

# パッケージマネージャ全般のFAQ

- [パッケージマネージャ全般のFAQ](#)
  - [RPMとDpkgってどこが違うの？](#)
    - [設定ツールの分離 vs 統合](#)
    - [スクリプトの付属形態](#)
    - [srpm vs diff](#)

## RPMとDpkgってどこが違うの？

「ソフトをビルドしてファイルにまとめ、マシンにインストールする」という基本部分は共通しているものの、いくらか違いがあるので以下に列挙します。

### 設定ツールの分離 vs 統合

RPMでは、パッケージシステム自体には、インストールした設定ファイルを編集するフロントエンド系の機能は含まれない。独自アプリとして分離されている。

フロントエンド部分が他のアプリと同じように独立しているので、製作者が独自の設定ツールを載せられる。ディストロ開発者の意欲次第で工夫が可能。RPM系のディストロでは、Dpkgを採用したものに比べて設定系のフロントエンドが使いやすく工夫されてるものが多め。

逆に、Dpkgではパッケージシステムに設定機構を搭載し、パッケージ製作者は簡単なスクリプトを書くだけで同じ設定機構を再利用できるようにした。わざわざソフトごとにフロントエンドを作る必要がないし簡単に対応できるので、ソフト開発者側で対応しやすい。設定できるソフトはRPM系のディストロよりも多め。だが、どのソフトも同じインターフェースを再利用する関係から、ただ値を入れて進めるだけの質素な画面が出てくることが多い。

どちらにしてもウィンドウシステムとかサーバーみたいなシステム構築系のソフト用。設定ファイルがつかない単純なコマンド系のプログラムとか、設定の保管に独自の機構を使うデスクトップ系のアプリケーションではあまり関係なし。

### スクリプトの付属形態

RPMでは、基本的にソースファイルが入ったtarballとspecファイルを別のディレクトリに入れておいてパッケージを作成する。これは、ディストロ開発者がいるんなところからソースファイルをとってきた上で、それぞれに対応するspecファイルを記述するにはやりやすい慣習だが、ソフト開発者が自らそのディストロ用のパッケージを作ろうと思ったらやや面倒になる。

RPMにする前はコードを編集するたびにただmakeだけ実行してればよかったわけだが、RPMにしようと思ったらまずtar.gzを出力して、specも編集してrpmを吐く必要がある。既存のソフトのコードに変更を加える場合、diffで変更内容をパッチに取ったりする必要があるかも。

Dpkgでは、ソースファイルとパッケージデータを一緒に保持する。ソースファイルの中にファイルを加えてやるだけで対応できるし、コマンド1つでパッケージが作れるので楽。変更した内容も自動でdiffに加えてられるし。

RPMの方式はディストロ開発者から見れば単純で分かりやすい。パッケージ製作者が対応しやすいのはDpkgかもしれない。

### srpm vs diff

RPMではspecファイルを含めたソースパッケージの配布にsrc.rpmという独自の形式を使う。が、

Dpkgではオリジナルのtar.gzやdiffをバラに分ける。

RPMの場合、持ち運びしやすくかさばらない。Dpkgの場合はいちいちインストールしなくても中身が見れる。といった利点があるかも。