perlapio

Table des matières

1 NAME/NOM

perlapio - Interface d'abstraction des E/S internes à Perl

2 SYNOPSIS

```
PerlIO *PerlIO_stdin(void);
PerlIO *PerlIO_stdout(void);
PerlIO *PerlIO_stderr(void);

PerlIO *PerlIO_open(const char *,const char *);
int     PerlIO_close(PerlIO *);

int     PerlIO_stdoutf(const char *,...)
int     PerlIO_puts(PerlIO *,const char *);
int     PerlIO_putc(PerlIO *,int);
int     PerlIO_write(PerlIO *,const void *,size_t);
int     PerlIO_printf(PerlIO *, const char *,...);
int     PerlIO_vprintf(PerlIO *, const char *, va_list);
int     PerlIO_flush(PerlIO *);

int     PerlIO_eof(PerlIO *);
int     PerlIO_error(PerlIO *);
void    PerlIO_clearerr(PerlIO *);

int     PerlIO_getc(PerlIO *);
int     PerlIO_ungetc(PerlIO *,int);
int     PerlIO_read(PerlIO *,void *,size_t);

int     PerlIO_fileno(PerlIO *);
PerlIO *PerlIO_fdopen(int, const char *);
PerlIO *PerlIO_importFILE(FILE *, int flags);
FILE   *PerlIO_exportFILE(PerlIO *, int flags);
FILE   *PerlIO_findFILE(PerlIO *);
void    PerlIO_releaseFILE(PerlIO *,FILE *);

void    PerlIO_setlinebuf(PerlIO *);
```

```
long    PerlIO_tell(PerlIO *);
int     PerlIO_seek(PerlIO *,off_t,int);
int     PerlIO_getpos(PerlIO *,Fpos_t *)
int     PerlIO_setpos(PerlIO *,Fpos_t *)
void    PerlIO_rewind(PerlIO *);

int     PerlIO_has_base(PerlIO *);
int     PerlIO_has_cntptr(PerlIO *);
int     PerlIO_fast_gets(PerlIO *);
int     PerlIO_canset_cnt(PerlIO *);

char   *PerlIO_get_ptr(PerlIO *);
int     PerlIO_get_cnt(PerlIO *);
void    PerlIO_set_cnt(PerlIO *,int);
void    PerlIO_set_ptrcnt(PerlIO *,char *,int);
char   *PerlIO_get_base(PerlIO *);
int     PerlIO_get_bufsiz(PerlIO *);
```

3 DESCRIPTION

Les sources de Perl doivent utiliser les fonctions listées ci-dessus à la place de celles définies dans l'en-tête *stdio.h* de l'ANSI C. Les en-têtes de perl les remplacent par le mécanisme d'entrée-sortie selectionné lors de l'exécution de Configure en utilisant des `#define`.

Ces fonctions sont calquées sur celles de *sdio.h*, mais pour certaines, l'ordre des paramètres a été réorganisé.

PerlIO *

Remplace FILE *. Comme FILE * , il doit être traité comme un type opaque (c'est probablement prudent de le considérer comme un pointeur sur quelque chose).

PerlIO_stdin(), PerlIO_stdout(), PerlIO_stderr()

À utiliser à la place de `stdin`, `stdout`, `stderr`. Elles ressemblent à des « appels de fonctions » plutôt qu'à des variables pour que ce soit plus facile d'en faire des appels de fonctions si la plate-forme ne peut exporter des données vers des modules chargés, ou si (par exemple) des « threads » différents peuvent avoir des valeurs différentes.

PerlIO_open(path, mode), PerlIO_fdopen(fd,mode)

Correspondent à fopen()/fdopen() et utilise les mêmes arguments.

PerlIO_printf(f,fmt,...), PerlIO_vprintf(f,fmt,a)

Équivalent à fprintf()/vfprintf() .

PerlIO_stdoutf(fmt,...)

Équivalent à printf(). printf est #defined à cette fonction, donc il est (pour le moment) légal d'utiliser `printf(fmt,...)` dans les sources perl.

PerlIO_read(f,buf,compteur), PerlIO_write(f,buf,compteur)

Correspondent à fread () et fwrite (). Attention, les arguments sont différents, Il y a seulement un « compteur » et le fichier f est en premier.

PerlIO_close(f)

PerlIO_puts(f,s), PerlIO_putc(f,c)

Correspondent à fputs() et fputc(). Attention, les arguments ont été réorganisés pour que le fichier soit en premier.

PerlIO_ungetc(f,c)

Correspond à ungetc(). Attention, les arguments ont été réorganisé pour que le fichier soit en premier.

PerlIO_getc(f)

Correspond à getc().

PerlIO_eof(f)

Correspond à feof().

PerlIO_error(f)

Correspond à ferror().

PerlIO_fileno(f)

Correspond à fileno(). Attention, sur certaines plates-formes, la définition de « fileno » peut ne pas correspondre à celle d'Unix.

PerlIO_clearerr(f)

Correspond à clearerr(), c.-à-d., retire les drapeaux « eof » et « error » pour le « flux ».

PerlIO_flush(f)

Correspond à fflush().

PerlIO_tell(f)

Correspond à ftell().

PerlIO_seek(f,o,w)

Correspond à fseek().

PerlIO_getpos(f,p), PerlIO_setpos(f,p)

Correspondent à fgetpos() et fsetpos(). Si la plate-forme ne dispose pas de ces fonctions, elles sont implémentées à l'aide de PerlIO_tell() et PerlIO_seek().

PerlIO_rewind(f)

Correspond à rewind(). Note : elle peut être définie à l'aide de PerlIO_seek().

PerlIO_tmpfile()

Correspond à tmpfile(), c.-à-d., retourne un PerlIO anonyme qui sera automatiquement effacé lors de sa fermeture.

3.1 Co-existence avec stdio

Il existe un support de la coexistence de PerlIO avec stdio. Évidemment, si PerlIO est implémenté à l'aide de stdio, il n'y aucun problème. Mais si perlio est implémenté au-dessus de sfio (par exemple) il doit exister des mécanismes permettant de créer un FILE * qui peut être passé aux fonctions de bibliothèques qui utilisent stdio.

PerlIO_importFILE(f,flags)

Permet d'obtenir un PerlIO * à partir d'un FILE *. Peut nécessiter plus d'arguments, interface en cours de réécriture.

PerlIO_exportFILE(f,flags)

À partir d'un PerlIO * renvoie un FILE * pouvant être donné à du code devant être compilé et linké avec *stdio.h* de l'ANSI C.

Le fait qu'un FILE * a été « exporté » est enregistré, et peut affecter de futures opérations sur le PerlIO * original. PerlIO *.

PerlIO_findFILE(f)

Retourne un FILE * prédément « exporté » (s'il existe). Ceci est un bouche-trou jusqu'à ce que l'interface soit complètement définie.

PerlIO_releaseFILE(p,f)

L'appel à PerlIO_releaseFILE informe PerlIO que le FILE * ne sera plus utilisé. Il est alors retiré de la liste des FILE * « exporté », et le PerlIO * associé retrouve son comportement normal.

PerlIO_setlinebuf(f)

Correspond à setlinebuf(). Son utilisation est dépréciée en attendant une décision sur son sort. (Le noyau de Perl l'utilise *uniquement* lors du dump ; cela n'a rien à voir avec le vidage automatique $|.)

En plus de l'API utilisateur décrite précédement, il existe une interface d'« implémentation » qui permet à Perl d'accèder aux structures internes de PerlIO. Les appels suivants correspondent aux diverses macros FILE_xxx déterminées par Configure. Cette section n'a d'intérêt que pour ceux qui sont concerné par un descriptif détaillé de comportement du noyau perl ou qui implémentent un MAPPING PerlIO.

PerlIO_has_cntptr(f)

L'implémentation peut retourner un pointeur vers la position courante dans le « buffer » et le nombre d'octets disponibles dans le buffer.

PerlIO_get_ptr(f)

Retourne un pointeur vers le prochain octet du buffer à lire.

PerlIO_get_cnt(f)

Retourne le nombre d'octets du buffer restant à lire.

PerlIO_canset_cnt(f)
L'implémentation peut ajuster son idée sur le nombre d'octets dans le buffer.

PerlIO_fast_gets(f)
L'implémentation a toutes les interfaces nécessaire pour permetre au code rapide de perl d'utiliser le mécanisme <FILE> .

```
PerlIO_fast_gets(f) = PerlIO_has_cntptr(f) && \
                      PerlIO_canset_cnt(f) && \
                      'Can set pointer into buffer'
```

PerlIO_set_ptrcnt(f,p,c)
Place le pointeur dans le buffer et remplace le nombre d'octets encore dans le buffer. Doit être utilisé seulement pour positionner le pointeur à l'intérieur de la plage impliquée par un appel précédent à `PerlIO_get_ptr` et `PerlIO_get_cnt`.

PerlIO_set_cnt(f,c)
Obscure - force le nombre d'octets dans le buffer. Déprécié. Utilisé uniquement dans doio.c pour forcer un compteur <-1 à -1. Deviendra peut-être PerlIO_set_empty ou similaire. Cet appel peut éventuellement ne rien faire si « compteur » est déduit d'un pointeur et d'une « limite ».

PerlIO_has_base(f)
L'implémentation a un buffer, et peut retourner un pointeur vers celui-ci ainsi que sa taille. Utilisé par perl pour les tests **-T / -B**. Les autres usages seraient très obscurs...

PerlIO_get_base(f)
Retourne le *début* du buffer.

PerlIO_get_bufsiz(f)
Retourne la *taille totale* du buffer.

4 TRADUCTION

4.1 Version

Cette traduction française correspond à la version anglaise distribuée avec perl 5.005_02. Pour en savoir plus concernant ces traductions, consultez http://perl.enstimac.fr/.

4.2 Traducteurs

Marc Carmier <carmier@immortels.frmug.org>

4.3 Relecture

Gérard Delafond.

5 À propos de ce document

Ce document est la traduction française du document orginal distribué avec perl. Vous pouvez retrouver l'ensemble de la documentation française Perl (éventuellement mise à jour) en consultant l'URL <http://perl.enstimac.fr/>.

Ce document PDF a été produit Paul Gaborit. Si vous utilisez la version PDF de cette documentation (ou une version papier issue de la version PDF) pour tout autre usage qu'un usage personnel, je vous serai reconnaissant de m'en informer par un petit message <mailto:Paul.Gaborit@enstimac.fr>.

Si vous avez des remarques concernant ce document, en premier lieu, contactez la traducteur (vous devriez trouver son adresse électronique dans la rubrique TRADUCTION) et expliquez-lui gentiment vos remarques ou critiques. Il devrait normalement vous répondre et prendre en compte votre avis. En l'absence de réponse, vous pouvez éventuellement me contacter.

Vous pouvez aussi participer à l'effort de traduction de la documentation Perl. Toutes les bonnes volontés sont les bienvenues. Vous devriez trouver tous les renseignements nécessaires en consultant l'URL ci-dessus.