Towards a Single Lexical/Syntactic Description

Current situation:

- ANTLR and DLG have different syntax
- DLG syntax is non-standard.

Proposition:

Combine ANTLR/DLG syntax; i.e. use ANTLR notation to specify lexical structure as well as syntactic structure.

```
GETTOK          // special rule name
    :   WS
    |   ID
    |   NL
    ;

ID : ( 'a' .. 'z' )+ ;
WS : ( ' ' | '\t' )+ <<skip();>> ;
NL : '\n' <<skip(); newline();>> ;
```

Good things:

- Combined syntax
- Faster?
- Can insert actions during token recognition
- Can use semantic/syntactic predicates
- LL(k) stronger than regular expression; i.e., can call “lex modes” and have it return to call site.
- MUCH easier to debug (can read code rather than trace through a table of integers).

Bad things:

- Bigger code size
- No automatic left factoring

Left-factoring Problem

The following description could be a problem:

```
GETTOK
    :   INT
    |   FLT
    ;

INT :   (‘0’..’9’)+ ;
FLT :   INT ‘.’ INT ;
```

Rule `GETTOK` would not know how to choose between `INT` and `FLT` since both start with `INT`.

This could be overcome with some fancy ANTLR left-factoring.

My belief is that the advantages outweigh the disadvantages. Basically, it’s a faster way to build the scanner you’d build by hand.