ダウンロード・データあります

組み込みRustのライブラリ 便利クレート探偵団

第6回 任意のデータを シリアライズ/デシリアライズするserde

中林 智之

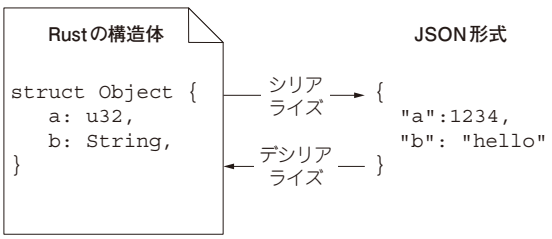

図1 シリアライズ/デシリアライズでJSON形式に変換した例

組み込みで使えるプログラミング言語として注目されているRustの組み込み開発で役立つライブラリ(クレート)を本連載では紹介します.

今回は,一度使うと他のプログラミング言語に戻れないほど便利なシリアライザ/デシリアライザのserde(サーディ)[1]を紹介します.

シリアライズ/デシリアライズとは

シリアライズはデータ構造やオブジェクトを文字列やバイト列に変換する処理を意味します.例えば,Rustの構造体をJSONのような文字列(**図1**)やMessagePack[注1]のようなバイト列に変換します.

このような変換をすることで,オブジェクトのデータをファイルに保存したり,ネットワーク経由で通信したりすることが可能になります.

デシリアライズはシリアライズの逆,つまり,文字列やバイト列をデータ構造やオブジェクトに復元する処理のことです.つまり,JSONやMessagePackからRustの構造体に変換することを言います.

シリアライズによって作られたデータをデシリアライズすることで,異なる環境で作られたデータを元の状態に復元できます.

注1:JSONの置き換えとして使うことができる効率の良いバイナリ形式のフォーマット.https://msgpack.org/ja.html

リスト1 serdeの基本的な使い方

```
#[derive(Serialize, Deserialize)]
struct Point {
    x: i32,
    y: i32,
}
```

Rustのシリアライズ/デシリアライズ事情

Rustでは多くのシリアライズ/デシリアライズ処理がserdeのフレームワークに従って実装されています(例外ももちろんあるが).Serde Overviewに概要から独自のシリアライザ/デシリアライザを実装する応用的な手順まで,ドキュメント[2]が整っています.

● 任意のフォーマットに対応している

serdeは,構造体に`#[derive(Serialize, Deserialize)]`というアトリビュートを書くだけで使えます(**リスト1**).

たったこれだけで,その構造体のオブジェクトを任意のフォーマットにシリアライズしたり,逆に任意のフォーマットから構造体オブジェクトにデシリアライズしたりできる優れものです.

serdeの強力なところは,シリアライズ/デシリアライズする対象のフォーマットが限定されない,という点です.対象フォーマットはJSONを始め,YAMLや,MessagePackのようなバイナリ形式のフォーマットまで,幅広くサポートされています(**図2**).

serdeのウェブ・ページに,serdeを使って実装されているシリアライザ/デシリアライザの一部が掲載されています(**表1**).このリストはほんのひと握りであり,crates.ioでserdeに依存するクレート[4]を見ると,2024年6月時点で37,000クレート以上あり,いかにserdeが重要なエコシステムになっているかを計り知ることができます.

ちなみに筆者もスマート・ホームの通信規格ECHONET Liteのserde実装echonet-lite[6]を作っています.

ダウンロード・データはこちら
https://interface.cqpub.co.jp/2410crate/