FASTA Parser Implemented in Rust with Nom

Tianyi Shi

2020-08-23

Contents

Nom makes it possible to

1 Parsing a single FASTA Record

An example of FASTA record is shown below.

```
>gi|2765658|emb|Z78533.1|CIZ78533 C.irapeanum 5.8S rRNA gene and ITS1 and ITS2 DNA
CGTAACAAGGTTTCCGTAGGTGAACCTGCGGAAGGATCATTGATGAGACCGTGGAATAAACGATCGAGTG
AATCCGGAGGACCGGTGTACTCAGCTCACCGGGGGCATTGCTCCCGTGGTGACCCTGATTTGTTGTTGGG
CCGCCTCGGGAGCGTCCATGGCGGGTT

>(another FASTA record)
```

Each record starts with a `>` character, which is followed by the title on the same line. The title can be further divided into an ID and description, which is separated by a space character. In this case, the ID is 'gi|2765658|emb|Z78533.1|CIZ78533' and the description is 'C.irapeanum 5.8S rRNA gene and ITS1 and ITS2 DNA'.

The sequence starts from the second line. There can be line breaks within the sequence.

1.1 Representing the Record in Rust

When we parse data, we *parse it into something that is meaningful in that language.* In Rust, users can define custom `struct` and `enum` data structures, and in this case, a C-like `struct` is the most appropriate.

To make it simple, we define one two fields, `title` and `seq`:

```rust
fn main(){
#[derive(Debug)]struct Record {  title: String,  seq: String}
}
```

1.2 Parsing the Title

To make it