# Monadic programming WTC-2017

- General exposition attitude
  - Good technical description of the presented monadic programming style in WL already exists
    - “Monad code generation and extension”
  - It is better to discuss the big picture in which monadic programming can be placed
    - Mathematical perspective
    - Software architecture perspective
    - Language and DSL design perspective
  - This is one of the last presentations of the first conference day — should be more fun
- Original presentation topics
  - Why monadic programming?
  - Monad definition
    - Formal
    - Informal
    - How-and-what
  - Monad laws
    - The laws
    - List/Apply as Unit/Bind
  - The Maybe monad
  - The State monad
  - R’s dplyr pipelines
  - Code generation for monads
  - Branching and iteration
  - Classification monad
  - Tracing monad
  - Software design with monads
    - Comparison with OOP design patterns
    - DSLs
    - For conversational agents
  - Using monad parsers
- Things to mention or emphasize
  - The document “Monad code generation and extension”
  - Related MSE discussions
    - Granular versus terse coding
    - Functions with changeable global variables
  - The big picture of monadic programming application
    - The Big Berta pattern
    - Functional analysis
    - Cartesian Closed Category
  - Mathematica’s kernel programming used / uses Maybe
  - Can change your perspective on programming
  - Might be the last design pattern you would ever learn
  - Eat your dog food or not
    - While programming a monad...
  - Programming style guide
  - Unit tests
  - Application to tests making and execution
  - Where have we seen this before?
    - UNIX pipeline operators are similar
      - Pipelines and filtters pattern
    - SQL is very close to monadic programming
  - On the RStudio conference this year the pipelining was sort of fetishized