xkrcma04

04: Lexical/syntactical structures and internals of ML programming language

This presentation will provide necessary information to recognize ML for attendees with a predominant C/Java/Python background. We will be informally using code snippets instead of Backus-Naur Form to showcase various constructs.

The listener should become aware of algebraic data types and of the comfort provided by pattern matching; understand that we do not have to write parentheses around function arguments and observe a corner case with programmer-defined priorities on infix operations, ponder why we manually declare types (not just because we can), compare module system designs, and get a little insight into how it is (all) done in Standard ML (of New Jersey), so these features could be easily implemented in future compilers.

The "internals" of a language could also mean its semantics; in this case, we would describe the impact of type system on the compilation process and foreshadow the utility of first class functions in an eager language without referential transparency.

We may spend some extra time on anonymous functions, recursive definitions, exception handling or other "popular" ML variants, like OCaml.