A Closer Look at Parsers

Parsers in detail

The previous section described parsers at a high-level and described when they may be used. In this section, we're going to look at parsers in closer look. The concepts discussed here are foundational to how CongoCC works and is required if you want to work on CongoCC or do advanced things with it.

Tokenizers and Lexers

Imagine you're writing a simple program to count the number of words in an input english-language string, like "The quick brown fox jumps over the lazy dogs". Your program would probably do something like this: `count_words = len(input_string.split())`. This works because the rules in english are simple: characters in a sentence not separated by a space are part of the same word and words are separated by a space. In this program, the sentence is split into "tokens" using `input_string.split()` and the number of tokens is the number of words.

A tokenizer takes as input a text file that follows the rule of some language and breaks the text file into individual tokens. It's more complicated than just splitting the text file by space because what constitutes a "token" is different in different languages. Take the following line of code:

```
string some_var = "hello world";
```

As you can see, there are 5 tokens:

- `string`
- `some_var`
- `=`
- `"hello world"`
- `;`

Unlike the rules of english, it's not enough to just on whitespace (e.g.: the space in `"hello world"` belongs to the token) and characters next to each other can be part of different tokens (e.g.: the `;` after `"hello world"`)

In our string counting program, we only cared about the *number* of words. In more advanced programs like parsers and compilers, we need to understand how the tokens actually relate to each other and to do that, we need to what the tokens represent. Following the earlier example, it's more useful if the tokenizer can look at the line of the code and produce the following output:

- Token: `string`, TokenType:
- `some_var`
- `=`
- `"hello world"`
- `;`

Revision #2
Created 9 July 2023 01:30:26 by Ali Razeen
Updated 9 July 2023 05:32:59 by Ali Razeen