Everything is a Function

- Everything is an expression, everything is a function
- “everything is a function returning a value” makes automation easy
    - transpiler simply bolts functions together
    - essence of *currying*
    - toolbox language should support functions efficiently
    - “syntax” is just a combination of functions bolted together in various ways
    - example: if-then-else-endif is a function (like in Lisp, Smalltalk, etc.) always returning a value
- compiler technology knows how to inline functions
    - no added cost in making everything a function
- Lisp macros are a step in this direction

Types and Classes and Prototypes

- Types are a skin on prototype-based substrate
- Classes are types
- Types help optimize apps by separating what can be done early vs. later
  - “early” is currently called “compile time”
  - “later” is currently called “run time”
  - *early —> later* is a continuum, compile-time vs. run-time is an artificial distinction
  - in the end, *everything* is interpreted (hardware is an interpreter)