unstable par une version d'ubuntu : *precise*, ou version actuelle
>
> Rejected: Unable to find section: unknown
>
> > Dans debian/control il faut remplacer unknown par la 'Sous-section' concernée[1] (http://packages.ubuntu.com/saucy/) : *devel* pour nos applications

Si vous renvoyez le fichier tel quel avec la correction, il sera refusé si le fichier *.upload* se trouve

dans votre répertoire:

> <UPLOADE_FILE> already exists in <LOCATION>
> *Rejected because the logfile <package>_source on <host> .upload exist. Just remove the .upload file and re-run dput, or invoke dput with the flag -lf* (-l run lintian before the upload; -f force an upload of an already uploaded package)
> Il faut supprimer le fichier .upload ou renvoyer le paquet corrigé avec les option -lf
> *dput -lf [host [:arg]] package.changes*
> Remarque : On peut éviter l'écriture du fichier .upload avec -option -U : *dput -U*

Résumé en cas d'échec :

Si vous ne modifiez que les fichiers du répertoire debian dans une série

Vous pouvez renvoyer les fichiers reconstruits après avoir effacé le fichier *.upload*

→ Dans une série, l'archive du répertoire debian est séparée. Suffixe: *.debian.tar.gz*

Si vous modifiez un fichier du répertoire *origine*, contenant aussi le répertoire *debian*, il faut reconstruire le paquet !

1. recharger l'original
2. régénérer la source
3. **faire un *patch*** ou **reconstruire le paquet** avec un numéro de version supérieur à la version en échec
4. tester le transfert avec l'option dput -o, débugger avec l'option dput -d, simuler avec dput -s
5. transférer le paquet source avec les changements (source.changes) avec dput -lf
6. accessoirement, supprimer la version en échec sur launchpad. Cela peut arriver si on change de version Ubuntu dans le fichier changelog. (ATTENTION : La version n'apparaît plus, mais n'est jamais complètement effacée.)

Mise à jour de paquets

- Voir **Mise à jour de paquets** [4]

Cas traités

1 L'intention d'empaqueter (création de paquets multiples)
2 Patch (rustine)
3 Mettre à jour (création de paquets binaires simples)
4 Nouvelle version (Paquets natifs)
5 Dépôts sur Launchpad

Notes et références

- Notes :

  Discussions[2] (http://edeulo.free.fr/phpBB3/viewforum.php?f=18) autour de l'empagetage debian et rpm (*mot de passe antispam :* mkd)

- Références :

1. https://launchpad.net/~jean-paul-louyot/+archive/mkd
2. http://www.debian.org/doc/manuals/maint-guide/dreq.fr.html
3. Créer et administrer un dépôt sur launchpad (http://doc.ubuntu-fr.org/tutoriel/creer_et_administrer_un_ppa_sur_launchpad)
4. https://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)/Dépôt_parrainé_du_paquet_"original"_mkd-131215/Mise_à_jour_de_paquets

Make documents

Mise à jour de paquets Debian-Ubuntu

Vous envisagez d'empaqueter ou de mettre à jour une de vos applications.

Vous désirez personnaliser un paquet existant pour :

- votre usage personnel,
- proposer votre version sur votre dépôt personnel (ppa:)

La suite des solutions aux exercices d'empaquetage devrait vous y aider.

À petit pas, avec les erreurs communes et leurs solutions.

Maintenance : En travaux

Page en cours de relecture

Votre participation est souhaitée pour faire évoluer les projets mkd; vous pouvez

- modifier les sources de mkd et ses dérivés (mkdw, mkdcppw, etc.) avec les restrictions d'usage : Consultez le copyright de chaque application ; citez l'origine de l'application et le nom des auteurs.
- participer aux traductions.
- Prérequis[1] : Avoir installé les paquets suivants : debhelper cdbs lintian build-essential fakeroot devscripts pbuilder dh-make debootstrap.

Sigles usuels que vous pouvez rencontrer

Paquets en souffrance et paquets souhaités[3] (http://www.debian.org/devel/wnpp/) Bogues, ITP:intention d'empaqueter, RFA:paquets disponibles à l'adoption, O:orphelins, RFH:demande d'aide, RFP:demande à empaqueter

L'intention d'empaqueter

Bien entendu, nous n'allons pas déposer une requête à Debian.
Nous nous contenterons des dépôts sur les serveurs Launchpad et des sources du projet mkd. (http://edeulo.free.fr/wiki/index.php/Projet_mkd)

Dans l'intention d'empaqueter on suppose, à priori que l'application n'a jamais été empaquetée, elle s'installe très bien avec une procédure par un fichier de commandes 'install.sh' ou avec 'make' dans un répertoire d'installation contenant le fichier 'Makefile'

Exemple d'un Makefile original

```
#!/bin/make
# file: Makefile for Ubuntu / Debian
# updated by JPL 2012-05-22
#
# Attention: avant de compiler, il faut valider les options de compilation
# dans version.h (notamment version UNIX ou version PC, ainsi que la langue)
#
PGM      = mkd
BINDIR   = /usr/bin
BINMODE  = 755
MAN_FR   = mkd_fr.1
MAN_EN   = mkd_en.1
CATMANDIR    = /var/cache/man/cat1                              # ubuntu / debian
CATMANDIR_FR = /var/cache/man/fr.UTF-8/cat1 debian

MANDIR       = /usr/share/man/man1
MANDIR_FR    = /usr/share/man/fr.UTF-8/man1
SRCS     = mkd.c asm.c cpp.c
HDRS     = version.h asm.h cpp.h find.inc.h internationalisation.h find.inc.c \
 basic.inc.c fortran.inc.c pascal.inc.c shell.inc.c tri.inc.c  syn-fr.i syn-eng.i
OBJS     =
LIBS     =
CFLAGS   =  -O
LDFLAGS  =
SPLINTFLAG = -weak

install: $(PGM) $(MAN_EN) $(MAN_FR)
          # install PGM in BINDIR
          -@if [ -d $(BINDIR) ]; then \
                    cp -f $(PGM) $(BINDIR)/$(PGM);\
                    chmod $(BINMODE) $(BINDIR)/$(PGM);\
                    if [ $(BINDIR) != "/bin" ]; then \
                              ln -s $(BINDIR)/$(PGM) /bin/$(PGM);\
                    fi \
          else \
                    echo "couldn't find  $(BINDIR)";\
          fi
          #
          ./install_manuals
          #
          ./install_languages
          #
          ./install_icons
          #
          ./install_mime_database
          # End install

$(PGM): $(SRCS) $(HDRS)
          $(CC) $(CFLAGS) -o $(PGM) $(SRCS)
          @#$(CC) $(LDFLAGS) -o $(PGM) $(OBJS) $(LIBS)
          -@strip $(PGM)                                  # clean ASM and LINK reloc.
```

```
$(MAN_EN).gz: manuals
          -$(PGM) -Ctw M manuals $(MAN_EN)
          -@gzip $(MAN_EN)                              # make en manual $(MAN_EN).g
z

$(MAN_FR).gz: manuals
          -$(PGM) -Ctw F manuals $(MAN_FR)
          -@gzip $(MAN_FR)                              # make en manual $(MAN_FR).g
z

clean:
          #rm -f $(PGM) $(OBJS) $(MAN)
```

Pour empaqueter il est nécessaire de disposer d'un fichier 'Makefile' contenant une directive 'clean' et d'une directive d'installation de l'application dans les répertoires usuels (Compilation, manuels, traductions, etc.)
L'empaquetage se fera dans le répertoire debian. Le binaire doit se trouver dans *debian/<app>/usr/bin/*, et pas dans */usr/bin/*. Pour que ce soit possible il est nécessaire d'introduire une macro $(DESTDIR).

Modifier le Makefile pour l'empaquetage

Il est nécessaire

1. de déplacer en tête de fichier la dépendance de création du binaire $(PGM)
2. vérifier que la dépendance 'install' dépende de la création du binaire; "$(PGM)" dans ce Makefile
3. d'effacer le fichier binaire $(PGM) dans la dépendance inconditionnelle 'clean'. Elle sera la première exécutée à la création du paquet
4. d'inclure les commandes séparées du Makefile précédent *./install_**
5. d'ajouter la macro $(DESTDIR)
6. de vérifier qu'aucune commande risque de retourner inutilement une valeur différente de zéro par exemple en cas d'absence de fichier à effacer.
7. nettoyer le répertoire à la fin de la compilation.

```
#!/bin/make
# File: Makefile for packaging mkd on debian or ubuntu systems
#########################################################
# Updated by JPL 2012-05-22
# Amended by JPL 2012-06-08
# Amended by Clara for Wikibooks 2014-01-12
#
# Copyright: ©EELL http://edeulo.free.fr/wiki/index.php/Projet_mkd#LE_DROIT_DE_CO
PIE:
# For tests type the command "make -d install DESTDIR=../tests" for tests
#########################################################
#
BIN    = $(DESTDIR)/usr/bin
CATMANDIR = $(DESTDIR)/var/cache/man
ETC    = $(DESTDIR)/etc
DOC    = $(DESTDIR)/usr/share/doc
ICONS = $(DESTDIR)/usr/share/icons/hicolor
```

```
LANG  = $(DESTDIR)/usr/share/locale
MAN   = $(DESTDIR)/usr/share/man
MIME  = $(DESTDIR)/usr/share/mime
PGM       = mkd
SRCS    = mkd.c asm.c cpp.c
HDRS    = version.h asm.h cpp.h find.inc.h internationalisation.h \
 find.inc.c basic.inc.c fortran.inc.c pascal.inc.c shell.inc.c tri.inc.c syn-fr.i
 syn-eng.i
OBJS    =
LIBS    =
CFLAGS  =  -O
LDFLAGS =
SPLINTFLAG = -weak

# create bin files: see clean : rm -f $(PGM)
# force recompilation $(SRCS) $(HDRS)
$(PGM):
          gcc -c $(CFLAGS) $(SRCS) $(LIBS)
          gcc -o $(LDFLAGS) $(PGM) $(OBJS) $(LIBS)
          chmod 755 $(PGM)

install: $(PGM)
          # create $(DESTDIR), sub-directories, and copy executable in bin direct
ory:
          install -d $(BIN) $(CATMANDIR) $(ETC) $(DOC) $(ICONS) $(LANG) $(MAN) $(
MIME)
          install $(PGM) $(BIN)/.
          #
          # install docs: see file ./install_docs
          # install -d $(DOC)/$(PGM)/html
          # install -m644 doc/*.html $(DOC)/$(PGM)/html/.
          #
          # install icons and update cache: see update in post-install package an
d see ./install_icons.
          install -d $(ICONS)/256x256/apps $(ICONS)/48x48/apps $(ICONS)/32x32/app
s $(ICONS)/16x16/apps
          chmod 644 icons/*
          cp -f icons/mkd-256x256.png $(ICONS)/256x256/apps/mkd.png
          cp -f icons/mkd-48x48.png $(ICONS)/48x48/apps/mkd.png
          cp -f icons/mkd-32x32.png $(ICONS)/32x32/apps/mkd.png
          cp -f icons/mkd-16x16.png $(ICONS)/16x16/apps/mkd.png
          if [ -e "/usr/share/icons/hicolor/16x16/apps/mkd.png" ]; \
                    then gtk-update-icon-cache -t /usr/share/icons/hicolor; fi
          #
          # install languages: see file ./install_languages.
          # Installs for tests the local languages, 'de', 'es', 'fr', it, ro, etc
.:
          install -d $(LANG)/en/LC_MESSAGES $(LANG)/de/LC_MESSAGES $(LANG)/fr/LC_
MESSAGES
          chmod 644 lang/*
          cp -f lang/mkd_en.mo $(LANG)/en/LC_MESSAGES/mkd.mo
          cp -f lang/mkd_fr.mo $(LANG)/fr/LC_MESSAGES/mkd.mo
          #
```

```
	# install manuals: see file ./install_manuals
	## install default manual:
	install -D -m644 lang/mkd_en.1.gz $(MAN)/man1/mkd.1.gz
	#
	## install 'fr' manual:
	if [ -d "/usr/share/man/fr.UTF-8/man1" ]; \
		then \
			install -d $(MAN)/fr.UTF-8/man1;\
			cp -f  lang/mkd_fr.1.gz $(MAN)/fr.UTF-8/man1/mkd.1.g
z; \
		elif [ -d "/usr/share/man/fr/" ]; \
			then \
				install -d $(MAN)/fr/man1; \
				cp -f lang/mkd_fr.1.gz $(MAN)/fr/man1/mkd.
1.gz ; \
	fi
	#
	# install catman only if selected manual exist: see file ./install_catm
an.
	if [ -e $(MAN)"/man1" ]; then catman -M $(MAN)/man1/mkd.1.gz; fi
	if [ -e $(MAN)"/fr.UTF-8/man1/mkd.1.gz" ]; then catman -M $(MAN)/fr.UTF
-8/man1/mkd.1.gz; fi
	if [ -e $(MAN)"/fr/man1/mkd.1.gz" ]; then catman -M $(MAN)/fr/man1/mkd.
1.gz ; fi
	#
	# install and update MIME database: see update in post-install packagea
nd see file ./install_mime_database
	install -D -m644  mkdw.xml $(MIME)/packages/mkdw.xml
	if [-e "/usr/share/mime/packages/mkdw.xml" ]; then update-mime-database
 /usr/share/mime/; fi
	# clean #
	rm -f $(PGM) $(OBJS) $(MAN) *.o

# create man only in programmers source file; not for packaging
# type command "make -d mkd.1" to create updated file "manuals"
mkd.1: manuals
	if [ -e "/usr/bin/mkd" ]; \
	then \
		mkd -Ct F manuals mkd.1; \
		gzip -cf mkd.1 > lang/mkd_fr.1.gz; \
		mkd -Ctw M manuals mkd.1; \
		gzip -cf mkd.1 > lang/mkd_en.1.gz; \
		rm -f mkd.1; \
	fi

clean:
	rm -f $(PGM) $(OBJS) $(MAN) *.o *.~
```

Tester la compilation

```
make -d install DESTDIR=../tests
```

Contenu du répertoire des tests. S'assurer que

1. le binaire *mkd* a bien été créé et copié dans le répertoire *usr/bin/*
2. tous les fichiers ont été copiés dans les répertoires usuels

Listing du répertoire "tests"

```
edeulo@jpl-ubuntu-13:~/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice-0$ find tests
tests
tests/usr
tests/usr/share
tests/usr/share/mime
tests/usr/share/mime/packages
tests/usr/share/mime/packages/mkdw.xml
tests/usr/share/doc
tests/usr/share/doc/mkd
tests/usr/share/doc/mkd/html
tests/usr/share/man
tests/usr/share/man/fr.UTF-8
tests/usr/share/man/fr.UTF-8/man1
tests/usr/share/man/fr.UTF-8/man1/mkd.1.gz
tests/usr/share/man/man1
tests/usr/share/man/man1/mkd.1.gz
tests/usr/share/icons
tests/usr/share/icons/hicolor
tests/usr/share/icons/hicolor/256x256
tests/usr/share/icons/hicolor/256x256/apps
tests/usr/share/icons/hicolor/256x256/apps/mkd.png
tests/usr/share/icons/hicolor/16x16
tests/usr/share/icons/hicolor/16x16/apps
tests/usr/share/icons/hicolor/16x16/apps/mkd.png
tests/usr/share/icons/hicolor/32x32
tests/usr/share/icons/hicolor/32x32/apps
tests/usr/share/icons/hicolor/32x32/apps/mkd.png
tests/usr/share/icons/hicolor/48x48
tests/usr/share/icons/hicolor/48x48/apps
tests/usr/share/icons/hicolor/48x48/apps/mkd.png
tests/usr/share/locale
tests/usr/share/locale/fr
tests/usr/share/locale/fr/LC_MESSAGES
tests/usr/share/locale/fr/LC_MESSAGES/mkd.mo
tests/usr/share/locale/en
tests/usr/share/locale/en/LC_MESSAGES
tests/usr/share/locale/en/LC_MESSAGES/mkd.mo
tests/usr/share/locale/de
tests/usr/share/locale/de/LC_MESSAGES
tests/usr/bin
tests/usr/bin/mkd
tests/var
```

vérifier que le contenu du répertoire d'origine n'a pas changé

1. vérifier les dates, éventuellement mettre les dates à jour avant de lancer le test. commande *touch -a*
2. s'assurer que le binaire a été effacé

Créer la configuration d'empaquetage

- Cela se traduit par la création d'un répertoire *debian* dans le répertoire racine.
- Ce répertoire debian contiendra tous le fichiers nécessaires à l'empaquetage.

```
dh_make --createorig
```

Cette commande va créer le répertoire debian et le fichier .orig.tar.xz au dessus du répertoire racine et se traduit par des questions :

1. Type of package: single binary, indep binary, multiple binary, library, kernel module, kernel patch?

```
[s/i/m/l/k/n] On peut choisir s, i ou m
```

l'option '-s' ne devrait être utilisée que pour produire un binaire mais on peut créer un paquet binaire plus complet avec le manuel et de la documentation.
l'option '-m' produit deux paquets : un paquet binaire et un paquet *-doc* commun à tous les paquets binaires générés.

```
edeulo@jpl-ubuntu-13:~/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice-0/mkd-0~12030
2$ dh_make --createorig

Type of package: single binary, indep binary, multiple binary, library, kernel mo
dule, kernel patch?
 [s/i/m/l/k/n] m

Maintainer name  : Edeulo
Email-Address    : edeulo@unknown
Date             : Sun, 12 Jan 2014 18:20:02 +0100
Package Name     : mkd
Version          : 0~120302
License          : blank
Type of Package  : Multi-Binary
Hit <enter> to confirm:
Done. Please edit the files in the debian/ subdirectory now. You should also
check that the mkd Makefiles install into $DESTDIR and not in / .
```

Fichiers créés dans le répertoire debian

```
edeulo@jpl-ubuntu-13:~/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice-0/mkd-0~12030
2/debian$ ls
changelog  docs             manpage.xml.ex  mkd.doc-base.EX  postrm.ex      READM
E.source
compat     init.d.ex        menu.ex         mkd-doc.docs     preinst.ex     rules
control    manpage.1.ex     mkd.cron.d.ex   mkd-doc.install  prerm.ex       sourc
e
copyright  manpage.sgml.ex  mkd.default.ex  postinst.ex      README.Debian  watch
.ex
```

Pour notre exercice on peut, à priori,

1. supprimer tous les fichier se terminant par '.ex', sauf postinst et postrm qui nous serviront à mettre les bases de données à jour après installation et désinstallation.
2. effacer les fichiers DEBIAN.* et le fichier mkd.doc-base.EX

Il est nécessaire de modifier

1. **changelog** : remplacer 'unstable' par votre version d'Ubuntu[2], 'saucy' pour ubuntu-13.10, en minuscules.
2. **control**[3] :

   Section il est impératif de remplacer unknown par devel (sous-section de Développement)[4] ou autre sous-section ...
   Maintainer compléter l'adresse électronique
   Homepage compléter l'adresse des mises à jour
   Description Les paquets mkd et mkd-doc doivent être présentés dans le paragraphe d'affichage dans l'installeur de la *Logithèque Ubuntu*
   Ajouter *Recommends:* et "Suggests:" pour, à l'installation, faire référence à l'un et à l'autre paquet.

   fichier *control* d'essais :

```
Source: mkd
Section: devel
Priority: extra
Maintainer: Edeulo <edeulo@free.fr>
Build-Depends: debhelper (>= 8.0.0)
Standards-Version: 3.9.4
Homepage: <insert the upstream URL, if relevant>
#Vcs-Git: git://git.debian.org/collab-maint/mkd.git
#Vcs-Browser: http://git.debian.org/?p=collab-maint/mkd.git;a=summary

Package: mkd
Architecture: any
Depends: ${shlibs:Depends}, ${misc:Depends}
Description: Exercice "Intention d'empaqueter"
 Voir "mkd (extracteur de documents)" sur Wikibooks
```

```
 Ce paquet debian Ubuntu est destiné à installer le binaire
Suggests: mkd-doc

Package: mkd-doc
Architecture: all
Description: Exercice "Intention d'empaqueter"
 Voir "mkd (extracteur de documents)" sur Wikibooks
 Ce paquet debian Ubuntu est destiné à présenter la
 documentation commune à tous les paquets binaires
Recommends: mkd
```

Autres fichiers à mettre à jour

1. **docs** : Contient les fichiers documentaires du répêrtoire d'origine : README, INSTALL, UNINSTALL, COPYRIGHT, AUTHORS etc. Si ce fichier est vide c'est que ces fichiers n'existaient pas à la création du répertoire debian. ces fichier se trouveront dans /usr/
2. **copyright** : Éditer et compléter ...
3. **postinst** : Inclure les commandes tels que les mises à jour non exécutable à la fabrication du paquet.
4. **postrm** : Idem !

Ajouter les quatre lignes de mise à jour des bases de données.

```
#!/bin/sh
# postrm script for mkd
#
# see: dh_installdeb(1)

set -e

# summary of how this script can be called:
#        * <postrm> `remove'
#        * <postrm> `purge'
#        * <old-postrm> `upgrade' <new-version>
#        * <new-postrm> `failed-upgrade' <old-version>
#        * <new-postrm> `abort-install'
#        * <new-postrm> `abort-install' <old-version>
#        * <new-postrm> `abort-upgrade' <old-version>
#        * <disappearer's-postrm> `disappear' <overwriter>
#          <overwriter-version>
# for details, see http://www.debian.org/doc/debian-policy/ or
# the debian-policy package

case "$1" in
    purge|remove|upgrade|failed-upgrade|abort-install|abort-upgrade|disappear)
          echo " postrm : update icon cache "
          gtk-update-icon-cache -t /usr/share/icons/hicolor
          echo " postrm : update MIME database "
          update-mime-database /usr/share/mime
    ;;
```

```
    *)
        echo "postrm called with unknown argument \`$1'" >&2
        exit 1
    ;;
esac

# dh_installdeb will replace this with shell code automatically
# generated by other debhelper scripts.
#DEBHELPER#
exit 0
```

Empaqueter

La commande suivante va lancer la construction du paquet

```
debuild -i -us -uc -b
```

En cas d'erreurs la construction sera annulée et la lecture du fichier mkd_0~120302-1_<processeur>.build vous permettra de savoir où et pourquoi la compilation s'est arrêtée.

Dans notre cas, alors que la compilation s'était bien déroulée dans les tests nous obtenons une erreur :

```
gcc -o mkd
gcc: fatal error: no input files
compilation terminated.
make[1]: *** [mkd] Erreur 4
make[1]: quittant le répertoire « /home/edeulo/Packaging/mkd/mkd_12XXXX/mkd-12030
2-exercice-0/mkd-0~120302 »
dh_auto_build: make -j1 returned exit code 2
make: *** [build] Erreur 2
dpkg-buildpackage: erreur: debian/rules build a produit une erreur de sortie de t
ype 2
```

Correction de la ligne OBJS

```
OBJS     = mkd.o asm.o cpp.o
```

À la relance de la construction tout se passe bien et nous obtenons

edeulo@jpl-ubuntu-13:~/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice-0$ ls -al
total 252
drwxr-xr-x 4 edeulo edeulo 4096 janv. 13 12:04 .
drwx------ 6 edeulo edeulo 4096 janv. 12 18:00 ..
-rwx--x--x 1 edeulo edeulo 375 mars 12 2013 **KONSOLE**
drwx------ 6 edeulo edeulo 4096 janv. 13 12:04 **mkd-0~120302**
-rw-r--r-- 1 edeulo edeulo 14058 janv. 13 12:04 **mkd_0~120302-1_amd64.build**
-rw-r--r-- 1 edeulo edeulo 1004 janv. 13 12:04 **mkd_0~120302-1_amd64.changes**
-rw-r--r-- 1 edeulo edeulo 2748 janv. 13 12:04 **mkd_0~120302-1_amd64.deb**
-rw-r--r-- 1 edeulo edeulo 205340 janv. 12 18:20 **mkd_0~120302.orig.tar.xz**
-rw-r--r-- 1 edeulo edeulo 2124 janv. 13 12:04 **mkd-doc_0~120302-1_all.deb**
drwxr-xr-x 5 edeulo edeulo 4096 janv. 12 16:35 **tests**

Notes

1. Le fichier mkd_0~120302.orig.tar.xz a pour date : janv. 12 18:20
2. En dehors du fichier Makefile rien n'a changé dans le répertoire d'origine. Les fichiers objets *.o ont bien été effacés.

On peut maintenant mettre à jour l'archive d'origine *.tar.xz*

1. Déplacer le répertoire *debian*. N'oubliez pas le point '.'
   mv -v mkd-0~120302/debian .
2. Créer le fichier d'origine *.tar.xz*
   tar cvJf mkd-0~120302.tar.xz mkd-0~120302
3. Vérifier que la taille du fichier d'origine est à peine plus grande que le fichier orig.tar.xz
   -rw-r--r-- 1 edeulo edeulo 205340 janv. 12 18:20 mkd_0~120302.orig.tar.xz
   -rw-r--r-- 1 edeulo edeulo 205384 janv. 13 15:37 mkd-0~120302.tar.xz
4. Replacer le répertoire *debian* précédemment déplacé
   mv -v debian mkd-0~120302/.
5. Effacer le fichier *mkd_0~120302.orig.tar.xz*
   rm mkd_0~120302.orig.tar.xz
6. Recréer le paquet avec le nouveau fichier d'origine
   Dans le répertoire d'origine : **debuild -i -us -uc -b**

Vérifier le contenu des paquets .deb avec le gestionnaire d'archives ou dans le répertoire debian

Notes :

1. Ces archives sont quasiment vides, il n'y a aucun binaire dans le paquet binaire, on le vérifie aussi en consultant le répertoire debian/mkd
2. Le répertoire debian contient un répertoire tmp qui, lui contient toute l'archive.

Conclusion *dh_install* n'a pas fonctionné pour le paquet binaire.

Le manuel de dh_install nous apprend que chaque paquet est construit par un fichier *debian/<paquet>.install*. Or, si le répertoire debian contient bien un fichier de construction *mkd-doc.install*, **il manque le fichier de construction binaire *mkd.install* que nous allons créer**. *C'est hélas courant avec ces paquets multiples*.

1. On copie le listing du fichier des tests que l'on colle dans un fichier *debian/mkd.install*
2. On supprime toutes les lignes qui ne contiennent que des répertoires.
3. On supprime le préfixe *tests/* de toutes les lignes qui restent.
4. À chaque ligne on ajoute le répertoire où le fichier doit être copié.

Fichier ***mkd.install*** :
Fichier à copier du répertoire *debian/tmp* vers le répertoire *debian/mkd*

```
usr/bin/mkd usr/bin
usr/share/mime/packages/mkdw.xml usr/share/mime/packages
usr/share/man/fr.UTF-8/man1/mkd.1.gz usr/share/man/fr.UTF-8/man1
usr/share/man/man1/mkd.1.gz usr/share/man/man1
usr/share/icons/hicolor/256x256/apps/mkd.png usr/share/icons/hicolor/256x256/apps
usr/share/icons/hicolor/16x16/apps/mkd.png usr/share/icons/hicolor/16x16/apps
usr/share/icons/hicolor/32x32/apps/mkd.png usr/share/icons/hicolor/32x32/apps
usr/share/icons/hicolor/48x48/apps/mkd.png usr/share/icons/hicolor/48x48/apps
usr/share/locale/fr/LC_MESSAGES/mkd.mo usr/share/locale/fr/LC_MESSAGES
usr/share/locale/en/LC_MESSAGES/mkd.mo usr/share/locale/en/LC_MESSAGES
```

On relance **debuild -i -us -uc -b**

Cette fois le paquet binaire est complet.

Pour information

- Éditer le fichier *.changes*[4] (http://www.debian.org/doc/debian-policy/ch-controlfields.html#s-f-Checksums). Notez le contenu. Le champ *Date* est la date de la mise à jour extraite du fichier changelog alors que le paquet a été construit le jour suivant.
- Éditer le fichier *.build* et vérifier qu'il n'y a pas d'erreur ou d'incompréhension. Exemple, ligne 204 : *W: mkd: bad-homepage <insert the upstream URL, if relevant>*

Actions complémentaires

- Modifier ou compléter les fichiers *changelog*, *control*, *copyright*, s'il y a lieu.
- Si tout est correct, le paquet est prêt à être installé sur la machine locale.

Remarques

- Nous n'avons rien touché au répertoire d'origine
- À l'installation du paquet, avec la *Logithèque Ubuntu* ou *GDebi*, le copyright, les suggestions de paquets et les paquets recommandés, n'apparaissent pas.

Préparer le paquet pour un dépôt

Rappel, consultez le chapitre *Dépôt sur Launchpad*
Pour déposer un paquet il faut disposer d'un compte de dépôt (PPA), et d'une clé PGP

Créer les conditions de transfert sécurisé

Si l'option -sn de *debuild* ne fonctionne pas il faut utiliser l'option -sa et simuler une nouvelle version .orig.tar.xz

1. renommer le répertoire mkd-0~120302.tar.xz mkd_0~120302.orig.tar.xz
2. entrer dans le répertoire d'origine mkd-0~120302

3. sécuriser le transfert avec la clé de Edeulo 230B1A04
   debuild -S -sa -k230B1A04
4. sortir du répertoire d'origine.

   Liste des nouveaux fichiers

   -rw-r--r-- 1 edeulo edeulo 3264 janv. 15 14:53 mkd_0~120302-1.debian.tar.gz
   -rw-r--r-- 1 edeulo edeulo 1486 janv. 15 14:53 mkd_0~120302-1.dsc
   -rw-r--r-- 1 edeulo edeulo 1407 janv. 15 14:53 mkd_0~120302-1_source.build
   -rw-r--r-- 1 edeulo edeulo 1723 janv. 15 14:53 mkd_0~120302-1_source.changes
   -rw-r--r-- 1 edeulo edeulo 205384 janv. 13 15:37 mkd_0~120302.orig.tar.xz
   L'archive mkd_0~120302-1.debian.tar.gz contient l'essentiel du répertoire debian.
   L'archive mkd_0~120302.orig.tar.xz contient le répertoire de référence d'origine.
   Les fichiers .dsc et .changes sont sécurisés pour le transfert au dépôt.
   Le fichier .build liste le détail de l'opération de sécurisation.

5. Vérifier le paquet

   dput -ol ppa:<nom-du-ppa>/mkd mkd_0~120302-1_source.changes

6. Simuler le dépôt

   dput -s ppa:<nom-du-ppa>/mkd mkd_0~120302-1_source.changes

7. **Tenter le dépôt du paquet sur Launchpad**; il peut encore y avoir un rejet.

Patch (rustine)

La rustine informatique est destinée à corriger les bogues apparus au cours de l'utilisation du logiciel, ou pour ajouter des fonctionnalités particulières.

Dans notre cas, à titre expérimental, on peut appliquer des rustine à des fichiers acceptés par le dépôt et qui ont fait apparaître des erreurs de compilation.

> ☞ Dans ces deux cas la somme de contrôle de la nouvelle version ne correspondra pas à celle qui a été déposée. La version doit être incrémentée.

- *Appliquer une rustine est nécessaire pour le 'suivi' d'une application*
- *Le travail nécessaire pour appliquer une rustine informatique est assez lourd. Il est souvent plus facile de créer une nouvelle version avec les défauts corrigés.*

  Il se pose alors la question du suivi de l'empaquetage.

Exemple

1. L'ordinateur du dépôt va compiler l'application pour plusieurs types de processeurs avec la rigueur nécessaire et va générer des erreurs. **Votre paquet est refusé.**
2. L'application ne fait apparaître aucune erreur localement et le paquet que vous avez produit fonctionne très bien.
3. L'erreur ou le type d'erreur est communiqué par courrier électronique.
4. Ce type d'erreur vient souvent du Makefile

Recréer l'environnement de travail

1. Créer le répertoire de travail 'natif' dans l'exemple.
2. Dans le répertoire natif : télécharger le fichier source à corriger, avec le navigateur, ou avec la commande wget
3. Afin de faciliter l'accès à la console dans le répertoire natif, créer la commande **KONSOLE** qu'il faut rendre exécutable.

```
#!/bin/bash
echo -e '\E['32';'01'm'"click on New Line to 'start' the Konsole"
# echo -e '\E['31';'01'm' "not 'start in a terminal'"
echo -e '\E['32';'01'm'"click on Ctrl-C to exit"
tput sgr0 # Reset text attributes to normal without clear
read pwd
pwd
echo $pwd
tput sgr0 # Reset text attributes to normal without clear
/usr/bin/konsole background-mode --workdir pwd dir
read
```

☞ Dans la console, utilisez les touches CTRL+SHIFT+c Pour copier, CTRL+SHIFT+v pour coller.

Télécharger ;

https://launchpad.net/~jean-paul-louyot/+archive/exercises/+packages
→ mkd_0~120302-1.debian.tar.gz (3.2 KiB)
→ mkd_0~120302.orig.tar.xz (200.6 KiB)

Copier les fichiers dans un répertoire mkd-120302-exercice-1

décompressez l'archive .orig.tar.gz → mkd-120302-exercice-1/mkd-0~120302 →
décompressez l'archive .debian.tar.gz dans mkd-0~120302 → mkd-0~120302/debian

Réserver les fichiers d'origine dans un répertoire des fichiers originaux.
Éditer le fichier mkd-0~120302/Makefile et corriger

Créer le patch

Voir le manuel des commandes unix *diff*, *patch*, *quilt* (ou dquilt)

Créer un patch consiste à faire la différence une *diff* entre un ancien et un nouveau fichier, puis de faire un *patch* (coller une rustine) afin de ne corriger que la partie modifiée pour mettre à jour ou pour être applicable dans d'autres conditions d'utilisation. Ceci évite de recharger tous les fichiers et de figer une application. On garde ainsi une souplesse d'utilisation.

Un patch pourrait par exemple, ne mettre à jour que des langues déjà installées.

Environnement de travail pour quilt

pour préciser où se trouvent les correctifs

ajouter dans le ~/.bashrc **export QUILT_PATCHES=debian/patches**

1. vérifier que l'environnement n'est pas déjà établi : **cat ~/.bashrc | grep patches**
2. sinon **echo "export QUILT_PATCHES=debian/patches" >> ~/.bashrc**

Avant de modifier des fichiers du patch

Dans le répertoire natif mkd-0~120302

1. vérifier qu'il n'existe pas déjà des patches :
   quilt applied
2. créer le répertoire patch : **mkdir debian/patches**
3. lancer la commande (dquilt pour debian) :
   → *Le patch fix-uploads.patch est maintenant au sommet*
   quilt new fix-uploads.patch
   Deux répertoire viennent d'être mis à jour ou créés depuis le répertoire courant :
   debian/patches et un caché **.pc** qui contient 'applied-patches' et un répertoire *fix-uploads-patch* avec le nom du patch en tête de liste, le sous répertoire fix-uploads.patch, et trois fichiers de construction cachés.
4. ajouter le (ou plusieurs) fichier à modifier en tête du patch avant toute modification. ***Les fichiers déjà modifiés ne peuvent pas être ajoutés***.
   quilt add Makefile.
   → *Le fichier Makefile a été ajouté au patch fix-uploads.patch*

Modifier les fichiers *(Makefile)*

```
#!/bin/make
# File: Makefile for packaging mkd on debian or ubuntu systems

######################################################
# Updated by JPL 2012-05-22
# Amended by JPL 2012-06-08
# Amended by Clara for Wikibooks 2014-01-12, 2014-01-16
#
# Copyright: ©EELL http://edeulo.free.fr/wiki/index.php/Projet_mkd#LE_DROIT_DE_CO
PIE:
# For tests type the command "make -d install DESTDIR=../tests" for tests
######################################################

BIN    = $(DESTDIR)/usr/bin
CATMANDIR = $(DESTDIR)/var/cache/man
ETC    = $(DESTDIR)/etc
DOC    = $(DESTDIR)/usr/share/doc
ICONS = $(DESTDIR)/usr/share/icons/hicolor
LANG   = $(DESTDIR)/usr/share/locale
MAN    = $(DESTDIR)/usr/share/man
MIME   = $(DESTDIR)/usr/share/mime

PGM       = mkd
SRCS     = mkd.c asm.c cpp.c
HDRS     = version.h asm.h cpp.h find.inc.h internationalisation.h \
 find.inc.c basic.inc.c fortran.inc.c pascal.inc.c shell.inc.c tri.inc.c syn-fr.i
 syn-eng.i
OBJS     = mkd.o asm.o cpp.o
LIBS     =
CFLAGS   =  -O
LDFLAGS =
SPLINTFLAG = -weak

# create bin files:
# force recompilation $(SRCS) $(HDRS)
$(PGM):
          # see clean : rm -f $(PGM)
          gcc -c $(CFLAGS) $(SRCS) $(LIBS)
          gcc -o $(LDFLAGS) $(PGM) $(OBJS) $(LIBS)
          chmod 755 $(PGM)
          # -@strip $(PGM) # clean ASM and Link reloc

install: $(PGM)
          # create $(DESTDIR), sub-directories, and copy executable in bin direct
ory:
          install -d $(BIN) $(CATMANDIR) $(ETC) $(DOC) $(ICONS) $(LANG) $(MAN) $(
MIME)
          install $(PGM) $(BIN)/.
          #
          # install docs:
```

```
	# see ./install_docs
	# install -d $(DOC)/$(PGM)/html
	# install -m644 doc/*.html $(DOC)/$(PGM)/html/.
	#
	# install icons and update cache: see update in post-install package.
	# see ./install_icons
	install -d $(ICONS)/256x256/apps $(ICONS)/48x48/apps $(ICONS)/32x32/app
s $(ICONS)/16x16/apps
	chmod 644 icons/*
	cp -f icons/mkd-256x256.png $(ICONS)/256x256/apps/mkd.png
	cp -f icons/mkd-48x48.png $(ICONS)/48x48/apps/mkd.png
	cp -f icons/mkd-32x32.png $(ICONS)/32x32/apps/mkd.png
	cp -f icons/mkd-16x16.png $(ICONS)/16x16/apps/mkd.png
	if [ -e "/usr/share/icons/hicolor/16x16/apps/mkd.png" ]; \
		then gtk-update-icon-cache -t /usr/share/icons/hicolor; fi
	#
	# install languages:
	# see ./install_languages
	# Installs for tests the local languages, 'de', 'es', 'fr', it, ro, etc
.:
	install -d $(LANG)/en/LC_MESSAGES $(LANG)/de/LC_MESSAGES $(LANG)/fr/LC_
MESSAGES
	chmod 644 lang/*
	cp -f lang/mkd_en.mo $(LANG)/en/LC_MESSAGES/mkd.mo
	cp -f lang/mkd_fr.mo $(LANG)/fr/LC_MESSAGES/mkd.mo
	#
	# install manuals:
	# see ./install_manuals
	## install default manual:
	install -D -m644 lang/mkd_en.1.gz $(MAN)/man1/mkd.1.gz
	#
	## install 'fr' manual:
	# if [ -d "/usr/share/man/fr.UTF-8/man1" ]; \
	#	then \
			install -d $(MAN)/fr.UTF-8/man1; \
			cp -f  lang/mkd_fr.1.gz $(MAN)/fr.UTF-8/man1/mkd.1.g
z; \
	#	elif [ -d "/usr/share/man/fr/" ]; \
	#		then \
	#			install -d $(MAN)/fr/man1; \
	#			cp -f lang/mkd_fr.1.gz $(MAN)/fr/man1/mkd.
1.gz ; \
	# fi
	#
	# install catman only if selected manual exist:
	# see ./install_catman
	# if [ -e $(MAN)"/man1" ]; then catman -M $(MAN)/man1/mkd.1.gz; fi
	# if [ -e $(MAN)"/fr.UTF-8/man1/mkd.1.gz" ]; then catman -M $(MAN)/fr.U
TF-8/man1/mkd.1.gz; fi
	# if [ -e $(MAN)"/fr/man1/mkd.1.gz" ]; then catman -M $(MAN)/fr/man1/mk
d.1.gz ; fi
	#
	# install and update MIME database: see update in post-install package
```

```
          # see install_mime_database
          install -D -m644  mkdw.xml $(MIME)/packages/mkdw.xml
          # if [-e "/usr/share/mime/packages/mkdw.xml" ]; then update-mime-databa
se /usr/share/mime/; fi
          # clean
          rm -f $(PGM)
          rm -f *.o
          rm -f *~

# create man only in programmers source file; not for packaging
# type command "make -d mkd.1" to create updated file "manuals"
# mkd.1: manuals
          # if [ -e "/usr/bin/mkd" ]; \
          # then \
          #         mkd -Ct F manuals mkd.1; \
          #         gzip -cf mkd.1 > lang/mkd_fr.1.gz; \
          #         mkd -Ctw M manuals mkd.1; \
          #         gzip -cf mkd.1 > lang/mkd_en.1.gz; \
          #         rm -f mkd.1; \
          # fi

clean:
          rm -f $(PGM)
          rm -f *.o
          rm -f *~
```

Après la modification des fichiers du patch

1. rafraîchir les modifications dans 'fix-uploads.patch'
   quilt refresh

→ *Patch fix-uploads.patch rafraîchi*

1. valider le patch[5] (http://dep.debian.net/deps/dep3/) : **quilt header -e fix-uploads.patch** pour éditer le correctif
   Description : Mise en commentaire des installations conditionnelles du fichier Makefile
   Author: Clara <clara@mail.com>
   Origin: Erreur au téléversement
   etc.
   → *En-tête du patch fix-uploads.patch remplacé*
2. voir les différences : **quilt diff**

```
Index: mkd-0~120302/Makefile
===================================================================
--- mkd-0~120302.orig/Makefile  2014-01-13 12:04:22.000000000 +0100
+++ mkd-0~120302/Makefile       2014-01-17 14:35:01.186811564 +0100
@@ -4,7 +4,7 @@
 ######################################################
 # Updated by JPL 2012-05-22
 # Amended by JPL 2012-06-08
```

```
-# Amended by Clara for Wikibooks 2014-01-12
+# Amended by Clara for Wikibooks 2014-01-12, 2014-01-16
 #
 # Copyright: <C2><A9>EELL http://edeulo.free.fr/wiki/index.php/Projet_mkd#LE_DRO
IT_DE_COPIE:
 # For tests type the command "make -d install DESTDIR=../tests" for tests
@@ -73,26 +73,26 @@
        install -D -m644 lang/mkd_en.1.gz $(MAN)/man1/mkd.1.gz
        #
        ## install 'fr' manual:
-       if [ -d "/usr/share/man/fr.UTF-8/man1" ]; \
-               then \
-                       install -d $(MAN)/fr.UTF-8/man1;\
+       # if [ -d "/usr/share/man/fr.UTF-8/man1" ]; \
+       #       then \
+                       install -d $(MAN)/fr.UTF-8/man1; \
                        cp -f  lang/mkd_fr.1.gz $(MAN)/fr.UTF-8/man1/mkd.1.gz; \
-               elif [ -d "/usr/share/man/fr/" ]; \
-                       then \
-                               install -d $(MAN)/fr/man1; \
-                               cp -f lang/mkd_fr.1.gz $(MAN)/fr/man1/mkd.1.gz ;
\
-       fi
+       #       elif [ -d "/usr/share/man/fr/" ]; \
+       #               then \
+       #                       install -d $(MAN)/fr/man1; \
+       #                       cp -f lang/mkd_fr.1.gz $(MAN)/fr/man1/mkd.1.gz ;
\
+       # fi
        #
        # install catman only if selected manual exist:
        # see ./install_catman
-       if [ -e $(MAN)"/man1" ]; then catman -M $(MAN)/man1/mkd.1.gz; fi
-       if [ -e $(MAN)"/fr.UTF-8/man1/mkd.1.gz" ]; then catman -M $(MAN)/fr.UTF-8
/man1/mkd.1.gz; f
-       if [ -e $(MAN)"/fr/man1/mkd.1.gz" ]; then catman -M $(MAN)/fr/man1/mkd.1.
gz ; fi
+       # if [ -e $(MAN)"/man1" ]; then catman -M $(MAN)/man1/mkd.1.gz; fi
+       # if [ -e $(MAN)"/fr.UTF-8/man1/mkd.1.gz" ]; then catman -M $(MAN)/fr.UTF
-8/man1/mkd.1.gz;
+       # if [ -e $(MAN)"/fr/man1/mkd.1.gz" ]; then catman -M $(MAN)/fr/man1/mkd.
1.gz ; fi
        #
        # install and update MIME database: see update in post-install package
        # see install_mime_database
        install -D -m644  mkdw.xml $(MIME)/packages/mkdw.xml
-       if [-e "/usr/share/mime/packages/mkdw.xml" ]; then update-mime-database /
usr/share/mime/;
+       # if [-e "/usr/share/mime/packages/mkdw.xml" ]; then update-mime-database
 /usr/share/mime/
        # clean
        rm -f $(PGM)
        rm -f *.o
```

```
@@ -100,17 +100,18 @@
```

3. incrémenter la version dans changelog :
dch -i "Add patch fix-uploads (Correctif de téléversement ppa Launchpad)"
4. éditer le fichier debian/changelog et remplacer UNRELEASED par saucy
changelog :

```
mkd (0~120302-1ubuntu1) saucy; urgency=low

  * Add patch fix-uploads (Correctif de téléversement ppa Launchpad)

 -- Edeulo <edeulo@jpl-ubuntu-13.04>  Fri, 17 Jan 2014 15:05:36 +0100

mkd (0~120302-1) saucy; urgency=low

  * Initial release

 -- Edeulo <edeulo@free.fr>  Sun, 12 Jan 2014 18:20:02 +0100
```

Créer et tester les paquets

vérifier la compilation

```
make -d install DESTDIR=../tests
```

1. vérifier que les fichiers listés dans debian/mkd.install se trouvent bien dans le répertoire ../tests/usr

créer les paquets locaux et on vérifie le contenu

```
debuild -i -us -uc -b
```

ls -l ..

drwxr-xr-x 2 edeulo edeulo 4096 janv. 17 14:33 Fichiers-d'origine
-rwx--x--x 1 edeulo edeulo 375 mars 12 2013 KONSOLE
drwx------ 7 edeulo edeulo 4096 janv. 17 16:55 mkd-0~120302
-rw-r--r-- 1 edeulo edeulo 3264 janv. 16 16:10 mkd_0~120302-1.debian.tar.gz
-rw-r--r-- 1 edeulo edeulo 13030 janv. 17 16:55 mkd_0~120302-1ubuntu1_amd64.build
-rw-r--r-- 1 edeulo edeulo 1072 janv. 17 16:55 mkd_0~120302-1ubuntu1_amd64.changes
-rw-r--r-- 1 edeulo edeulo 176286 janv. 17 16:55 mkd_0~120302-1ubuntu1_amd64.deb
-rw-r--r-- 1 edeulo edeulo 205384 janv. 16 15:56 mkd_0~120302.orig.tar.xz
-rw-r--r-- 1 edeulo edeulo 2538 janv. 17 16:55 mkd-doc_0~120302-1ubuntu1_all.deb
drwxr-xr-x 5 edeulo edeulo 4096 janv. 17 16:42 tests

créer les fichiers à transférer

```
debuild -S -sa -k230B1A04
```

```
 dpkg-source -b mkd-0~120302
dpkg-source: avertissement: Le numéro de version indique des modifications Ubuntu
 mais le champ « Maintainer: » ne comporte pas d'adresse Ubuntu
dpkg-source: avertissement: Le numéro de version indique des modifications Ubuntu
 mais il n'existe pas de champ « XSBC-Original-Maintainer: »
dpkg-source: info: utilisation du format source « 3.0 (quilt) »
dpkg-source: info: construction de mkd à partir de ./mkd_0~120302.orig.tar.xz
dpkg-source: erreur: impossible d'identifier les changements de icons/mkd_0~12030
2-1.debian.tar.gz : contenu d'un fichier binaire modifié
dpkg-source: erreur: Ajoutez icons/mkd_0~120302-1.debian.tar.gz dans debian/sourc
e/include-binaries si vous souhaitez conserver le binaire modifié dans le ficher
tar debian
dpkg-source: erreur: modifications non représentables des sources
dpkg-buildpackage: erreur: dpkg-source -b mkd-0~120302 a produit une erreur de so
rtie de type 2
debuild: fatal error at line 1361:
dpkg-buildpackage -rfakeroot -d -us -uc -S -sa failed
```

enregistrer les modifications

```
dpkg-source --commit
```

relancer debuild -S -sa -k230B1A04

→ Successfully signed dsc and changes files

drwxr-xr-x 2 edeulo edeulo 4096 janv. 17 14:33 Fichiers-d'origine
-rwx--x--x 1 edeulo edeulo 375 mars 12 2013 KONSOLE
drwx------ 7 edeulo edeulo 4096 janv. 17 17:58 mkd-0~120302
-rw-r--r-- 1 edeulo edeulo 3264 janv. 16 16:10 mkd_0~120302-1.debian.tar.gz
-rw-r--r-- 1 edeulo edeulo 13358 janv. 17 17:58 mkd_0~120302-1ubuntu1_amd64.build
-rw-r--r-- 1 edeulo edeulo 1072 janv. 17 17:58 mkd_0~120302-1ubuntu1_amd64.changes
-rw-r--r-- 1 edeulo edeulo 176286 janv. 17 17:58 rouge|mkd_0~120302-1ubuntu1_amd64.deb
-rw-r--r-- 1 edeulo edeulo 7765 janv. 17 18:25 **mkd_0~120302-1ubuntu1.debian.tar.gz**
-rw-r--r-- 1 edeulo edeulo 1514 janv. 17 18:25 **mkd_0~120302-1ubuntu1.dsc**
-rw-r--r-- 1 edeulo edeulo 1755 janv. 17 18:25 **mkd_0~120302-1ubuntu1_source.build**
-rw-r--r-- 1 edeulo edeulo 1839 janv. 17 18:25 **mkd_0~120302-1ubuntu1_source.changes**
-rw-r--r-- 1 edeulo edeulo 205384 janv. 16 15:56 mkd_0~120302.orig.tar.xz
-rw-r--r-- 1 edeulo edeulo 2534 janv. 17 17:58 mkd-doc_0~120302-1ubuntu1_all.deb
drwxr-xr-x 5 edeulo edeulo 4096 janv. 17 16:42 tests

Tester, simuler, l'envoi au dépôt

1. Vérifier le paquet
   dput -ol ppa:jean-paul-louyot/exercises mkd_0~120302-1ubuntu1_source.changes
2. Simuler le dépôt
   dput -s ppa:jean-paul-louyot/exercises mkd_0~120302-1ubuntu1_source.changes
3. **Tenter le dépôt du paquet sur Launchpad**; il peut encore y avoir un rejet.

Mettre à jour

En cas d'échec de transfert vers un dépôt, il est souvent plus facile d'effacer le dépôt en échec et de faire une mise à jour après corrections. (Le fichier déposé ne disparaît jamais dans les listes).

* La somme de contrôle des fichiers (Checksum) ne doit pas changer pour une surcharge de version (voir l'option -f de dput).
* En cas d'échec de dépôt par une différence de *checksum* vous **devez** changer le numéro de version.

On fait généralement des mises à jour d'une application en conservant le numéro de version suivi d'un tiret et d'un chiffre de version des mêmes sources.

Exemple

mkd-0~120508 devient mkd-0~120508-1 puis mkd-0~120508-2

En général, les sources compilables restent les mêmes. On révise, ou on ajoute, des traductions, des images, etc.

Déroulement normal des opérations de mise à jour

Cette opération consiste à récupérer les fichiers de la version précédente.
Elle devra être téléchargée du dépôt
On peut aussi reprendre les fichiers locaux, mais le risque de se voir refuser un nouveau dépôt est grand. Les fichiers du dépôt ne sont pas nécessairement identiques aux fichiers locaux.

On récupère

1. le fichier *.orig.tar.gz* de la version précédente
   Ce fichier est le même depuis la création de l'application et tout changement dans ce fichier d'origine sera refusé par le serveur de dépôt.
2. le fichier *.debian.tar.gz* de la même version. Ce fichier contient toutes les informations pour créer une mise à jour.
3. éventuellement tous les autres fichiers.

On peut faire un nouveau patch. La version *ubuntu2* **remplacera** la version *ubuntu1* sur le serveur du dépôt.

Notre opération de mise à jour

Quelle mise à jour ?

1. On veut ajouter les messages en langue allemande.
2. On veut créer un seul fichier, car le fichier commun .all.deb n'a pas d'intérêt pour nous. Nous avons peu de documents à proposer.
3. La version ? *mkd (0~120302)*
4. Le nom du paquet ? *mkd*
5. Le nom des paquets debian .deb ? *mkd (0~120302-2)*

Créer une nouvelle version *origine* sans paquet version doc

1. copier l'archive *version dans un répertoire* mkd-120302-exercice-2/fichiers-d'origine
2. copier *mkd_0~120302-1ubuntu1.debian.tar.gz* le décompresser dans le répertoire *mkd-120302-exercice-2/fichiers-d'origine/.*
3. décompresser ces fichiers
4. déplacer le répertoire décompressé mkd-0~120302 dans *mkd-120302-exercice-2/."*

Créer les nouvelles directives d'empaquetage (DEBIAN)

- Début d'un petit fichier de travail *Build* exécutable :

```
#!/bin/bash
echo "supprimer un éventuel fichier .orig.tar.gz"
rm -f *.orig.tar.gz
echo " créer le fichier orig .tar.gz "
tar cvzf mkd_0~120302.orig.tar.gz mkd-0~120302
echo " entrant dans le répertoire mkd-0~120302"
cd mkd-0~120302
echo " action : dh_make -c GPL -e edeulo@free.fr -s -f ../mkd-0~120302.tar.gz -p
mkd"
echo " ATTENTION"
echo " le répertoire debian ne sera pas modifié si il existe !!!"
dh_make -c GPL -e edeulo@free.fr -s -f ../mkd_0~120302.orig.tar.gz -p mkd
echo " sortant du répertoire mkd-0~120302"
cd ..
```

Options :
-c GPL pour le copyright
-e <email> pour l'adresse du mainteneur
-s binaire seul
-n natif
-f <fichier> fichier d'origine
-p <version> version de paquet

```
 entrant dans le répertoire mkd-0~120302
 action : dh_make -c GPL -e edeulo@free.fr -s -f ../mkd-0~120302.tar.gz -p mkd
 ATTENTION
 le répertoire debian ne sera pas modifié si il existe !!!
Maintainer name  : Edeulo
Email-Address    : edeulo@free.fr
Date             : Mon, 20 Jan 2014 16:58:42 +0100
Package Name     : mkd
Version          : 0~120302
License          : gpl3
Type of Package  : Single
Hit <enter> to confirm:
Skipping creating ../mkd_0~120302.orig.tar.gz because it already exists
Done. Please edit the files in the debian/ subdirectory now. You should also
check that the mkd Makefiles install into $DESTDIR and not in / .
```

```
sortant du répertoire mkd-0~120302
```

Mise à jour des fichiers du répertoire source

Ajouter les fichiers de langue mkd_de.mo et mkd_de.1.gz dans le répertoire 'lang'

Modifier le Makefile

1. Remplacer le Makefile par celui de l'exercice-1
2. Éditer le Makefile
3. Supprimer les lignes de commentaires inutiles
4. Ajouter les lignes d'installation de la langue allemande

```
#!/bin/make
# File: Makefile for packaging mkd on debian or ubuntu systems

######################################################
# Updated by JPL 2012-05-22
# Amended by JPL 2012-06-08
# Amended by Clara for Wikibooks 2014-01-12, 2014-01-16, 2014-01-20
#
# Copyright: ©EELL http://edeulo.free.fr/wiki/index.php/Projet_mkd#LE_DROIT_DE_CO
PIE:
# For tests type the command "make -d install DESTDIR=../tests" for tests
######################################################

BIN    = $(DESTDIR)/usr/bin
CATMANDIR = $(DESTDIR)/var/cache/man
ETC    = $(DESTDIR)/etc
DOC    = $(DESTDIR)/usr/share/doc
ICONS = $(DESTDIR)/usr/share/icons/hicolor
LANG   = $(DESTDIR)/usr/share/locale
MAN    = $(DESTDIR)/usr/share/man
MIME   = $(DESTDIR)/usr/share/mime

PGM       = mkd
SRCS     = mkd.c asm.c cpp.c
HDRS     = version.h asm.h cpp.h find.inc.h internationalisation.h \
 find.inc.c basic.inc.c fortran.inc.c pascal.inc.c shell.inc.c tri.inc.c syn-fr.i
 syn-eng.i
OBJS     = mkd.o asm.o cpp.o
LIBS     =
CFLAGS   =  -O
LDFLAGS =
SPLINTFLAG = -weak

# create bin files:
# force recompilation $(SRCS) $(HDRS)
$(PGM):
          # see clean : rm -f $(PGM)
```

```
	gcc -c $(CFLAGS) $(SRCS) $(LIBS)
	gcc -o $(LDFLAGS) $(PGM) $(OBJS) $(LIBS)
	chmod 755 $(PGM)
	# -@strip $(PGM) # clean ASM and Link reloc

install: $(PGM)
	# create $(DESTDIR), sub-directories, and copy executable in bin direct
ory:
	install -d $(BIN) $(CATMANDIR) $(ETC) $(DOC) $(ICONS) $(LANG) $(MAN) $(
MIME)
	install $(PGM) $(BIN)/.
	#
	# install icons and update cache: see update in post-install package.
	# see ./install_icons
	install -d $(ICONS)/256x256/apps $(ICONS)/48x48/apps $(ICONS)/32x32/app
s $(ICONS)/16x16/apps
	chmod 644 icons/*
	cp -f icons/mkd-256x256.png $(ICONS)/256x256/apps/mkd.png
	cp -f icons/mkd-48x48.png $(ICONS)/48x48/apps/mkd.png
	cp -f icons/mkd-32x32.png $(ICONS)/32x32/apps/mkd.png
	cp -f icons/mkd-16x16.png $(ICONS)/16x16/apps/mkd.png
	#if [ -e "/usr/share/icons/hicolor/16x16/apps/mkd.png" ]; \
	#         then gtk-update-icon-cache -t /usr/share/icons/hicolor; fi
	#
	# install languages:
	# see ./install_languages
	# Installs for tests the local languages, 'de', 'es', 'fr', it, ro, etc
.:
	install -d $(LANG)/en/LC_MESSAGES $(LANG)/de/LC_MESSAGES $(LANG)/fr/LC_
MESSAGES
	chmod 644 lang/*
	cp -f lang/mkd_en.mo $(LANG)/en/LC_MESSAGES/mkd.mo
	cp -f lang/mkd_fr.mo $(LANG)/fr/LC_MESSAGES/mkd.mo
	cp -f lang/mkd_de.mo $(LANG)/de/LC_MESSAGES/mkd.mo
	#
	# install manuals:
	# see ./install_manuals
	## install default manual:
	install -D -m644 lang/mkd_en.1.gz $(MAN)/man1/mkd.1.gz
	#
	## install 'fr' manual:
	install -d $(MAN)/fr.UTF-8/man1; \
	cp -f  lang/mkd_fr.1.gz $(MAN)/fr.UTF-8/man1/mkd.1.gz; \
	## install 'de' manual:
	install -d $(MAN)/de/man1; \
	cp -f  lang/mkd_de.1.gz $(MAN)/de/man1/mkd.1.gz; \
	#
	# install and update MIME database: see update in post-install package
	# see install_mime_database
	install -D -m644  mkdw.xml $(MIME)/packages/mkdw.xml
	# if [-e "/usr/share/mime/packages/mkdw.xml" ]; then update-mime-databa
se /usr/share/mime/; fi
	# re-clean
```

```
	rm -f $(PGM)
	rm -f *.o
	rm -f *~

clean:
	rm -f $(PGM)
	rm -f *.o
	rm -f *~
```

Valider les modifications par un examen du fichier des tests

make -d install DESTDIR=../tests *(Dans le répertoire des sources)*

Vérifier que tout est en ordre et que la langue allemande est bien à sa place

Ajouter un fichier README-INSTALL

```
Remarque importante

La commande mkd est une commande UNIX comme mkdir, ls, grep, ...
L'installation universelle de la commande mkd est exécutée dans le répertoire
des sources par un super utilisateur :

1. Télégargez le fichier d'origine sur le serveur Launchpad
   https://launchpad.net/~jean-paul-louyot/+archive/mkd/+copy-packages

2. Décompactez le fichier mkd_<version>.tar.gz ou .orig.tar.gz
3. Tapez la commande sudo make install pour les versions Debian ou dérivés ou
   sous utilisateur root (Voir la commande su)

4. Pour désintaller tapez la command ./uninstall dans le répertoire des sources.

Les paquets d'installation peuvent être utilisés par facilité.
Les paquets debian peuvent être mués en paquets rpm ou slp avec la commande
alien :

   alien [--to-deb] [--to-rpm] [--to-tgz] [--to-slp] [options] file [...]
```

Mise à jour du répertoire debian

Commencer par mettre ce répertoire à jour

1. le supprimer dans le répertoire des sources
2. Relancer la commande Build *(dans le répertoire de l'exercice)*
3. Entrer dans le répertoire debian

4. Éditer le fichier changelog et remplacer ***mkd (0~120302-1) unstable*** par *"""mkd (0~120302-2) saucy*. **La version courante de linux ubuntu est *saucy* en janvier 2014. Supprimer ce qui suit *Initial release***
5. Editer le fichier *control* et remplacer *unknown* à la ligne *Section* par *devel*. Pour le reste, se reporter à l'exercice précédent

```
Source: mkd
Section: devel
Priority: extra
Maintainer: Edeulo <edeulo@free.fr>
Build-Depends: debhelper (>= 8.0.0)
Standards-Version: 3.9.4
Homepage: https://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)/Dépôt_parra
iné_du_paquet_"original"_mkd-131215/Mise_à_jour_de_paquets#Mettre_à_jour
#Vcs-Git: git://git.debian.org/collab-maint/mkd-120302-2.git
#Vcs-Browser: http://git.debian.org/?p=collab-maint/mkd-120302-2.git;a=summary

Package: mkd
Architecture: any
Depends: ${shlibs:Depends}, ${misc:Depends}
Description: Exercice "Intention d'empaqueter"
 Voir "mkd (extracteur de documents)" sur Wikibooks
 Ce paquet debian Ubuntu est destiné à installer le binaire
```

6. Éditer le fichier *copyright* et remplacer la ligne Source: par

Source: http://edeulo.free.fr/pub/Packaging-exercises/mkd-120302-exercice-2.tar.xz
Puis recopier le reste de Files: * ... jusqu'à la fin.

```
Format: http://www.debian.org/doc/packaging-manuals/copyright-format/1.0/
Upstream-Name: mkd
Source: http://edeulo.free.fr/pub/Packaging-exercises/mkd-120302-exercice-2.tar.x
z

Files: *
Copyright: 1986-2001 Jean-Paul Louyot <jpl@cem2.univ-montp2.fr>
           2001-2014 Jean-Paul Louyot <jean.paul.louyot@gamail.com>
           2001-2014 Clara Jimenez <clara.jimenez2@gmail.com>
License: EUPL-1.1
 This package is free software; you can redistribute it and/or modify
 it under the terms of the European Licence EUPL.
 http://edeulo.free.fr/wiki/index.php/Projet_mkd#LE_DROIT_DE_COPIE:
 .
 Sauf obligation légale ou contractuelle écrite, le logiciel distribué sous la
 Licence est distribué «en l'état»,
 SANS GARANTIES OU CONDITIONS QUELLES QU'ELLES SOIENT, expresses ou implicites.
 Consultez la Licence pour les autorisations et les restrictions linguistiques
 spécifiques relevant de la Licence.

# If you want to use GPL v2 or later for the /debian/* files use
```

```
# the following clauses, or change it to suit. Delete these two lines
Files: debian/*
Copyright: 2014 Edeulo <edeulo@free.fr>
License: GPL-2+
 This package is free software; you can redistribute it and/or modify
 it under the terms of the GNU General Public License as published by
 the Free Software Foundation; either version 2 of the License, or
 (at your option) any later version.
 .
 This package is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 GNU General Public License for more details.
 .
 You should have received a copy of the GNU General Public License
 along with this program. If not, see <http://www.gnu.org/licenses/>
 .
 On Debian systems, the complete text of the GNU General
 Public License version 2 can be found in "/usr/share/common-licenses/GPL-2".

# Please also look if there are files or directories which have a
# different copyright/license attached and list them here.
# Please avoid to pick license terms that are more restrictive than the
# packaged work, as it may make Debian's contributions unacceptable upstream.
```

7. Vérifier que le fichier doc contienne bien 'README-INSTALL'
8. Supprimer tous les fichiers .ex, .EX; et README.*
9. Copier les fichiers *postinst* et *postrm* de l'exercice précédent. puisque rien n'a changé ...

Contenu du répertoire debian après mise à jour

-rw-r--r-- 1 edeulo edeulo 128 janv. 20 15:36 changelog
-rw-r--r-- 1 edeulo edeulo 2 janv. 20 15:26 compat
-rw-r--r-- 1 edeulo edeulo 696 janv. 20 15:36 control
-rw-r--r-- 1 edeulo edeulo 1394 janv. 20 15:26 copyright
-rw-r--r-- 1 edeulo edeulo 34 janv. 20 15:26 docs
-rw-r--r-- 1 edeulo edeulo 1285 janv. 13 10:43 postinst
-rw-r--r-- 1 edeulo edeulo 1092 janv. 13 10:41 postrm
-rwxr-xr-x 1 edeulo edeulo 442 janv. 20 15:26 rules
drwxr-xr-x 2 edeulo edeulo 4096 janv. 20 15:26 source

Créer les paquets locaux

```
debuild -i -us -uc - b
cd ..
edeulo@jpl-ubuntu-13:~/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice
-2$ ls -al
-rwxr-xr-x 1 edeulo edeulo    530 janv. 20 17:06 Build
drwxr-xr-x 4 edeulo edeulo   4096 janv. 20 14:12 Fichiers-d'origine
drwxr-xr-x 5 edeulo edeulo   4096 janv. 20 16:55 Fichiers-intermédi
ares
-rwx--x--x 1 edeulo edeulo    375 mars  12  2013 KONSOLE
```

```
drwx------ 6 edeulo edeulo   4096 janv. 20 17:34 mkd-0~120302
-rw-r--r-- 1 edeulo edeulo  11536 janv. 20 17:34 mkd_0~120302-2_amd
64.build
-rw-r--r-- 1 edeulo edeulo    638 janv. 20 17:34 mkd_0~120302-2_amd
64.changes
-rw-r--r-- 1 edeulo edeulo 222430 janv. 20 17:34 mkd_0~120302-2_amd
64.deb
-rw-r--r-- 1 edeulo edeulo 290955 janv. 20 16:58 mkd_0~120302.orig.
tar.gz
drwxr-xr-x 5 edeulo edeulo   4096 janv. 20 11:57 tests
```

Créer, vérifier, et tester les fichiers de transfert au dépôt

```
cd mkd-0~12030
debuild -S -sa -k230B1A04
```

Les fichiers nécessaires au transfert ont été créés sans erreur

- Vérifier le paquet
  dput -ol ppa:jean-paul-louyot/exercises mkd_0~120302-2_source.changes

Good signature on /home/edeulo/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice-2/mkd_0~120302-2.dsc.
Package includes an .orig.tar.gz file although the debian revision suggests that it might not be required. Multiple uploads of the .orig.tar.gz may be rejected by the upload queue management software.

- Simuler le dépôt
  dput -s ppa:jean-paul-louyot/exercises mkd_0~120302-2_source.changes

Même message d'avertissement : *Peut être refusé par la file d'attente de téléchargement*

- **Tenter malgré tout le dépôt du paquet sur Launchpad** :

On peut s'attendre à avoir un rejet.

Nouvelle version

Dans tous les exercice précédents, le seul but était d'apprendre à créer des paquets et à les envoyer en dépôt sur le serveur Launchpad.

Or notre application se veut universelle compilable sur tous les systèmes Unix, Linux, et accessoirement sous d'autres systèmes tels que MS-Windows ou Mac.
L'application se compile en ligne de commande à l'aide d'un Makefile ou un shell de commandes install *(install.sh, install.bat)*

Les paquets sont agréables à installer.
L'idée est donc de créer des paquets debian avec un original contenant des sources compilables sur tous les systèmes d'exploitation, pour n'importe quelle machine, quel que soit son processeur.

Pour ce faire, nous allons transposer tout le répertoires des sources dans un répertoire d'*origine/sources* et créer un Makefile spécialisé pour la construction de paquets.
Nous devrons aussi donner un maximum de renseignements pour installer l'application sur d'autres machines.

Créer le nouvel environnement de travail

1. Créer les répertoire contenant l'ensemble des éléments de travail
   mkdir mkd-120515-exercice-3 mkd-120515-exercice-3/mkd-120515
2. Copier dans ce répertoire les utilitaires *KONSOLE* et *Build* de l'exercice précédent.
3. Copier *mkd_0~120302.orig.tar.xz* de l'exercice-0 dans le répertoire *mkd-120515*
4. Décompacter *mkd_0~120302.orig.tar.xz* et renommer *src* le répertoire décompacté.
5. Entrer dans le répertoire src et lancer la commande :
   make -d install DESTDIR=../.

Cette commande va créer les répertoires utiles à la fabrication des paquets.

1. Vérifier le contenu du répertoire *mkd-120515/usr*

```
edeulo@jpl-ubuntu-13:~/Packaging/mkd/mkd_12XXXX/mkd-120515-exercice-3/mkd-120515/
usr$ find
.
./share
./share/mime
./share/mime/packages
./share/mime/packages/mkdw.xml
./share/doc
./share/man
./share/man/fr.UTF-8
./share/man/fr.UTF-8/man1
./share/man/fr.UTF-8/man1/mkd.1.gz
./share/man/man1
./share/man/man1/mkd.1.gz
./share/icons
./share/icons/hicolor
./share/icons/hicolor/256x256
./share/icons/hicolor/256x256/apps
```

```
./share/icons/hicolor/256x256/apps/mkd.png
./share/icons/hicolor/16x16
./share/icons/hicolor/16x16/apps
./share/icons/hicolor/16x16/apps/mkd.png
./share/icons/hicolor/32x32
./share/icons/hicolor/32x32/apps
./share/icons/hicolor/32x32/apps/mkd.png
./share/icons/hicolor/48x48
./share/icons/hicolor/48x48/apps
./share/icons/hicolor/48x48/apps/mkd.png
./share/locale
./share/locale/fr
./share/locale/fr/LC_MESSAGES
./share/locale/fr/LC_MESSAGES/mkd.mo
./share/locale/en
./share/locale/en/LC_MESSAGES
./share/locale/en/LC_MESSAGES/mkd.mo
./share/locale/de
./share/locale/de/LC_MESSAGES
./bin
./bin/mkd
```

On voit qu'il y manque la langue allemande pour les messages et le manuel

Créer le Makefile d'empaquetage

- Seul le binaire usr/bin/mkd doit être mis à jour pour les différents processeurs

Il faut donc

1. précompiler tous les fichiers d'accompagnement de l'application (langues, icônes, etc.),
2. mettre le binaire mkd à jour pour chaque processeur par le serveur Launchpad.

Construction du Makefile à l'aide du Makefile des sources

```
#!/bin/make
# File: Makefile for packaging mkd on debian or ubuntu systems
# https://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)_Dépôt_parrainé_du_p
aquet_"original"_mkd-131215/Mise_à_jour_de_paquets#Créer_le_Makefile_d'empaquetag
e

######################################################
# Updated by JPL 2012-05-22
# Amended by JPL 2012-06-08
# Amended by Clara for Wikibooks 2014-01-12, 2014-01-22
#
# Copyright: ©EELL http://edeulo.free.fr/wiki/index.php/Projet_mkd#LE_DROIT_DE_CO
PIE:
#
######################################################
```

```
PGM      = mkd
SRC       = src
BINDIR   = usr/bin
BINMODE = 755
CC       = cc
SRCS     = $(SRC)/mkd.c $(SRC)/asm.c $(SRC)/cpp.c
#
HDRS     = $(SRC)/version.h $(SRC)/asm.h $(SRC)/cpp.h $(SRC)/find.inc.h \
$(SRC)/internationalisation.h $(SRC)/find.inc.c $(SRC)/basic.inc.c $(SRC)/ \
$(SRC)/fortran.inc.c $(SRC)/pascal.inc.c $(SRC)/shell.inc.c $(SRC)/tri.inc.c \
$(SRC)/syn-fr.i $(SRC)/syn-eng.i
#
OBJS     =
LIBS     =
CFLAGS   =  -O
LDFLAGS =
SPLINTFLAG = -weak

$(PGM): $(SRCS) $(HDRS)
          # cette section doit provoquer une erreur si $(BINDIR) n'existe pas
          #-@echo "*********** update $(PGM) ***********"
          $(CC) $(CFLAGS) -o $(PGM) $(SRCS)
          -@if [ -d $(BINDIR) ]; then \
                    mv -f $(PGM) $(BINDIR)/$(PGM);\
                    chmod $(BINMODE) $(BINDIR)/$(PGM);\
          else \
                    echo "couldn't find $(BINDIR)";\
          fi

install:
          mkdir -p debian/mkd/usr/bin
          cp -rf usr debian/mkd/.

clean:
          rm -rf *~
          rm -f $(PGM)
          rm -f $(PGM) $(BINDIR)/$(PGM)
```

- vérifier que usr/bin/mkd est mis à jour : (heure de compilation actualisée)

```
edeulo@jpl-ubuntu-13:~/Packaging/mkd/mkd_12XXXX/mkd-120515-exercice
-3/mkd-120515$ \
make mkd
ls -l usr/bin/mkd
→ -rwxr-xr-x 1 edeulo edeulo 35122 janv. 22 15:49 usr/bin/mkd
```

Créer les conditions d'empaquetage debian avec dh_make

```
tar cvzf mkd-120515.tar.gz mkd-120515
cd mkd-120515/
dh_make -c GPL -e edeulo@free.fr -s -n -f mkd-120515.tar.gz
```

Modifier les fichiers comme dans les exercices précédents

debian/changelog

```
mkd (120515) saucy; urgency=low

  * Initial Release.

 -- Edeulo <edeulo@free.fr>  Wed, 22 Jan 2014 17:20:01 +0100
```

debian/control

```
Source: mkd
Section: devel
Priority: extra
Maintainer: Edeulo <edeulo@free.fr>
Build-Depends: debhelper (>= 8.0.0)
Standards-Version: 3.9.4
Homepage: https://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)/Dépôt_parra
iné_du_paquet_"original"_mkd-131215/Mise_à_jour_de_paquets#Nouvelle_version

Package: mkd
Architecture: any
Depends: ${shlibs:Depends}, ${misc:Depends}
Description: Exercice "Nouvelle version"
 Voir "mkd (extracteur de documents)" sur Wikibooks
 Ce paquet debian Ubuntu est destiné à installer la commande mkd
```

debian/copyright

```
Format: http://www.debian.org/doc/packaging-manuals/copyright-format/1.0/
Upstream-Name: mkd
Source: http://edeulo.free.fr/pub/Packaging-exercises/mkd-120302-exercice-3.tar.x
z
Files: *
Copyright: 1986-2001 Jean-Paul Louyot <jpl@cem2.univ-montp2.fr>
           2001-2014 Jean-Paul Louyot <jean.paul.louyot@gamail.com>
           2001-2014 Clara Jimenez <clara.jimenez2@gmail.com>
License: EUPL-1.1
```

```
 This package is free software; you can redistribute it and/or modify
 it under the terms of the European Licence EUPL.
 http://edeulo.free.fr/wiki/index.php/Projet_mkd#LE_DROIT_DE_COPIE:
 .
 Sauf obligation légale ou contractuelle écrite, le logiciel distribué sous la
 Licence est distribué «en l'état»,
 SANS GARANTIES OU CONDITIONS QUELLES QU'ELLES SOIENT, expresses ou implicites.
 Consultez la Licence pour les autorisations et les restrictions linguistiques
 spécifiques relevant de la Licence.

Files: debian/*
Copyright: 2014 Edeulo <edeulo@free.fr>
License: GPL-3.0+
 This program is free software: you can redistribute it and/or modify
 it under the terms of the GNU General Public License as published by
 the Free Software Foundation, either version 3 of the License, or
 (at your option) any later version.
 .
 This package is distributed in the hope that it will be useful,
 but WITHOUT ANY WARRANTY; without even the implied warranty of
 MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 GNU General Public License for more details.
 .
 You should have received a copy of the GNU General Public License
 along with this program. If not, see <http://www.gnu.org/licenses/>.
 .
 On Debian systems, the complete text of the GNU General
 Public License version 3 can be found in "/usr/share/common-licenses/GPL-3".
```

debian/docs

```
README-INSTALL
README-INSTALL.pdf
```

debian/README

```
The Debian Package mkd
----------------------------

Nouveau projet exercice N°4

 -- Edeulo <edeulo@free.fr>  Wed, 22 Jan 2014 17:20:01 +0100
```

recopier les fichiers debian/postinst et debian/postrm des exercices précédents

Ajouter le fichier *README-INSTALL* dans le répertoire origine *mkd-120515*

et README-INSTALL.pdf de l'exercice précédent :

README-INSTALL

```
Remarque importante

La commande mkd est une commande UNIX comme mkdir, ls, grep, ...
L'installation universelle de la commande mkd est exécutée dans le répertoire
des sources par un super utilisateur :

1. Télégargez le fichier d'origine sur le serveur Launchpad
   https://launchpad.net/~jean-paul-louyot/+archive/mkd/+copy-packages

2. Décompactez le fichier mkd_<version>.tar.gz ou .orig.tar.gz
3. Tapez la commande sudo make install pour les versions Debian ou dérivés ou
   sous utilisateur root (Voir la commande su)

4. Pour désinstaller tapez la command ./uninstall dans le répertoire des sources.

Les paquets d'installation peuvent être utilisés par facilité.
Les paquets debian peuvent être mués en paquets rpm ou slp avec la commande
alien :

   alien [--to-deb] [--to-rpm] [--to-tgz] [--to-slp] [options] file [...]
```

Créer le paquet local

Dans le répertoire origine :

```
debuild -i -us -uc -b
```

```
dpkg-deb : construction du paquet « mkd » dans « ../mkd_120515_amd64.deb ».
 dpkg-genchanges -b >../mkd_120515_amd64.changes
dpkg-genchanges: envoi d'un binaire - aucune inclusion de code source
 dpkg-source -i --after-build mkd-120515
dpkg-buildpackage: envoi d'un binaire seulement (aucune inclusion de code source)
Now running lintian...
Finished running lintian.
```

Relisant le fichier créé :*mkd_120515_amd64.deb* avec le gestionnaire d'archives on s'aperçoit que l'on a oublié la langue allemande :

1. copier les fichiers mkd-de.1.gz et mkd_de.mo dans src/lang
2. ajouter *cp -f lang/mkd_de.mo $(LANG)/de/LC_MESSAGES/mkd.mo* dans src/Makefiles (ligne ±69)

3. ajouter *install -D -m644 lang/mkd_de.1.gz $(MAN)/de/man1/mkd.1.gz* dans src/Makefile (ligne ±75)
4. relancer **make -d install DESTDIR=../.** dans le répertoire src
5. contrôler que la langue allemande est maintenant présente

./share/man/fr.UTF-8/man1/mkd.1.gz
./share/man/de/man1/mkd.1.gz
./share/man/man1/mkd.1.gz
...
./share/locale/fr/LC_MESSAGES/mkd.mo
./share/locale/en/LC_MESSAGES/mkd.mo
./share/locale/de/LC_MESSAGES/mkd.mo

Tout reconstruire

Build

```
#!/bin/bash
echo " créer le fichier d'origine native .tar.gz "
tar cvzf mkd-120515.tar.gz mkd-120515
echo " Il est inutile de recréer les fichiers debian"
echo " entrant dans le répertoire mkd-0~120302"
cd mkd-120515
echo " action : création du paquet debian : debuild -i -us -uc -b"
debuild -i -us -uc -b
echo " sortant du répertoire mkd-120515"
cd ..
echo " Contrôler le contenu du paquet .deb avec le gestionnaire d'archives"
echo " Vérifier la présence des langues"
echo " Vérifier le copyright et README-INSTALL dans /usr/share/doc/mkd"
read
```

Préparer l'envoi au dépôt Launchpad

- Créer les fichiers pour Launchpad

```
cd mkd-120515
debuild -S -sa -k230B1A04
```

Éditer et vérifier les fichiers : *mkd_120515_source.build, mkd_120515_source.changes, mkd_120515.dsc*

- Vérifier le paquet

```
cd ..
dput -ol ppa:jean-paul-louyot/exercises mkd_120515_source.changes
```

- Simuler le dépôt

```
dput -s ppa:jean-paul-louyot/exercises mkd_0~120302-2_source.changes
```

- **Tenter le dépôt du paquet sur Launchpad :**

Dépôt sur Launchpad

Pour déposer un paquet il faut disposer d'un compte de dépôt (PPA)[6] (http://doc.ubuntu-fr.org/ppa) et d'une clé PGP avec GnuPG

Création d'un dépôt pour les exercices

Pour la démonstration on a créé un nouvel espace :

Dans le ppa:jean-paul-louyot, on a créé le répertoire *exercises* avec le nouveau ppa *Do a package*
https://launchpad.net/~jean-paul-louyot/+archive/exercises

Launchpad - Premier exercice : ITP mkd_0~120302-1

La commande

dput ppa:jean-paul-louyot/exercises mkd_0~120302-1_source.changes
a correctement téléversé les fichiers suivants :

```
Uploading to ppa (via ftp to ppa.launchpad.net):
  Uploading mkd_0~120302-1.dsc: done.
  Uploading mkd_0~120302.orig.tar.xz: done.
  Uploading mkd_0~120302-1.debian.tar.gz: done.
  Uploading mkd_0~120302-1_source.changes: done.
Successfully uploaded packages.
```

Si vous vous êtes trompé dans l'envoi vous trouverez le fichier .upload qu'il faudra supprimer si vous désirez faire un nouvel envoi.

Réception du courrier après build

```
 * Source Package: mkd
 * Version: 0~120302-1
 * Architecture: amd64
 * Archive: jean-paul-louyot-exercises PPA
 * Component: main
 * State: Failed to build
 * Duration: 3 minutes
 * Build Log: https://launchpad.net/~jean-paul-louyot/+archive/exercises/+build/5
463514/+files/buildlog_ubuntu-saucy-amd64.mkd_0%7E120302-1_FAILEDTOBUILD.txt.gz
 * Builder: https://launchpad.net/builders/seaborgium
 * Source: not available
```

Edition du fichier Build Log *_FAILEDTOBUILD.txt.gz*

```
552 make[1]: Leaving directory `/build/buildd/mkd-0~120302'
553    dh_install -a
```

```
554 cp: cannot stat 'debian/tmp/usr/share/man/fr.UTF-8/man1/mkd.1.gz': No such fi
le or directory
555 dh_install: cp -a debian/tmp/usr/share/man/fr.UTF-8/man1/mkd.1.gz debian/mkd/
usr/share/man/fr.UTF-8/man1/ returned exit code 1
556 make: *** [binary-arch] Error 2
557 dpkg-buildpackage: error: /usr/bin/fakeroot debian/rules binary-arch gave err
or exit status 2
```

La construction des paquets s'est donc arrêtée faute de trouver le fichier mkd.1.gz. La construction de notre paquet local n'a pas généré d'erreur et le répetoire des tests que nous avons utilisé pour créer le fichier mkd.install montre bien le fichier mkd.1.gz dans le bon répertoire.

Pourquoi ça n'a pas fonctionné alors que le répertoire debian/tmp a bien été créé

Pour cela il faut pour cela relire la compilation dans *_FAILEDTOBUILD.txt.gz* :

```
## install 'fr' manual:
if [ -d "/usr/share/man/fr.UTF-8/man1" ]; \
          then \
                    install -d /build/buildd/mkd-0~120302/debian/tmp/usr/share/ma
n/fr.UTF-8/man1;\
                    cp -f  lang/mkd_fr.1.gz /build/buildd/mkd-0~120302/debian/tmp
/usr/share/man/fr.UTF-8/man1/mkd.1.gz; \
          elif [ -d "/usr/share/man/fr/" ]; \
                    then \
                              install -d /build/buildd/mkd-0~120302/debian/tmp/usr
/share/man/fr/man1; \
                              cp -f lang/mkd_fr.1.gz /build/buildd/mkd-0~120302/de
bian/tmp/usr/share/man/fr/man1/mkd.1.gz ; \
fi
```

On voit que la condition d'installation du fichier *mkd.1.gz* dépend de la machine sur laquelle on va effectuer la compilation.

Il est impératif d'éliminer les compilations conditionnelles

On peut installer

1. des fichiers, sans avoir recours à une copie conditionnelle.
2. les fichiers dans un répertoire déterminé (là où il sera installé avec le paquet). En cas de nécessité, on pourra déplacer ces fichiers par un script dans debian/post-install.

Reconstruction du paquet par modification du Makefile.

Launchpad - Deuxième exercice : Patch mkd_0~120302-1ubuntu1

```
dput ppa:jean-paul-louyot/exercises mkd_0~120302-1ubuntu1_source.ch
anges
```

→ message sur console

```
edeulo@jpl-ubuntu-13:~/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice-1$ dput ppa:j
ean-paul-louyot/exercises mkd_0~120302-1ubuntu1_source.changes
Checking signature on .changes
gpg: Signature faite le ven. 17 janv. 2014 18:25:20 CET avec la clef RSA d'identi
fiant 230B1A04
gpg: Bonne signature de « Editeurs Européens (Logiciels libres) <edeulo@free.fr>
»
gpg:                 alias « Jean-Paul Louyot (clé pour launchpad) <jean.paul.lou
yot@free.fr> »
Good signature on /home/edeulo/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice-1/mkd
_0~120302-1ubuntu1_source.changes.
Checking signature on .dsc
gpg: Signature faite le ven. 17 janv. 2014 18:25:20 CET avec la clef RSA d'identi
fiant 230B1A04
gpg: Bonne signature de « Editeurs Européens (Logiciels libres) <edeulo@free.fr>
»
gpg:                 alias « Jean-Paul Louyot (clé pour launchpad) <jean.paul.lou
yot@free.fr> »
Good signature on /home/edeulo/Packaging/mkd/mkd_12XXXX/mkd-120302-exercice-1/mkd
_0~120302-1ubuntu1.dsc.
Uploading to ppa (via ftp to ppa.launchpad.net):
  Uploading mkd_0~120302-1ubuntu1.dsc: done.
  Uploading mkd_0~120302.orig.tar.xz: done.
  Uploading mkd_0~120302-1ubuntu1.debian.tar.gz: done.
  Uploading mkd_0~120302-1ubuntu1_source.changes: done.
Successfully uploaded packages.
```

→ retour par courrier électronique

```
Rejected:
Unable to identify 'Edeulo':<edeulo@jpl-ubuntu-13.04> in launchpad
Further error processing not possible because of a critical previous error.
```

Consultation du dépôt

https://launchpad.net/~jean-paul-louyot/+archive/exercises
La version n'apparaît pas, on peut donc refaire le fichier _sources.changes avec une bonne signature :

Lecture du fichier _sources.changes.clear

Cliquer sur le fichier _source.changes.

Si vous avez la bonne clé vous obtiendrez le fichier _source.changes.clear

ligne 9 : Maintainer: Edeulo <edeulo@free.fr>
ligne 10 : Changed-By: Edeulo <edeulo@jpl-ubuntu-13.04>

Modifier le fichier debian/changelog

ligne 5 :
— -- Edeulo <edeulo@jpl-ubuntu-13.04> Fri, 17 Jan 2014 15:05:36 +0100
+ -- Edeulo <edeulo@free.fr> Fri, 17 Jan 2014 15:05:36 +0100

Recommencer

1. **debuild -S -sa -k230B1A04** dans le répertoire dorigine, et vérifier que le fichier *_source.changes* est bien correctement modifié,
2. déplacer ou supprimer *mkd_0~120302-1ubuntu1_source.ppa.upload* pour qu'il n'y ait pas une nouvelle erreur de transfert.
3. **dput ppa:jean-paul-louyot/exercises mkd_0~120302-1ubuntu1_source.changes** au dessus du répertoire d'origine.
4. Voir les fichiers compilés et empaquetés après un temps d'attente variable en fonction du nombre de paquets en instance ...

```
https://launchpad.net/~jean-paul-louyot/+archive/exercises/+packages
```

Remarques

1. L'archive debian ***mkd_0~120302-1ubuntu1.debian.tar.gz***, contient maintenant , en plus de sa propre archive ***debian***, une archive ***icon*** dans laquelle on retrouve l'ancienne archive *mkd_0~120302-1.debian.tar.gz*
   → ***debian*** habituel
   + ***incons/mkd_0~120302-1.debian.tar.gz*** ajouté
2. Le répertoire debian contient maintenant, en plus du fichier *format*, le fichier *includes-binaries* avec le texte "icons/mkd_0~120302-1.debian.tar.gz"
3. Sur Launchpad : Le nouveau dépôt, avec le patch, a remplacé celui qui avait échoué.

Launchpad - Troisième exercice : Mettre à jour

```
dput ppa:jean-paul-louyot/exercises mkd_0~120302-2_source.changes
```

Réponse par courriel

```
Accepted:
 OK: mkd_0~120302.orig.tar.gz
 OK: mkd_0~120302-2.debian.tar.gz
 OK: mkd_0~120302-2.dsc
     -> Component: main Section: devel

mkd (0~120302-2) saucy; urgency=low

  * Initial release
```

Les fichiers ont été acceptés. Pas de problème pour les sommes de contrôle; mais la compilation ?

Les paris sont ouverts

Réponse du serveur

Tout le monde a faux !

La nouvelle version 0~120302-2 remplace la version 0~120302-1ubuntu1 avec une différence de 3,8Kilo-Bytes Le serveur Launchpad a répondu *Erreur de compilation pour amd64* et *Compilation copmplète pour i386*

```
Changelog

mkd (0~120302-2) saucy; urgency=low

  * Initial release
 -- Edeulo <edeulo@free.fr>   Mon, 20 Jan 2014 15:26:11 +0100
Available diffs

diff from 0~120302-1ubuntu1 to 0~120302-2 (3.8 KiB)
Builds

[FAILEDTOBUILD] amd64
[FULLYBUILT] i386
Built packages

mkd Exercice "Mettre à jour"
Package files

mkd_0~120302-2.debian.tar.gz (2.8 KiB)
mkd_0~120302-2.dsc (1.4 KiB)
mkd_0~120302-2_i386.deb (215.8 KiB)
mkd_0~120302.orig.tar.gz (284.1 KiB)
```

Dans https://launchpad.net/~jean-paul-louyot/+archive/exercises/+build/5484224 on a la possibilité de recompiler la version amd64

[FAILEDTOBUILD] Failed to build Retry this build
Recompilation correcte pour amd64

Launchpad - Quatrième exercice : Nouvelle version

```
dput ppa:jean-paul-louyot/exercises mkd_120515_source.changes
```

→ Retour par courriel :

```
Accepted:
 OK: mkd_120515.tar.gz
 OK: mkd_120515.dsc
     -> Component: main Section: devel

mkd (120515) saucy; urgency=low

  * Initial Release.
```

Consultation du dépôt

https://launchpad.net/~jean-paul-louyot/+archive/exercises/+copy-packages → résulat :

```
Publishing details

Published 36 minutes ago
Changelog

mkd (120515) saucy; urgency=low

  * Initial Release.
 -- Edeulo <edeulo@free.fr>   Wed, 22 Jan 2014 17:20:01 +0100
Available diffs

diff from 0~120302-2 to 120515 (111.4 KiB)
Builds

[FULLYBUILT] amd64
[FULLYBUILT] i386
Built packages

mkd Exercice "Nouvelle version"
Package files

mkd_120515.dsc (1.1 KiB)
mkd_120515.tar.gz (446.3 KiB)
mkd_120515_amd64.deb (218.0 KiB)
mkd_120515_i386.deb (216.9 KiB)
```

Résumés, notes et références

Résumés des solutions

Exercice-0 *Intention d'empaqueter*

On y apprend à

1. modifier le fichier *Makefile* de l'application d'origine,
2. créer des ***paquets multiples*** et à modifier les fichiers du répertoire *debian*, en particulier pour les paquets multiples,
3. préparer les fichiers nécessaires à un dépôt sur Launchpad
4. déposer sur Launchpad une application testée et sécurisée par une clé personnelle
5. trouver les raisons de l'échec au dépôt
6. renvoyer les fichiers corrigés après une erreur de compilations sur Launchpad

Exercice-1 *Patch (rustine)*

On y apprend à

1. créer les conditions de travail dans l'environnement utilisateur : *QUILT_PATCHES*,
2. créer un **patch avec *quilt*** et ajouter les fichiers bogués au patch,
3. modifier les fichiers bogués et rafraîchir le patch
4. contrôler et modifier le fichier debian/changelog
5. créer les nouveaux paquets
6. mettre les modifications à jour avec *dpkg-source --commit*, et créer les fichiers de transfert sécurisés,
7. préparer les fichiers nécessaires à un dépôt sur Launchpad,
8. modifier une signature de clé erronée sans changer le *checksum* du fichier d'origine,
9. renvoyer le fichier *-source.changes* et contrôler les applications déposées

Exercice-2 *Mettre à jour*

On y apprend à

1. à créer une **nouvelle version *origine*** (Sans le paquet *doc*. La version remplacera la précédente sur le dépôt)
    1. Ajouter des fichiers dans le répertoire source (Langues et README-INSTALL)
    2. (Makefile) Recréer les conditions de compilation adaptées à ces nouveaux fichiers, sans passer par un patch.
    3. Adapter les fichiers du répertoire debian pour une version binaire simple
2. Créer les conditions de transfert comme dans les exercices précédents
3. Relancer la compilation d'une version partiellement en échec

Exercice-3 *Nouvelle version*

1. Création d'un **paquet natif**
2. Simplification du répertoire d'origine
3. Simplification du Makefile d'installation

Notes

Téléchargement des exercices

1. Exercice-0 *Intention d'empaqueter* mkd-120302-exercice-0.tar.xz (http://edeulo.free.fr/pub/Packaging-exercises/mkd-120302-exercice-0.tar.xz)
2. Exercice-1 *Patch (rustine)* mkd-120302-exercice-1.tar.xz (http://edeulo.free.fr/pub/Packaging-exercises/mkd-120302-exercice-1.tar.xz)
3. Exercice-2 *Mettre à jour* mkd-120302-exercice-2.tar.xz (http://edeulo.free.fr/pub/Packaging-exercises/mkd-120302-exercice-2.tar.xz)
4. Exercice-3 *Nouvelle version* mkd-120515-exercice-3.tar.xz (http://edeulo.free.fr/pub/Packaging-exercises/mkd-120515-exercice-3.tar.xz)

Voir aussi

- Un exemple de menu : *debian/menu* et *usr/share/applications/mkdcppw.desktop* dans l'archive mkdcppw_140201.tar.gz[5]
  → Le paquet *menu* n'existe plus dans les versions récentes d'Ubuntu. Cependant il est possible d'ajouter une application manuellement en suivant cet exemple.

Références

1. Pré-requis (http://doc.ubuntu-fr.org/tutoriel/creer_et_administrer_un_ppa_sur_launchpad#pre-requis)
2. Versions d'Ubuntu (https://fr.wikipedia.org/wiki/Liste_des_versions_d'Ubuntu)
3. Fichier *control* (http://www.delafond.org/traducmanfr/deb/man5/deb-control.5.html)
4. Sections Ubuntu (http://packages.ubuntu.com/saucy/)
5. Exemple de *menu* dans mkdcppw (https://launchpad.net/~mkdcppw/+archive/mkdcppw/+files/mkdcppw_140201.tar.gz)

Codes QR des téléchargements

> **Maintenance**
>
> Cette page est susceptible de subir des modifications occasionnelles

Make documents

Les **codes QR** *(Quick Response)* permettent aux lecteurs des pages imprimées de se connecter rapidement sur le wikilivre à l'aide d'un lecteur de codes QR *(disponible sur les smartphones)* pour:

- recompiler le livre sous une forme différente ou obtenir une version à jour,
- compiler d'autres pages.

Article wikibook
Mkd
Extracteur de documents

Compilation
La commande mkd
avec
QR des téléchargements

Compilation pour **ePub**
Exercices avec mkd,
avec les réponses aux exercices

Compilation:Extracteur de documents mkd avec Exercices et QR des téléchargements

---

		
Packs mkd sources + binaires Ubuntu (i386+amd64) **Téléchargement**	mkd Windows Beta 2010 (x86) **Téléchargement**	Packs mkdcppw sources + binaires Ubuntu (i386+amd64) **Téléchargement**

NOTES ET ANNEXES

Notes

Notes Icônes des fichiers et applications héritées

Note sur les icônes sous linux

- Lorsque vous installez mkd ou mkdcppw, les icônes des fichiers de projets d'extension *.pj* et *.prj* facilitent leur repérage visuel.
- Les extensions de fichiers *.pj* et *.prj* ne sont pas obligatoires pour les fichiers de projets mais facilitent leur reconnaissance avec l'option -j.

Note sur les applications périphériques de mkd

Les modules de fonctions sont évidemment utilisables dans d'autres applications:

- La commande **mkdcpp** a été créée pour éprouver la fonction cpp_ et pour évaluer une nouvelle ligne de commande avec les tiret moins et moins-moins. [-Codes (5*ASCII)], --[Options (nst)]

    Cette application utilise le module cpp.c

- La commande **mkdasm** a été créé dans le même esprit que l'application mkdcpp.

    Cette application utilise le module asm.c

- L'application **mkdcppw** a été créée sous linux pour évaluer la faisabilité d'une version fenêtrée avec, en fin de processus, une fenêtre d'impression au format **UTF-8**.

    Cette application utilise le module cpp.c et est composée de nouveaux modules, notamment pour tester le codage des caractères UTF-8 dans les textes.

- Une icône de l'application mkdcppw est visible dans la barre des tâches lorsque cette application est active.

ANNEXES ET LIENS

1.mkd-Manuel (fr)

    https://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)/mkd-Manuel_(fr)

2.mkddocu-Manuel (fr)

    https://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)/mkddocu-Manuel_(fr)

3.Compilations de wiki-livres

https://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)/Compilations

6.Licences

https://fr.wikibooks.org/wiki/Mkd_(Extracteur_de_documents)/Licences

Récupérée de « https://fr.wikibooks.org/w/index.php?title=Mkd_
(Extracteur_de_documents)/Version_imprimable&oldid=526490 »

Dernière modification de cette page le 4 septembre 2016, à 18:15.
Les textes sont disponibles sous licence Creative Commons attribution partage à l'identique ;
d'autres termes peuvent s'appliquer.
Voyez les termes d'utilisation pour plus de détails.