Perl/Ruby + MySQL による 動的 Web ページ作成

VA Linux Systems ジャパン(株) OSDN 事業部
Debian Project
安井 卓 <tach@valinux.co.jp>, <tach@debian.org>

Agenda

- 動的 Web ページのしくみ
  - データベースの必要性
- MySQL について
- Perl DBI
  - 使い方・例
- Ruby-MySQL
  - 使い方・例
- さらにいろいろ
  - mod_perl / mod_ruby
  - eperl / eruby
- 参考文献

動的 Web ページのしくみ

- サーバサイド
  - CGI
  - サーバ組み込み型(PHP/mod_perl/mod_ruby)
  - JavaServlet
- クライアントサイド
  - Javascript
  - JavaApplet

サーバサイド 動的 Web

- 長所
  - サーバ側で処理をするので，クライアントに依存しない
  - 一貫してデータなどを管理できる
- 短所
  - 派手な見せ方はできない
  - 接続が多いと，サーバに負荷がかかる

クライアントサイド 動的 Web

- 長所
  - クライアントが処理するので，サーバの負担が少ない
  - クライアントの機能をふんだんに使って派手なことができる
- 短所
  - データの管理などができない
  - クライアントによっては利用できない場合がある

バックエンド DB の必要性

最近はいろんなことを Web でするようになってきた

- アクセス数の増大
- 処理データの増大･複雑化

テキストファイルや簡易 DB では対応しきれない場合がある

- 高速化
- データの安全性
- 大量のデータを処理

でも，データベースが不要な場合も多い

- 個人サイト
- さまざまな環境で動作するように設計されたもの

MySQL について

一次配布元: http://www.mysql.com/

SQL 言語が利用できるデータベースエンジン

- 現在のバージョン
  - 3.23.x
  - 4.0 が alpha release された
- ライセンス
  - GPL と独自ライセンス (MySQL Free Public License, FPL) のデュアルライセンス

MySQL の特徴 - 高速動作

- マルチスレッド
- 高速で動作することを前提にコーディング
- 動作が遅くなるコードは入れない
- ベンチマーク
  http://www.mysql.com/information/benchmarks.html
  - 単純な文では Postgres の数倍 ～ 100 倍以上

そのために，他の DB エンジンにあるような機能がなかったりする

MySQL の特徴 - 対応プラットフォーム

- Linux, FreeBSD, NetBSD などの Free PC-UNIX
- Solaris (SPARC/x86)
- HP-UX
- MacOS X
- IRIX
- その他 多くの UNIX 互換システム
- OS/2 Warp
- Windows 95/98/NT/2000

MySQL の特徴 - 対応言語

- C 言語 (libmysql)
- Perl (DBI, MySQL driver)
- Ruby
- PHP
- Python
- Java (JDBC)
- ODBC

など

MySQL の特徴 - オープンソース

- 最新版(3.23.19 以降)は GPL で配布
- 自由に改変・再配布が可能
- 変更点は公開しなければならない
- ライブラリをリンクする場合，そのソフトは GPL にしなければならない

- closed な利用をするなら，MySQL のライセンスを購入 (FPL)
  - MySQL の改変 → ソース非公開
  - ソース非公開の組み込み利用

MySQL に適した場所は?

- 単純な問い合わせが多い
- とにかく高速で動作することが必要
- データの更新が頻繁ではない
- 多くの機能を必要としない

Web は上記の条件をみたしており
MySQL に最適な環境

世界中の多くのサイトのバックエンド DB として利用されている

MySQL の日本語対応

パッチ等は必要なく，デフォルトで EUC-JP Shift_JIS UTF-8 に対応
ただし，コンパイルオプションに注意

- configure 時に --with-charset=xxx で指定
- configure 時に --with-extra-charsets=all を指定し，実行時に --default-charactor-set=xxx オプションで指定

多くの関数が日本語に対応しているが，日本語に対応していない関数もある

MySQL のインストール

- ソースからのインストール
  - 省略… ^^;
- RPM パッケージ
  - http://www.mysql.com/ に RPM, SRPM が存在する
  - 万全を期すために SRPM をダウンロードして rpm --rebuild MySQL-*.src.rpm で作り直してインストール
- deb パッケージ
  - apt-get install mysql-server
  - potato(Debian 2.2) の MySQL は 3.22.32 なので，ライセンス (FPL) に注意
  - 必要なら unstable から最新のソースを持ってきてパッケージを作り直す

MySQL のコマンド

- **mysql**
  - MySQL サーバに接続して SQL で操作するクライアント
  - SQL を手で打ち込んでデータベースを操作するときにつかう
- **mysqladmin**
  - MySQL サーバの管理を容易にするためのコマンド
- **mysqldump**
  - データベースの内容を SQL 文でファイルにダンプする

主要な SQL 文

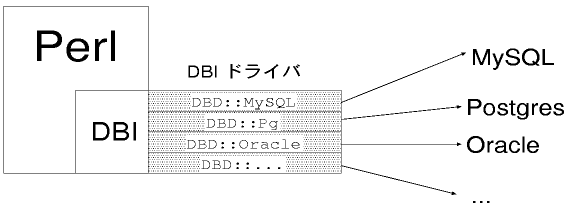
- **CREATE DATABASE / DROP DATABASE**
  - データベースの作成･消去
- **CREATE TABLE / DROP TABLE**
  - テーブルの作成･消去
- **ALTER TABLE**
  - テーブルの構造の変更
- **SELECT**
  - レコードの検索
- **INSERT / UPDATE**
  - レコードの挿入･更新
- **DELETE**
  - レコードの削除
- **GRANT / REVOKE**
  - 権限の付与･取り消し

Perl DBI とは

データベースエンジンを Perl で利用するための共通インターフェース
Database independent interface for Perl

- http://dbi.perl.org/
- 最新版は 1.20
- DBIドライバにより各種 DB エンジンに対応
  - MySQL（最新版は 1.2216）
  - PostgreSQL
  - Oracle
  - ...

Perl DBI のしくみ



- DBI が各バックエンドの差を吸収し，インターフェースを統一
- DBD::Foobar と呼ばれる DBIドライバを用意

Perl DBI のインストール

- ソースからのインストール
  - やっぱり省略 ^^;
- RPM パッケージ
  - ソースから作るのが安全? ^^;
  - cpanflute を使おう
    - mkdir -p /tmp/cpan/junk /tmp/cpan/temp
    - /usr/lib/rpm/cpanflute DBI-1.20.tar.gz
    - rpm --rebuild /tmp/cpan/temp/perl-DBI-1.20-6.src.rpm
    - /usr/lib/rpm/cpanflute Msql-Mysql-modules-1.2216.tar.gz
    - rpm --rebuild /tmp/cpan/temp/perl-Msql-Mysql-modules-1.2216-6.src.rpm
- deb パッケージ
  - apt-get install libdbi-perl libdbd-mysql-perl (楽ちん)

Perl DBI の使用例 (1)

```
use DBI; # DBI モジュールをロード(必須)

# データベースに接続
$dbh = DBI->connect('DBI:mysql:database_name:server_name',
                    'user_name', 'password');
```

- DBI->connect - データベースに接続
  - server_name : MySQL サーバのホスト名
  - user_name : MySQL サーバに接続するユーザ
  - password : 接続するときのパスワード

$dbh がデータベースハンドルオブジェクトになる.
データベースにアクセスするときに使う.

Perl DBI の使用例 (2)

```
# SQL 構文を準備
 $sth = $dbh->prepare("SELECT foo,bar FROM table WHERE hoge='areg
e'");

# SQL を実行
$sth->execute;
```

- $dbh->prepare - あとで実行するための単一文を準備
  - ステートメントハンドルオブジェクトを生成
- $sth->execute - SQL 文を実行

Perl DBI の使用例 (3)

```
# 結果を処理
while ( @row = $sth->fetchrow_array ) {
    print join(',', @row) . "\n";
}

$sth->finish; # 後処理

$dbh->disconnect; # 切断
```

- $sth->fetchrow_array - 結果の各行を配列で返す
  - 実行するごとに次の行へとすすみ，最後まで行くと undef を返す
- $sth->finish
  - メモリの解放
- $dbh->disconnect;
  - 接続を切る

Perl DBI よく使うメソッド

- DBI
  - $dbh = DBI->connevt - データベースに接続
- $dbh(データベースハンドル)
  - $sth = $dbh->prepare - SQL 文を準備
  - $dbh->selectrow_array - prepare, execute, fetchrow_array を同時に行う
  - $dbh->selectall_arrayref - prepare, execute, fetchall_arrayref を同時に行う
  - $dbh->do - 指定された SQL 文を実行
  - $dbh->disconnect - データベースから切断
- $sth(ステートメントハンドル)
  - $sth->execute - 準備された SQL 文を実行
  - $sth->fetchrow_array - 結果を一行ずつ配列で返す
  - $sth->fetchrow_arrayref - 結果を一行ずつ配列参照で返す
  - $sth->fetchrow_hashref - 結果を一行ずつハッシュ参照で返す
  - $sth->finish - メモリを解放する

DBI を使った CGI の実例 (1)

ユーザ名 とパスワードの入力を促し，合ってるかどうかを確かめる.

使用するテーブル

```
create table passwd (
    user char(8),
    pass char(8)
);
```

- データベースとユーザを作成
  - mysqladmin create login
  - echo 'GRANT ALL PRIVILEGES ON login.* TO foo@localhost IDENTIFIED BY "bar";' | mysql -u root

DBI を使った CGI の実例 (2)

```
#!/usr/bin/perl

use DBI; # DBI モジュール
use CGI qw(:standard -no_xhtml); # CGI モジュール

my $cgi = CGI->new();
my $user = $cgi->param('user');
my $pass = $cgi->param('pass');

print $cgi->header;

if ( ! $user || ! $pass ) {
  print $cgi->start_html('パスワードチェック'),
        $cgi->start_form,
        "ユーザ:", $cgi->textfield('user','',8,8),
        "パスワード:", $cgi->password_field('pass','',8,8),
        $cgi->submit,
        $cgi->end_form;
  exit;
}
```

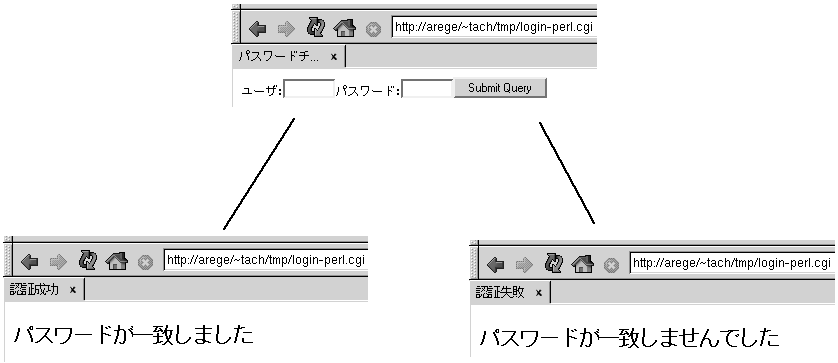
DBI を使った CGI の実例 (3)

```
my $dbh = DBI->connect('DBI:mysql:login:localhost',
                'foo', 'bar');

my $db_pass = $dbh->selectrow_array("SELECT pass FROM passwd WHERE user='$user'");

if ( $db_pass && $pass eq $db_pass ) {
    print $cgi->start_html('認証成功'),
          $cgi->h1("パスワードが一致しました")
} else {
    print $cgi->start_html('認証失敗'),
          $cgi->h1("パスワードが一致しませんでした")
}
```

実行結果



MySQL Ruby Module

- MySQL の Ruby インターフェース
- http://www.tmtm.org/ja/mysql/ruby/
- 最新版は version 2.4

- クラス
  - Mysql - 基本クラス，サーバへの接続など
  - MysqlRes - クエリの結果
  - MysqlField - テーブルのフィールド
  - MysqlError - エラー

MySQL-Ruby のインストール

- ソースからインストール
  - tar xvzf mysql-ruby-x.y.z.tar.gz
  - cd mysql-ruby-x.y.z
  - ruby extconf.rb
  - make; make install
- rpm パッケージ
  - 自作しないといけない?
- deb パッケージ
  - apt-get install libmysql-ruby（やっぱり楽ちん）

MySQL-Ruby の使用例 (1)

```
require 'mysql' # mysql モジュールをロード

# データベースに接続
my = Mysql::new('server_name', 'user_name', 'password',
                database_name)
```

- Mysql::new - データベースに接続
  - server_name : MySQL サーバのホスト名
  - user_name : MySQL サーバに接続するユーザ
  - password : 接続するときのパスワード
  - database_name : データベース名
  - Mysql クラスのインスタンス (my) を返す
  - データベース名は，あとから選択することもできる

MySQL-Ruby の使用例 (2)

```
# クエリの実行
res = my.query("SELECT foo,bar FROM table WHERE hoge='arege'")

# 結果を処理
res.each do |foo, bar|
    puts "#{foo},#{bar}"
end

# 閉じる
my.colse
```

- Mysql#query - クエリの実行
  - MysqlRes クラスのインスタンス (res) を返す
- MysqlRes#each - 各行ごとにループするイテレータ
- Mysql#close - 接続を閉じる

MySQL-Ruby よく使うメソッド

- Mysql クラス
  - Mysql::new - インスタンス生成，データベース接続
  - Mysql#query - クエリの実行
  - Mysql#close - 接続を閉じる
- MysqlRes クラス
  - MysqlRes#each - 各レコードごとのイテレータ(配列)
  - MysqlRes#each_hash - 各レコードごとのイテレータ(ハッシュ)
  - MysqlRes#num_rows - 総レコード数

Ruby-MySQL を使った CGI の実例 (1)

```
#!/usr/bin/ruby

require 'mysql' # MySQL モジュール
require 'cgi' # CGI モジュール

cgi = CGI::new("html4Tr")
user = cgi['user'][0]
pass = cgi['pass'][0]

if ( ! user || ! pass )
  cgi.out( { 'charset' => 'euc-jp'} ) do
    cgi.html() do
      cgi.head { cgi.title{'パスワードチェック'} } +
      cgi.body() do
        cgi.form() do
          "ユーザ:" + cgi.text_field('user', '', 8, 8) +
          "パスワード:" + cgi.password_field('pass', '', 8, 8) +
          cgi.submit
        end
      end
    end
  end
  exit
end
```

Ruby-Mysql を使った CGI の実例 (2)

```
my = Mysql::new('localhost','foo', 'bar',
                'login');

db_pass = my.query("SELECT pass FROM passwd WHERE user='#{user}'").fetch_row[0]

if ( db_pass && pass == db_pass )
    cgi.out( { 'charset' => 'euc-jp'} ) do
        cgi.html() do
            cgi.head { cgi.title{'認証成功'} } +
            cgi.body() do
                cgi.h1{'パスワードが一致しました'}
            end
        end
    end
else
    cgi.out( { 'charset' => 'euc-jp'} ) do
        cgi.html() do
            cgi.head { cgi.title{'認証失敗'} } +
            cgi.body() do
                cgi.h1{'パスワードが一致しませんでした'}
            end
        end
    end
end
```

ユーザ ID 作成機能追加 (Perl)

- フォーム作成部分の変更
  - ユーザ作成ボタンを作る

```
print $cgi->start_html('パスワードチェック'),
    $cgi->start_form,
    "ユーザ:", $cgi->textfield('user','',8,8),
    "パスワード:", $cgi->password_field('pass','',8,8),
    $cgi->submit({ name => 'opLogin', value => 'ログイン' }),
    $cgi->submit({ name => 'opCreate', value => 'ユーザ ID 作成' }),
    $cgi->end_form;
```

- ボタンの name 属性でどちらの命令かを判断

ユーザ ID 作成機能追加 (Perl)

submit ボタンの name 属性で判定して実行

```
if ( $cgi->param('opLogin') ) {
   my $db_pass = $dbh->selectrow_array("SELECT pass FROM passwd WHERE user='$user'");
   if ( $db_pass && $pass eq $db_pass ) {
      print $cgi->start_html('認証成功'),
      $cgi->h1("パスワードが一致しました");
   } else {
      print $cgi->start_html('認証失敗'),
      $cgi->h1("パスワードが一致しませんでした");
   }
} elsif ( $cgi->param('opCreate') ) {
   if ( $dbh->selectrow_array("SELECT user FROM passwd WHERE user='$user'") ) {
      print $cgi->start_html("すでに $user が存在します"),
      $cgi->h1("すでに $user が惣菜します");
   } else {
      if ( $dbh->do("INSERT INTO passwd VALUES ('$user','$pass')") ) {
         print $cgi->start_html('ユーザ作成完了'),
      } else {
         print $cgi->start_html('ユーザ作成'),
         $cgi->h1("ユーザ $user の作成に失敗しました");
      }
   }
```

ユーザ ID 作成機能追加 (Perl)

- どっちの命令もない場合は，フォームを生成

```
} else {
    print $cgi->start_html('パスワードチェック'),
    $cgi->start_form,
    "ユーザ:", $cgi->textfield('user','',8,8),
    "パスワード:", $cgi->password_field('pass','',8,8),
    $cgi->submit({ name => 'opLogin', value => 'ログイン' }),
    $cgi->submit({ name => 'opCreate', value => 'ID 作成'}),
    $cgi->end_form;
}

$cgi->end_html;
```

さらに速く動かすために (1)

- テーブルのインデックスを作る
  - 検索時の索引のようなもの
  - CREATE TABLE 時に INDEXで指定
    - CREATE TABLE foo (bar CHAR(12) NOT NULL, INDEX key1 (bar));
  - EXPLAIN を使って SELECT の効率を調べる
    - EXPLAIN SELECT ...
- MySQL のチューニング
  - 変数で設定可能
    - コマンドラインオプション or 設定ファイル my.cnf
  - 少ないメモリでも快適に動作するようになっている
  - メモリをたくさん持っている場合は，有効に活用するように修正
  - 例 (my.cnf):
    - set-variable = key_buffer_size=128M
    - set-variable = sort_buffer=4M

さらに速く動かすために (2)

CGI は実行ごとにプロセスを生成するので遅い

- プロセス生成は結構重い処理
- 頻繁にアクセスのあるページを CGI で書くのは自殺行為

プロセスを生成しない → デーモン化?

- httpd がデーモンで動いている
- httpd (apache) に組み込んじゃえ!

↓

mod_perl mod_ruby
php

mod_perl と mod_ruby

http://perl.apache.org/ mod_perl
http://www.modruby.net/ mod_ruby

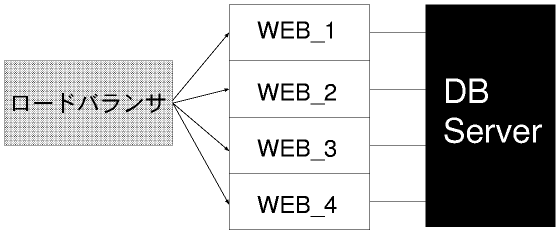
- Apache のモジュール
- Apache のプロセスとして動作するので高速
  - 10 倍以上のスピード?
- CGI と同様のインターフェースを備えているため，CGI からの移行が楽

- 逆にそれが命取りにもなる
  - プログラムが変だと Apache まで影響を受ける

さらに速く動かすために (3)

Web クラスタリング

- ロードバランサ(負荷分散装置)を前段に置く
- ユーザはロードバランサに設定されたの疑似 IP にアクセス
- ロードバランサが複数のサーバに分散アクセス



負荷分散

データは DB で持っているので，共通･安全

ePerl eRuby - 埋め込みスクリプト

- ドキュメントなどの中にコードを埋め込む
- mod_perl/mod_ruby で PHP のように利用できる

```
<html>
<head><title>Test</title></head>
<body>
  <?
    print "I am ePerl.";
  !>
</body>
</html>
```

参考文献 (MySQL)

- http://www.mysql.com/ MySQL Web (MySQL AB)
  - MySQL 開発･一次配布･サポート
- http://www.mysql.gr.jp/ 日本 MySQL ユーザ会 Web
  - MySQL のドキュメントの日本語訳
  - MySQL の日本語のメーリングリスト
- MySQL 徹底入門
  - 日本 MySQL ユーザ会 著，とみたまさひろ･SoftAgency 監修
  - 翔泳社，2001 年

参考文献 (Perl)

- http://www.perl.org/ Perl.com: The Source for Perl
  - Perl 一次配布
- http://www.cpan.org/ CPAN
  - Perl Module など
- プログラミング Perl 改訂版
  - Larry Wall, Tom Christiansen, Randal L. Schwartz 共著
  - 近藤 嘉雪 訳
  - オライリー･ジャパン，1997 年
- 入門 Perl DBI
  - Alligator Descartes, Tim Bunce 著
  - 田中 幸 訳
  - オライリー･ジャパン，2001年

参考文献 (Ruby)

- http://www.ruby-lang-org/ Ruby Home Page
  - Ruby 一次配布
  - リファレンスなどのドキュメント
  - アプリケーションアーカイブ (RAA)
- Ruby プログラミング入門
  - 原 信一郎 著, まつもとゆきひろ 監修
  - オーム社, 2000 年
- プログラミング Ruby
  - David Thomas, Andrew Hunt 著
  - 田和 勝 訳, まつもとゆきひろ 監修

近いうちに, 資料を以下の URL で公開する予定です.

http://osdn.jp/event/linuxworldcd2001.shtml

END