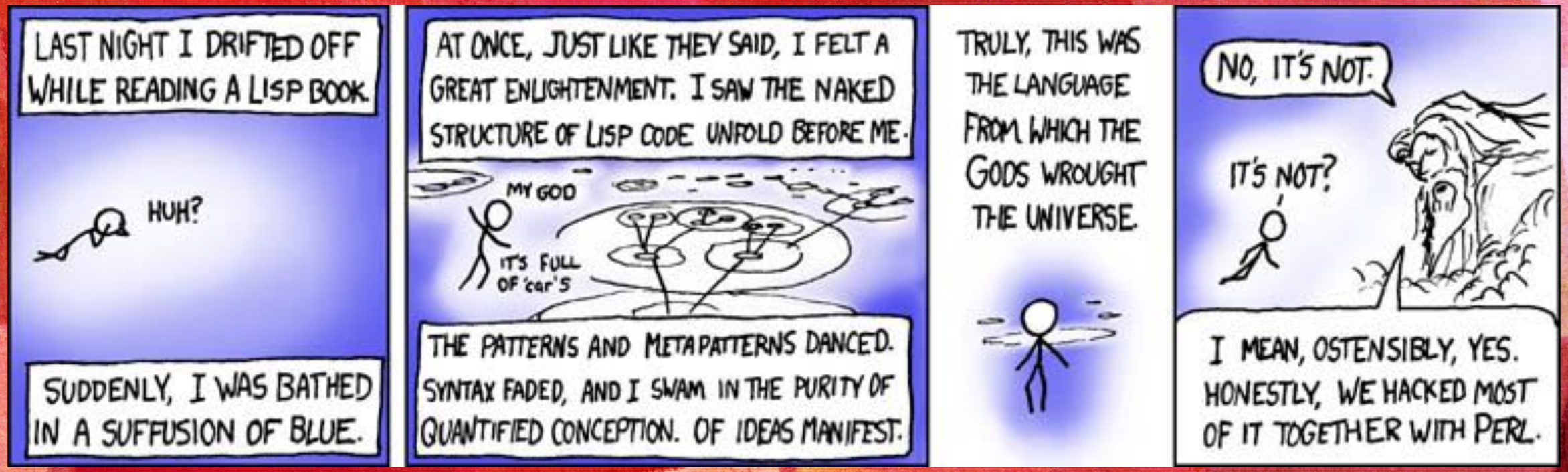
MACROS

Under the micro lens



source: https://xkcd.com/224/

MACROS

- are a way to modify a program during compile time.

- are functions that run at compile time,
  - unevaluated s-expressions => modified s-expression evaluated at run-time

- lisp = list processing

(= CODE DATA)

DIFFERENCES FROM FUNCTIONS

- Arguments to a function are evaluated before the function call, arguments to a macro are unevaluated s-expressions

- Functions are evaluated inside out, while macros are evaluated outside in.

- Macro evaluation happens at compile-time, while function execution happens at run-time.

(OR MACROS (NOT MACROS))*

- Reducing syntax
- Compile-time optimisations
- Defining control structures
- Designing DSLs

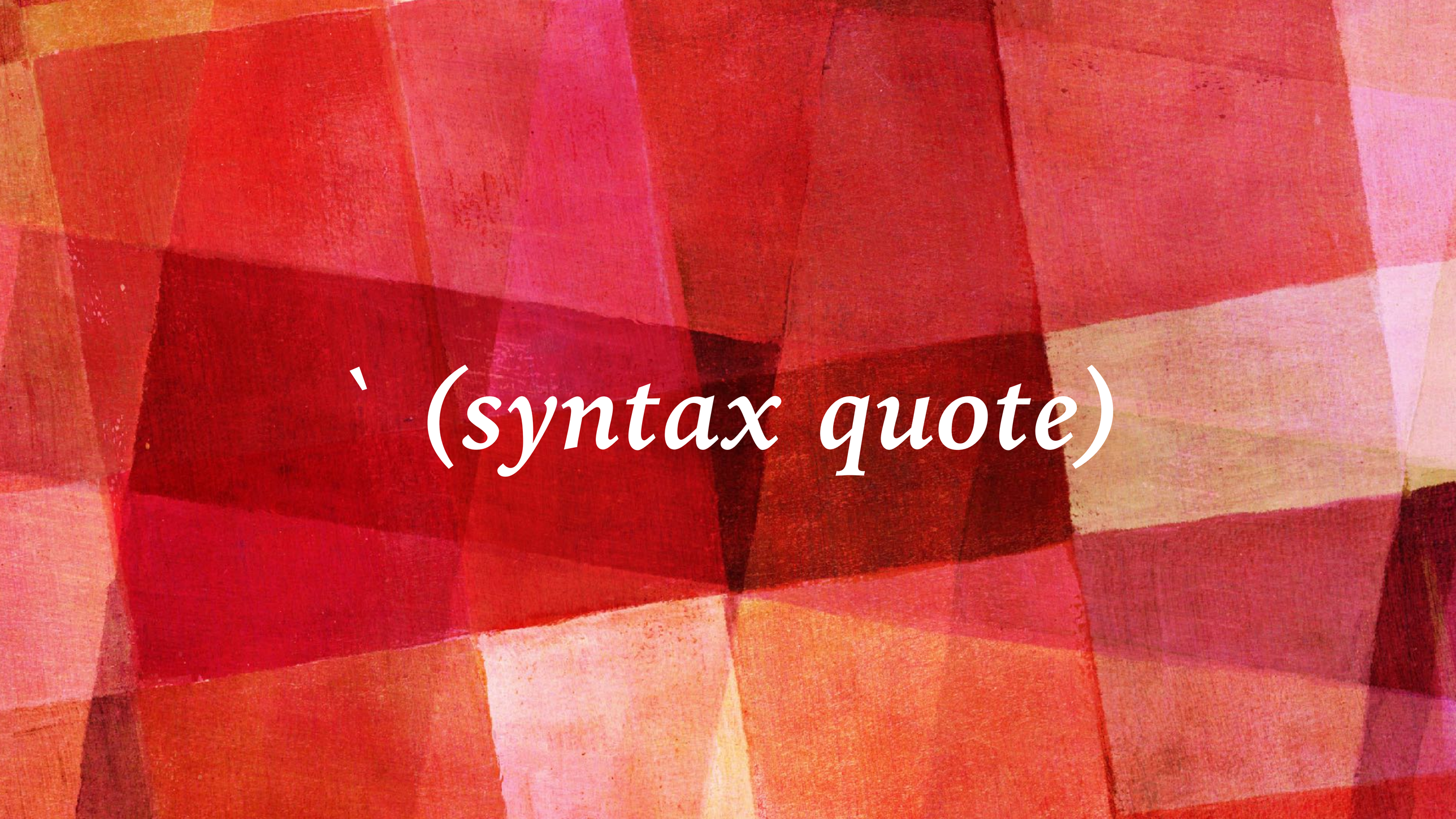
** The use of macros is subject to a risk of increased complexity, please read the documentation carefully before investing*

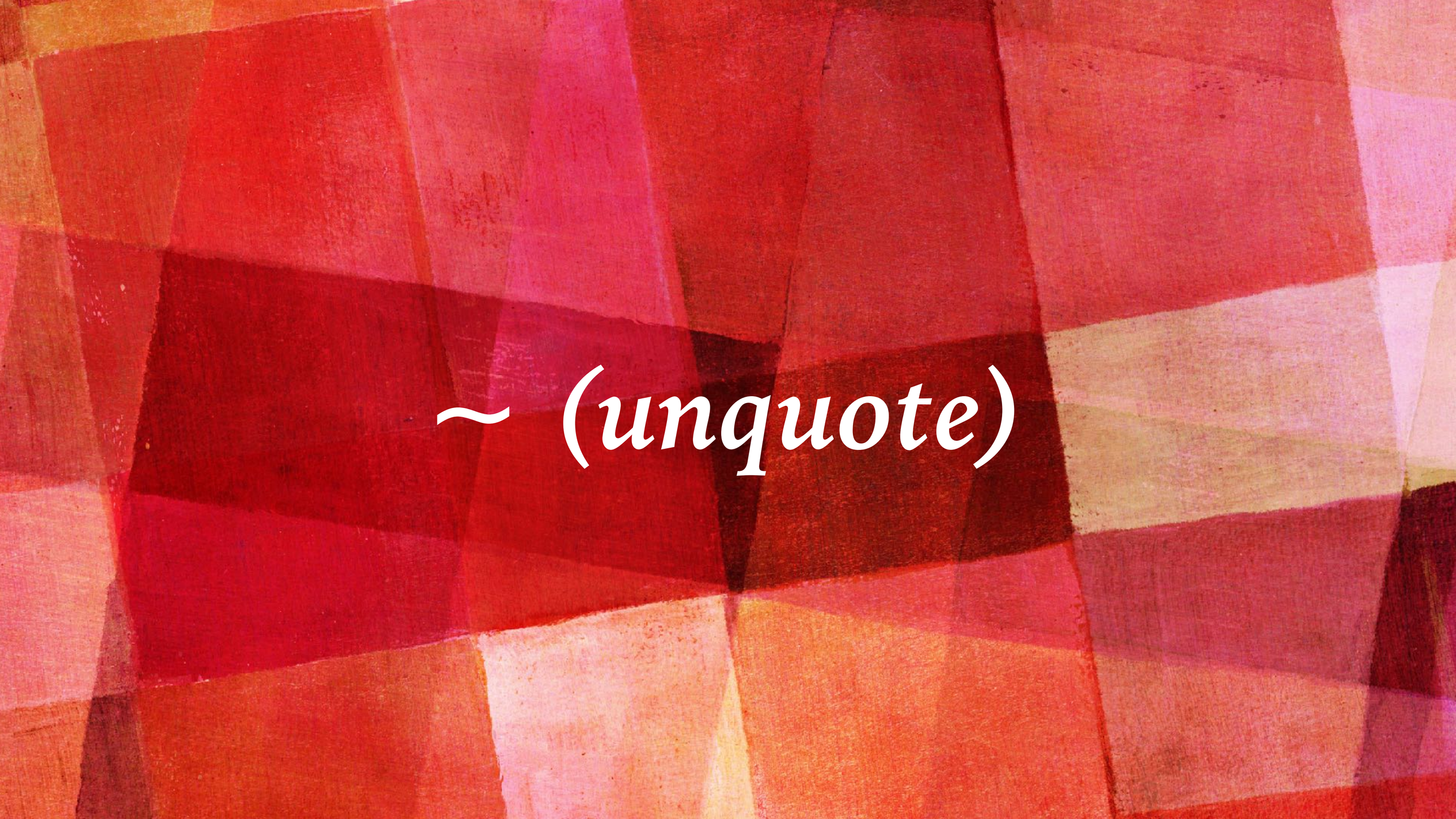
REASONS TO NOT USE MACROS

- Increased complexity. (Runtime vs. Compile time)
- Macros are not first-class in Clojure.
- Data > Functions > Macros

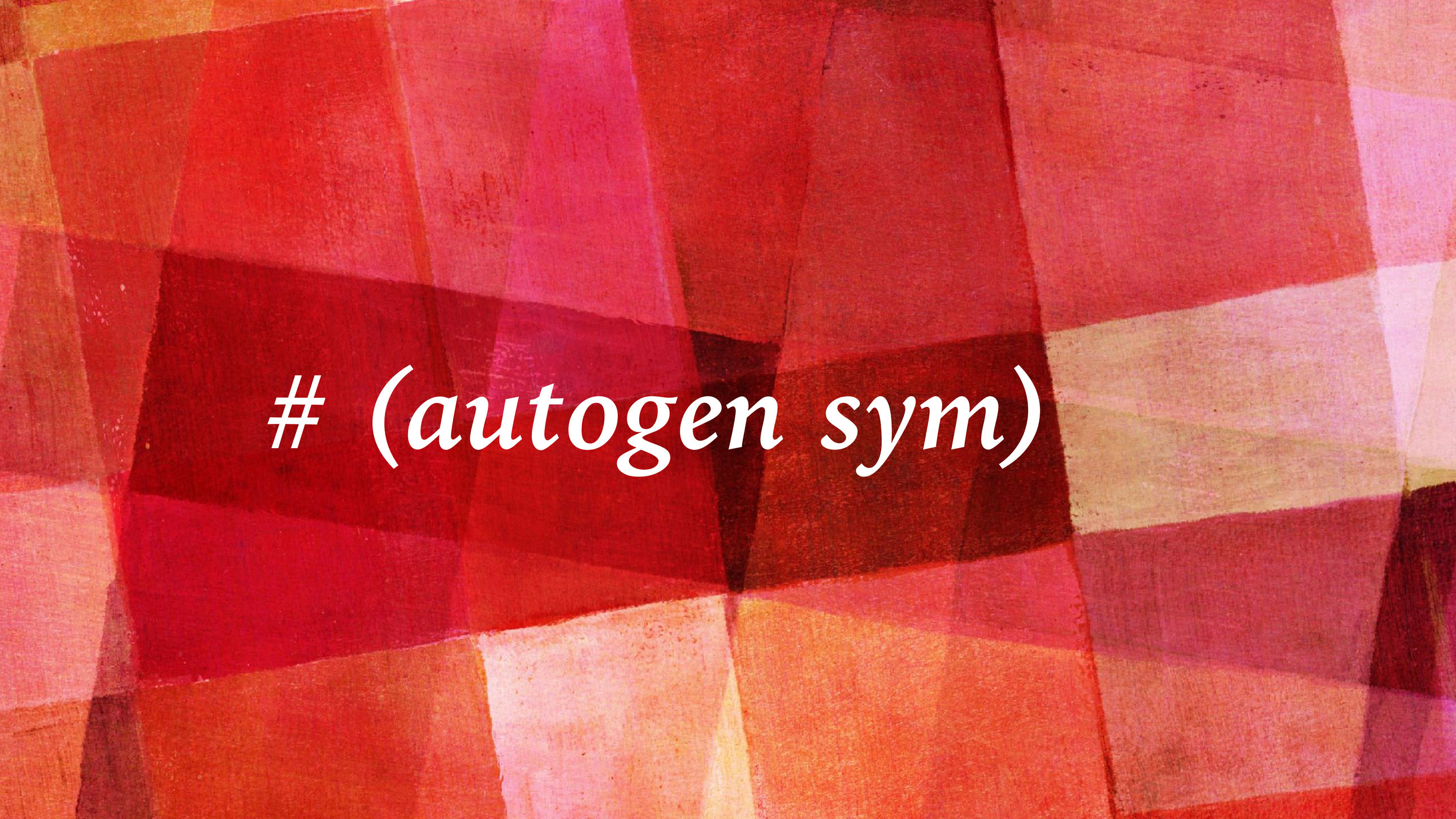

defmacro

‘ (quote)


` (syntax quote)


~ (unquote)


~@ (unquote-splicing)


(autogen sym)

EXAMPLES

THANK YOU