Extension de Perl
les modules

LPRO ISVD
Olivier FLAUZAC
olivier.flauzac@univ-reims.fr

Modules Perl

Extension de perl

- Utilisation de modules
- Banque de modules : CPAN
- Installation de modules additionnels
- Extension des types
- Extension des protocoles

Modules

- Fichier .pm
- Peut être dépendant du système
  - modules spécialisés
  - dépend des bibliothèques systèmes
- Architecture hiérarchique

Hiérarchie des modules

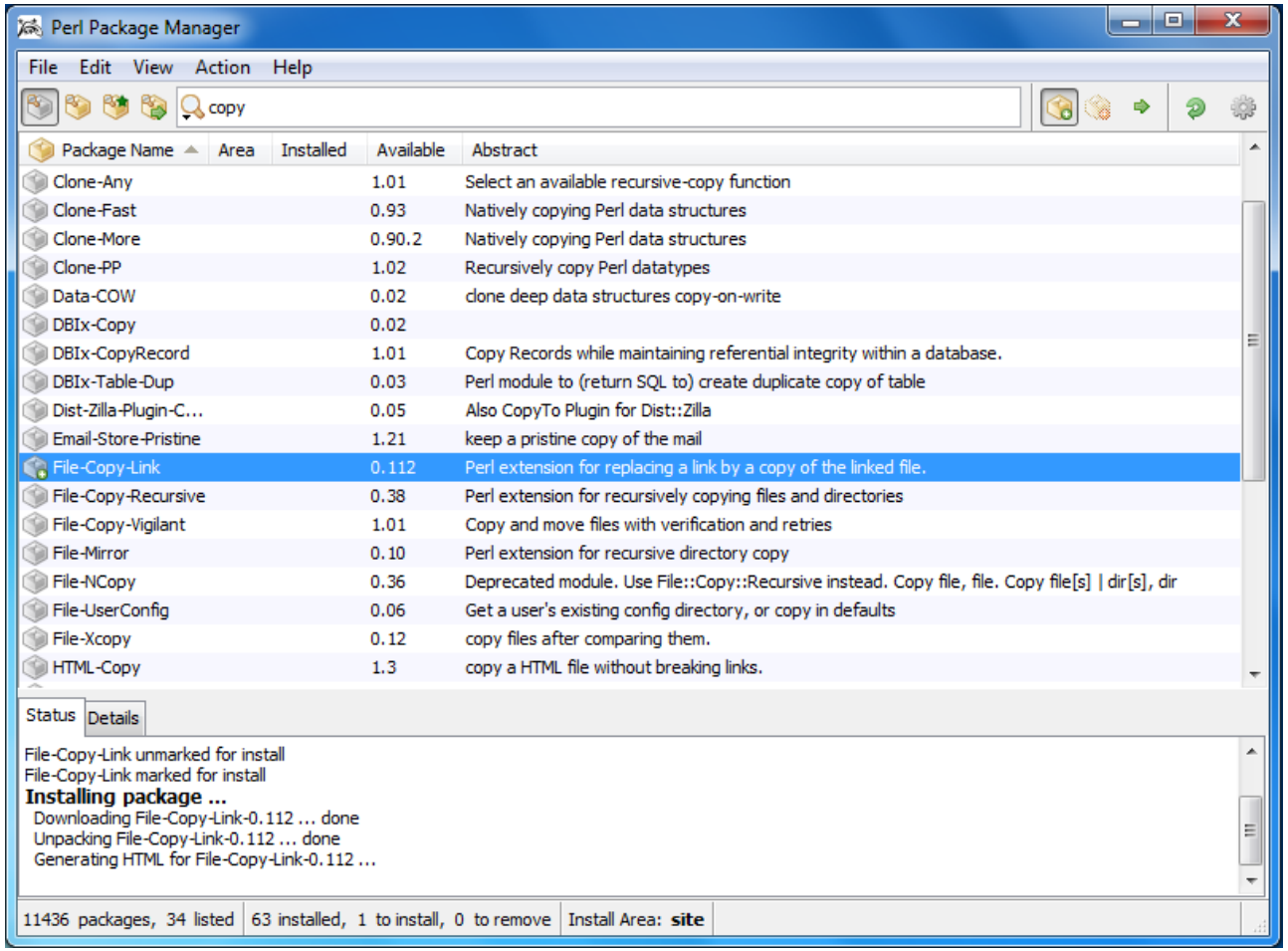
- Archive
  - Archive::tar
  - Archive::Zip
  - Archive::Rar
- Net
  - Net::arp
  - Net::bonjour
  - Net::SSH
  - Net::ip
  - Net::http
- File
  - File::copy
  - File::find
  - File::homedir
- WWW
  - WWW::Curl
- Apache2
  - Apache2::HTTP
  - Apache2::REST
- ...

Installation d'un module perl

- Dépendant de la plate-forme
- Gestion des dépendances
- Applications graphiques
- Ligne de commande
  `cpan -i module`
- Installation utilisateur ou admin
- Installation package distribution
  - libxxx-yyy-zzz-perl (Debian)

Installation d'un module perl (WIN)

- Perl package Manager



Utilisation d'un module

- Inclusion (use)
- Définition de la liste des fonctions utilisées
- Utilisation
  - procédurale
  - objet

Copie de fichier

```
use File::Copy::Recursive qw(fcopy fmove);

fcopy("./Images/i4.jpg","./image.jpg") or die $!;
if(-e "./image.jpg"){
  print "le fichier existe \n";
}else{
  exit 1;
}
fmove("./image.jpg","./toto.jpg") or die $!;

exit 0;
```

Un peu de programmation objet

Programmation objet

- Programmation centrée sur les types et les données
- Utilisation de classes
- Création d'objets
- Application de méthodes sur ces objets

Perl et les objets

- Nombreux modules objets en Perl
- Définition de classes dans les modules
- Modules
  - pur objets
  - mixtes
  - procéduraux
- Exploitation de ->

Construction d'un objet

- Utilisation de la fonction new
  - exploitation de paramètres

```
my $ftp = Net::FTP->new("192.168.1.17")
          or die "Connexion impossible";
```

Utilisation d'une méthode

- Exploitation de ->

```
$ftp->login("log","pass")
          or die "Cannot login ", $ftp->message;

$ftp->touch("toto.txt");
$ftp->quit;
```

```
use Net::FTP::File;

my $ftp = Net::FTP->new("192.168.1.17")
      or die "Connexion impossible";

$ftp->login("olivier","mitsou00")
      or die "Cannot login ", $ftp->message;

$ftp->touch("toto.txt");
$ftp->quit;
```

Exemple de module CGI

CGI.PM

- Création de scripts côté serveur
- Gestion des requêtes
- Construction des réponses
- Gestion des éléments HTTP et HTML
  - Cookie
  - Formulaires ...
- Génération d'éléments HTML
  - images
  - liens ...

Mise en oeuvre

- Utilisation d'un serveur WEB
  - WAMP
  - configuration pour l'exec de CGI
    - activation du module CGI
    - directive `+ExecCGI`
- Installation de CGI.pm
- Utilisation
  - objet
  - procédural

Exemple

```
#!C:\Program Files\VMware\VMware vSphere CLI\Perl\bin\perl.exe

use CGI;

$cgi=new CGI;

print $cgi->header;
print $cgi->start_html("hello world !");
print $cgi->h1("hello world !");
print $cgi->p("salut les amis");
print $cgi->end_html;
```

Exemple

```
#!C:\Program Files\VMware\VMware vSphere CLI\Perl\bin\perl.exe
use CGI ':standard';

print header;
print start_html('hello world !');
print h1('hello world !');
print p("salut les amis");
print end_html;
```

Code généré

```
<!DOCTYPE html
   PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US">
<head>
<title>hello world !</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body>
<h1>hello world !</h1><p>salut les amis</p>
</body>
</html>
```

Génération d'éléments HTML

```
#!C:\Program Files\VMware\VMware vSphere CLI\Perl\bin\perl.exe
use CGI ':standard';

print header;
print start_html('Eléments HTML');

print h1('Insertion d\'images');
print img {  src=>'i4.jpg',
          width=>"200px"};



print h1('Insertion de lien');
print a(
    {href => 'http://search.cpan.org/dist/CGI.pm-3.45/lib/CGI.pm'},
    "Vers la documentation de CGI.pm");

print h1('Des listes');
print ul(li("un"),li("deux"),li("trois"));
print ol(li("un"),li("deux"),li("trois"));
```

Génération d'éléments HTML (suite)

```
print h1('Un tableau');
print table({border=>"1"},
          Tr(
              [
              td("1","2","3"),
              td("4","5","6")
              ]
          )
          );
print end_html;
```

GESTION DES FORMULAIRES

- 2 acteurs
  - le formulaire
  - le script de traitement
- 2 méthodes
  - POST
  - GET
- Récupération des valeurs entrées
  - fonction `param`

Traitement de formulaire

```
<html>
   <head></head>
   <body>
      <form name="" method="POST" action="trait.cgi">
         <input type="text" name="txt" />
         <select name="slt">
            <option value="1"> option 1 </option>
            <option value="2"> option 2 </option>
            <option value="3"> option 3 </option>
         </select>
         <input type="submit" value="OK" />
      </form>
   </body>
</html>
```

Traitement de formulaire

```
#!C:\Program Files\VMware\VMware vSphere CLI\Perl\bin\perl.exe
use CGI ':standard';

print header;
print start_html('Formulaires');
$texte = param("txt");
print p("la valeur texte entrée est $texte");

$val = param("slt");
print p("la valeur du select est $val");

print end_html;
```

VMWARE

API

- Une API de gestion des plate-formes
- Intégré à une distribution de Perl
- Utilisation objet
- Des méthodes de base
- Des fonctions utilisables

Exploitation

- Développement depuis les objets élémentaires
- Appel à la banque de fonctions
- Chemin d'installation

```
C:\Program Files\VMware\VMware vSphere CLI
```

Structure de l'installation

Répertoire de base

```
C:\Program Files\VMware\VMware vSphere CLI\Perl\apps
```

Structure de l'installation

- Répertoire de base
  `C:\Program Files\VMware\VMware vSphere CLI\Perl\apps`
- Sous répertoires
  - `AppUtil`
  - `general`
  - `host`
  - `performance`
  - `session`
  - `vm`

GENERAL

- connect.pl
  - script de connexion à une machine
  - display_servertime

PERFORMANCE

- viperformance.pl
  - contrôle d’état

VM

- vmcontrol.pl
  - contrôle des machine (on , off ...)
- vmsnapshot.pl
- vmsnapshtmanager.pl
  - snapshot et restore des machines
- vmmigrate.pl
  - migration des machines

Exploitation

- Appels aux scripts en exécution
- Création de scripts originaux
- Appel aux fonctions

Un script original

```
use VMware::VIRuntime;
use AppUtil::VMUtil;

Opts::set_option("server","addr");
Opts::set_option("username","usr");
Opts::set_option("password","pass");

print "Données enregistrées \n";

Util::connect();

print "Connecté \n";

my $host = Vim::find_entity_views(view_type => "VirtualMachine");
```

Un script original

```
foreach my $vm (@$host){
    $vm->update_view_data();
    print "*****************************\n";
    print "le nom de la VM : " .
          $vm->name . "\n";
    print "l'état de la VM : " .
          $vm->runtime->powerState->val ."\n";
    if($vm->runtime->powerState->val eq "poweredOn"){
       print "boottime : ". $vm->runtime->bootTime . "\n";
       print "nombre de processeurs : ".
             $vm->runtime->maxCpuUsage . "\n";
       print "mémoire : ". $vm->runtime->maxMemoryUsage . "\n";
    }
}
```

Un script original

```
my $machine = Vim::find_entity_view(
              view_type => "VirtualMachine",
              filter => {name => "WinXP-3"});

print "le nom de la VM : " .
              $machine->name . "\n";
print "l'état de la VM : " .
              $machine->runtime->powerState->val ."\n";
$machine->PowerOnVM();
```

Exploitation de scripts Externes

```
$addr = "addr";
$usr = "user";
$pass = "password";
$cmd = "connect.pl";

$res = `perl $cmd --server $addr --username $usr --password $pass`;
print $res;

exit 0;
```