第4章 大規模開発向け…必要なパッケージをダウンロードし，ビルド&リンクを実行する

パッケージ・マネージャ Cargo

池田 有

ビルド・システム兼パッケージ・マネージャ「Cargo」

Cargoにおけるパッケージとは，パッケージ・マネージャで管理できるRustプログラムの単位のことです．

ここで言う管理とは，簡単にはそのRustプログラムをビルド・リンクする際に必要なパッケージ（大体の場合ライブラリ）を把握し，ダウンロードし，ビルド&リンクを実行することです．これがCargoの役目です．

● 小規模プログラムなら使わない場合も

Cargoの説明をする前に，Cargoを使わないでRustプログラムを作成する方法について説明します．Rustをちょっと試してみたいとき，小さく簡単なプログラムであればCargoを使わなくても大丈夫です．

手順は次の通りです．

1，ソースコードなどを格納するフォルダを作成
2，main.rsを好きなテキスト・エディタで書く
3，コンパイル：rustc main.rs
4，実行：./main.exe

gccでコンパイルするときと，あまり変わりありません．ライブラリを使わない，ごくごく小さなプログラムだと，これで問題はありません．.

より詳しい説明は公式ドキュメントにも説明が掲載されています．

```
https://doc.rust-jp.rs/book-ja/ch01-02-hello-world.html
```

● Cargoの使い方

開発が大規模になり，さまざまなライブラリを使いたくなった場合を考えてみます．

例えばgccの場合，あらかじめ必要なライブラリを準備し，ビルド手順を明示的に表すためmakefileを作ります．Rustの場合は，Cargoを使います．

Cargoは構成管理やビルドを行うシステムであると同時にコマンドでもあるので，Cargoを使うには小文字のcargoコマンドを使います，

▶新規アプリケーション・パッケージ作成

```
cargo new hello_cargo
```

このコマンドを実行すると以下の物が作成されます．

- フォルダhello_cargo
- フォルダ内にmain.rsと，Cargoパッケージの情報を記述するCargoマニフェスト・ファイルであるCargo.toml
- 新しいGitリポジトリ

▶ビルド

```
cargo build
```

このコマンドで以下が実行されます．

- 中間・最終生成物格納フォルダ（デバッグ・ビルドの場合 target/debug）や，その他必要なフォルダが作成され，実行ファイルが格納されます
- プロジェクトの全依存関係を記録するCargo.lockも作成されます
- cargo runとすると，ビルドだけでなく実行まで一気に行えます

このひとまとまりを，パッケージと呼びます．

▶コード・チェック

```
cargo check
```

このコマンドで，バイナリ生成せずにコードをチェックして，エラーの有り無しを確認できます．

▶リリース・ビルド

```
cargo build --release
```

このコマンドで，releaseフォルダに，最適化された実行ファイルを生成できます

より深いCargoの使用方法は，公式ドキュメントに掲載されています．

```
https://doc.rust-jp.rs/book-ja/ch01-03-hello-cargo.html
```

このように，言語仕様のレベルで，パッケージ・マネージャの機能が定義されていることも，Rust言語の特徴です．

ここまでで，Cargoの役目の中のビルド&リンク・フェーズにおけるパッケージ情報を管理する部分を説