Monadische Parserkombinatoren

Ingo Blechschmidt
<iblech@speicherleck.de>

Ziel: S-Ausdrücke parsen

```haskell
data Exp = Atom String | List [Exp]

-- Eingabe:
(+ 1 (* 2 3) (* 4 5))

-- Syntaxbaum:
List
    [ Atom "+"
    , Atom "1"
    , List [ Atom "*", Atom "2", Atom "3" ]
    , List [ Atom "*", Atom "4", Atom "5" ]
    ]
```

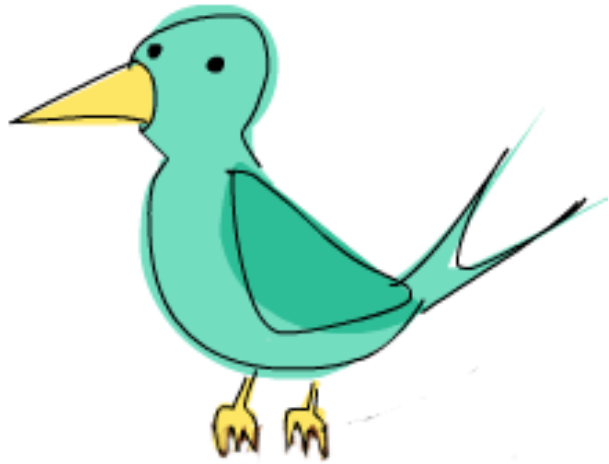
Ziel: S-Ausdrücke parsen

```haskell
data Exp = Atom String | List [Exp]

parseExp :: Parser Exp
parseExp = choice [ parseSymbol, parseList ]

parseSymbol :: Parser Exp
parseSymbol =
    fmap Atom $ many1 alphaNum `andThen` spaces

parseList :: Parser Exp
parseList = do
    token "("
    elems <- many parseExp
    token ")"
    return $ List elems
```

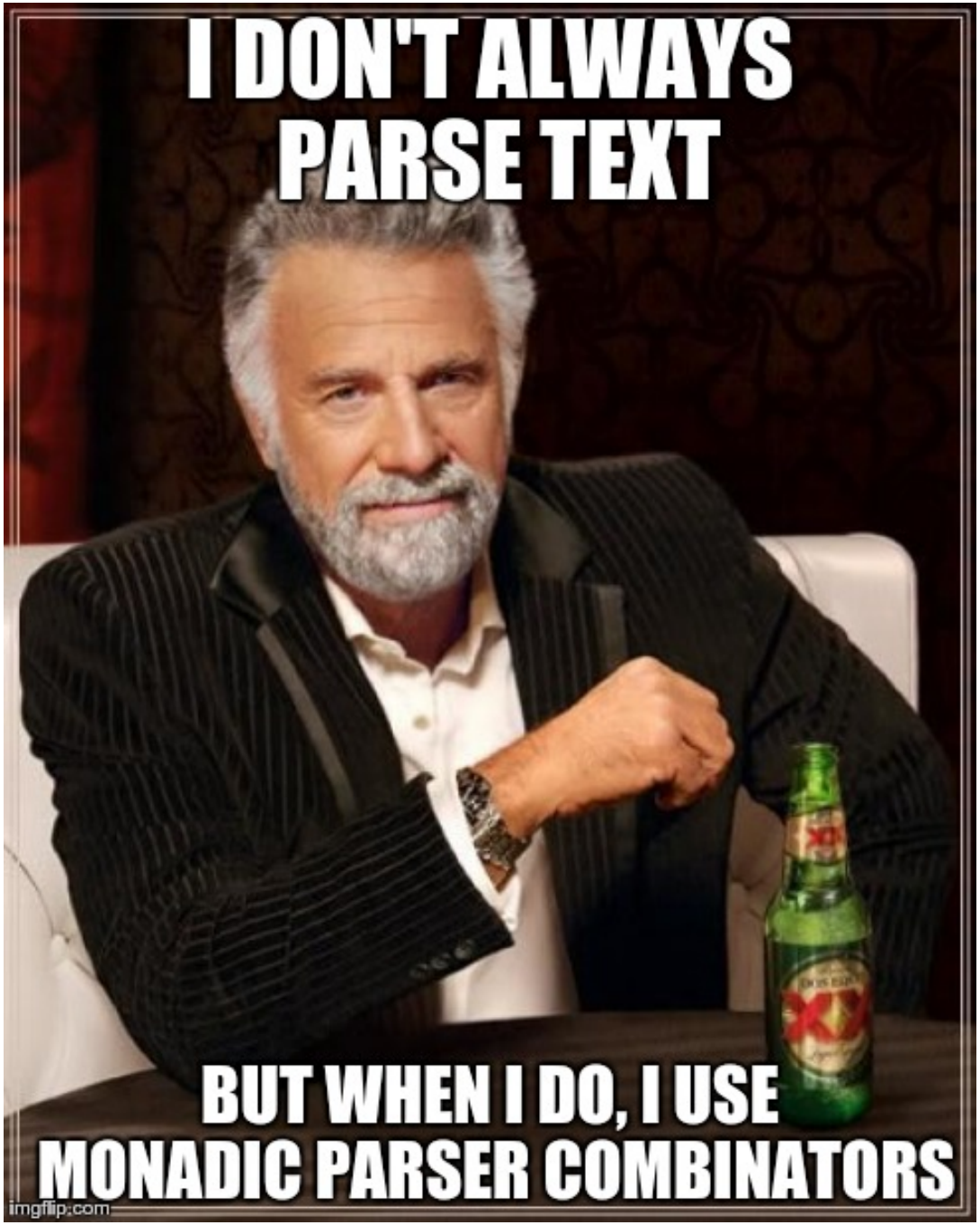


Live-Coding

```
$ vim parsen-macht-spaß.hs
```

Was fehlt noch?

- Unsere naive Bibliothek leckt Speicher.
- Wir geben keine guten Parse-Fehlermeldungen aus.
- Wir haben keine Kombinatoren zum Parsen von Termen mit Operatoren.


I DON'T ALWAYS
PARSE TEXT
BUT WHEN I DO, I USE
MONADIC PARSER COMBINATORS
imgflip.com