第9章

押さえておきたいPerlの基礎

本書は Perl の応用テクニックや開発手法を主題として紹介していますが、その際に最低限知っておいてもらいたい変数やパッケージ宣言などの基礎知識、それから CPAN の仕組みなどをここで紹介します。

CPAN は開発手法としても Perl モジュールの宝庫としても非常に有効なツールである以上に、そもそも Perl という言語自体が CPAN を抜いては語れないものになっています。本書が紹介する様々なノウハウも、その大部分が CPAN を通してしか得られないものになっていますので、CPAN の全体像や CPAN 形式の概要などをよく知らない場合は、ここで理解していただければ幸いです。

また、Windows での Perl のインストールに関しても簡単に紹介します。

9.1 CPAN

CPAN（Comprehensive Perl Archive Network；シーパン）とは Perl 開発者であれば誰もが一度はお世話になったことのある、巨大なモジュール共有インフラストラクチャです。本書も様々な場面において CPAN を引用・利用しますが、Perl は CPAN とともに培われてきたといっても過言ではない言語ですので、CPAN を直接使わずとも要所要所でこの知識が必要となってきます。

あなたがもし誰も未だ夢想していない画期的なアプリケーションを開発するのであれば、CPAN にあがっているリソースはあまり役に立たないかもしれません。しかし、もしあなたが Perl を毎日業務で使うシスアドであれば、日々書いているコードの半分から 8 割くらいは CPAN にすでにあがっているモジュールで事足りることでしょう。

CPAN の仕組み

一口に CPAN と言っても、様々なコンポーネントで構成されている上に、皆一様に名前のどこかに CPAN という文字列が入っているので最初は分かりにくいかもしれません。CPAN の構成は大まかに以下のようになっています：

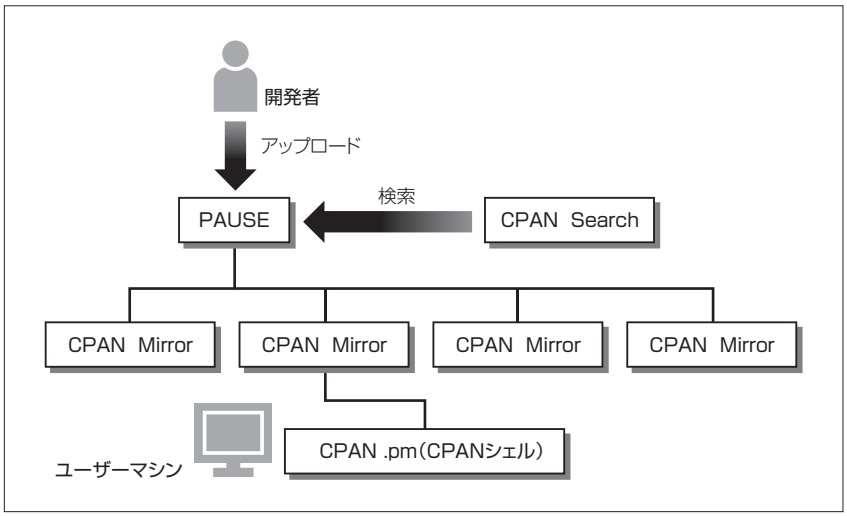

図 9.1 CPAN の構成

CPAN の本質は、複数のミラーサイトによる Perl モジュールのアーカイブです。これらのモジュールを開発している人たちは、まず「PAUSE」という開発者向けのポータルにモジュールをアップロードします。この際、モジュール本体の他に必要なメタデータおよび同梱ファイル類を一つにまとめたものを「ディストリビューション」と呼び、PAUSE にはモジュール単体ではなくディストリビューションとしてアップロードします。開発者からアップロードされたディストリビューションは、特定のパターンで構成されたディレクトリに格納されるとともに、そのディストリビューションに含まれる情報がインデックスされます。筆者の場合、以下のようなディレクトリ以下にディストリビューションが格納されています：

```
D/DM/DMAKI/ モジュール名 - バージョン .tar.gz
```

PAUSE の情報は世界中に存在するアーカイブのミラーサイト（CPAN Mirror）が定期的にアクセスし、更新分を取得するようになっています。後述の CPAN.pm がディストリビューションを取得してくるのは、実際にはこの CPAN Mirror のどれかからになります。

CPAN モジュールの検索

使えそうなモジュールを探す際にもっとも便利なのが「CPAN Search」（http://search.cpan.org）です。CPAN Search では、前述の CPAN Mirror と同様に PAUSE から情報を作成・インデックスし、Web から検索可能な状態にします。モジュールをインストールする前に CPAN Search で確認してから、実際に CPAN.pm でインストールするという手法が最も一般的です。

画面 9.2　CPAN Search

CPAN モジュールの利用

apt-get、rpm や MacPorts 等のパッケージ管理システムがすでにインストールされている環境ならば、それらを使うと楽です。なお、本書に出てくるモジュールをインストールする際に、CPAN.pm を使うか、それらのパッケージ管理システムを使うかは、皆さんの運用ポリシーに合わせて適時変更してください。筆者の経験上、どちらも特に問題なく使えますが、C コンパイラや外部ライブラリが必要なモジュールに関してはパッケージ管理システムを使用したほうが楽です。

なお、以下の設定は主に Mac を含む UNIX 系の環境を前提としています。Windows 系の場合は「9.6 Windows 環境での Perl の準備」を参照してください（P.319）。

Column CPAN 以外のパッケージ管理システムでの注意点

Perl を使う上で Perl 本体およびモジュールの管理を aptitude や rpm 等のパッケージ管理システムに任せている環境は数々ありますが、これらを CPAN からのインストールと併用して使用するのには少しだけ注意が必要です。

仮に、特定の CPAN モジュールにバグが見つかり、すぐに作者が対応して CPAN に新バージョンをアップロードしたとしても、これらの管理システムに最新の CPAN モジュールが登録されるまでにはかなりのタイムラグがあります。

テストツールの章で触れている Test::Exception は、現時点で Debian（Stable) に 0.21 がインストールされますが、最新版は 0.27 で、実に 3 年前のものが依然 aptitude に登録されているわけです。

それぞれのモジュールの進化が激しい Perl は、バグなどの対応が早い反面、このように他のツール側の対応が大幅に遅れるケースが目立ちます。C ライブラリの依存関係などはパッケージ管理システムガ便利ですが、筆者のおすすめは、可能な限り Perl 本体は自分でコンパイルし、モジュール類は全て CPAN から最新のものをインストールするという手法です。

必要な外部ツール

CPAN 経由でモジュールをインストールする際には、使用されているシステムによって別途外部ツールをインストールする必要があります。例えば Debian では、一般的に以下のツールが必要になります：

- make
- gcc
- g++
- unzip
- ftp/wget/curl

システムによってインストールが必要なツールは変化しますので、自分のシステムに合わせて準備してください。

CPANシェルの起動

CPANからモジュールをインストールする場合に、実際にユーザーが対話するのがCPAN.pmおよび、CPAN.pmを使用したインタラクティブシェルの「CPANシェル」です。CPANシェルを起動するには管理者権限で以下のコマンドを実行します：

```
> perl -eshell -MCPAN
```

これが初めての起動なら、システム情報やCPANの設定を聞かれるので、ご自分のシステムに合わせて適宜設定してください。

CPAN.pmのアップグレード

CPANシェルを起動したら、CPAN.pmをアップグレードします。CPAN.pm自体はPerlの標準モジュールですので、すでにインストールされていますが、最新版にしておいたほうが一般的には便利です：

```
cpan> install Bundle::CPAN
```

と打ち込んでください。これで最新のCPAN.pmおよびその依存関係がダウンロードされ、展開・インストールされます。

CPANモジュールのインストール

CPANシェルを起動して、インストールしたいモジュール名を

```
cpan> install モジュール名
```

のように指定すると、あらかじめ選択しておいたCPAN Mirrorから現在登録されているモジュールのリスト等のメタデータが取得され、該当するモジュールが含まれる最新のディストリビューションが取得・展開・テスト、そしてインストールされます。この作業さえ行ってしまえば、そのモジュールが使えるようになるわけです。

Column インストールのショートカット

最新の CPAN.pm にすると、cpan コマンドも同時にインストールされます。その後の CPAN モジュールのインストールは、シェルを起動せずとも

```
> cpan Module名
```

とすると、インストールが行えるようになります。

例えば MD5 ハッシュを使おうと思った際、自分で最初から実装していたら時間がかかってしまいますが、CPAN が利用可能であれば cpan シェルから

```
cpan> install Digest::MD5
```

と入力して、Digest::MD5 モジュールをインストールしてから

```
use Digest::MD5 qw(md5_hex);
print md5_hex($value); # MD5
```

のようなコードを書くだけで簡単に MD5 ハッシュの計算ができます。このコードを書いた後に気が変わり、やっぱり MD5 ではなく SHA-1 を使いたいと思ったのであれば

```
cpan> install Digest::SHA1
```

とすれば、これまた SHA1 が利用可能になります。Perl は PHP のように組み込み関数は多くはありませんが、その代わり、後からいくらでも拡張可能であるという部分に開発者達は創造力を働かせ、ユーザーはその恩恵にあずかることができるようになっています。すなわち CPAN を利用すると、先人達が築き上げてきてくれた知恵の結晶を容易に組み込めるようになり、開発効率を大幅にアップすることが可能になります。

これらのインフラストラクチャ全体を含めて CPAN と呼びます。CPAN そのものは 1995 年に Jarkko Hietaniemi が立ち上げ、その後このインフラストラクチャのおかげで開発者同士がお互いのモジュールを利用することが容易になり、今の Perl 文化の生成に大きな影響を及ぼしています。

CPAN を利用せずに Perl を利用することも当然可能ですが、Perl という言語が持つ機能の大部分を使えないのと同等です。ぜひ積極的に CPAN を利用するようにしてみてください。

Column　CPAN Mirror のタイムラグ

CPAN をすでに使っている方なら、一度は CPAN Search と実際に CPAN.pm でインストール可能なモジュールが違う現象に出くわしたことがあるかと思います。

これは本文の通り、CPAN インフラストラクチャ内の各コンポーネントが同期するタイミングが微妙に違うためです。例えば CPAN Search は PAUSE にアップロードが発生してから 1、2 時間以内に更新されることがほとんどですが、ミラーサイトは更新までの期間が比較的長いため、1 日後にならないと情報が同期されないことがあります。

また、CPAN Search そのものも複数サイトにミラーリングされているので、アクセスするタイミングによって、物理的に違う場所にホスティングされたミラーサイトにアクセスしていることがあります。その際、それぞれのサイトが同期されるタイミングが違うので、ここでもまた微妙にずれが見られることがあります。

9.2 CPAN からダウンロードしたディストリビューションの処理

CPAN からダウンロードされてきたディストリビューションは、ただ単純に展開されてコピーされるわけではありません。CPAN に登録されているディストリビューションはどのような OS のどのバージョンの Perl に対してインストールされるかは前もって分からないので「インストールしたけど動かない」という状況を回避するために様々な仕掛けがほどこされています。

CPAN.pm はディストリビューションをダウンロードしてくると、それをインストールするまでに大まかに以下の処理を行います：

1. 圧縮ファイルの解凍・確認

まず圧縮されているディストリビューションが解凍され、その中にある MANIFEST ファイルが確認されます。この MANIFEST ファイルに載っているのに存在しないファイルがある場合は警告されます。

2. Makefile.PL の実行

CPAN.pm を起動した perl 実行ファイル（同一マシンに複数個 perl 実行ファイルがインストールされている可能性もあるため）を使って、Makefile.PL というファイルを実行します。この際、Makefile.PL 内で実行されるコードが、使用された perl から Perl のバージョンや、インストール先のパスなどの様々な環境情報を取得しつつ、Makefile というファイルを作成します。Perl 本体の必須バージョンの確認や、C ライブラリの確認などはこの段階で行われます。

この Makefile は make プログラム（Windows では nmake/dmake 等）が使用し、その後の作業は基本的にこの make プログラムと Makefile に書かれた情報によって進められます。

なお、上記 Makefile.PL の代わりに Build.PL というファイルも使用できますが、本書では Makefile.PL に統一します。

3. 依存関係の解決

Makefile が作成された後、依存関係の解決が試みられます。依存関係として指定されているモジュール類もダウンロードされ、同様のインストール処理が行われた後に再度現在のモジュールのインストールが試みられます。

4. make の実行

make コマンドを引数無しで実行すると、Makefile 内のデフォルトルールが適用され実行されます。この場合は一般的に「ディストリビューション内のファイル群をインストール可能な状態まで準備する」という複数のコマンドが走ります。例えば C ファイルがコンパイルされ、モジュールファイル内からドキュメントが抽出され、man コマンドから扱える状態に加工されます。

5. make test の実行

C ファイルのコンパイルやその他の準備が終わったら、テストの実行です。make test コマンド実行すると t/ ディレクトリ以下のテストが実行され、ディストリビューションが動作するか確認されます。

6. インストール

テストに問題がなければ make install が実行され、他のモジュール達と一緒に /usr/local/lib/perl5 以下などの所定の位置にモジュールがインストールされ、使用可能な状態になります。

このように一口に「CPAN からインストールする」と言っても、実は様々なステップが絡んできます。しかしこの面倒なステップのおかげで、CPAN にあげられたモジュールは自動的に我々のマシンにインストールされて使用できるようになるわけです。