Parsec Cheat Sheet

Parse several occurrences of a function and look for an end of file afterwards.

```
do result <- many line
   eof
   return result
```

The following parsers match a char or a string.

```
char ‘a’
string “ab”
```

If the parser on the left doesn’t consume any input, go to the right one and look if it succeeds.

```
char ‘a’ <|> char ‘b’ <|> char
‘b’
```

Display an error message if the parser before didn’t succeed

```
string “\n” <?> “end of line
missing”
```

Apply the function, if the Parser before succeeded.

```
char ‘a’ >> line
```

Return a parser that returns all characters not in the list.

```
noneOf “abc”
```

Parses zero or more occurrences of the given parser.

```
many (char ‘a’)
```

Parses one or more occurrences of the given parser.

```
many1 (char ‘a’)
```

Parses zero or more occurrences of p, until `newline` succeeds. Returns a list of values returned by p.

```
manyTill (char ‘a’) newline
```

Skips zero or more white space characters.

```
spaces
```

Succeeds for any character and returns the parsed character.

```
anyChar
```

Runs a parser `(char ‘a’)` over `“ab”`. The `“(unknown)”` (FilePath) is only used in error messages and may be the empty string. Returns either a ParseError (Left) or a value of type a (Right).

```
parse (char ‘a’) “(unknown)”
“ab”
```

Parses zero or more occurrences of `(char ‘a’)`, separated by `(char ‘b’)`. Returns a list of values returned by `(char ‘a’)`.

```
sepBy (char ‘a’) (char ‘b’)
```

Parses zero or more occurrences of `(char ‘a’)`, separated and ended by `(char ‘b’)`.

```
endBy (char ‘a’) (char ‘b’)
```

Like `(char ‘a’)` except that it doesn’t consume any input when the parser doesn’t match.

```
try (char ‘a’)
```

if the first doesn’t succeed then the next is applied.

```
product :: Parser Transaction
product =
  do
    string "refund"
    spaces
    p <- price
    char ';'
    spaces
    return (Refund p)
  <|>
  do
    productName <- many1 letter
    spaces
    p <- price
    char ';'
    spaces
    return (Product productName p)
```

Quellen

Hoogle[1]
Real World Haskell, Chapter 16[2]
Monads Are Not Scary - Examples.hs[3]

[1] http://www.haskell.org/hoogle/, abgerufen am 04.06.2013

[2] http://book.realworldhaskell.org/read/using-parsec.html, abgerufen am 04.06.2013

[3] http://www.algorithm.com.au/downloads/talks/monads-are-not-scary/Examples.hs, abgerufen am 04.06.2013