F# Experiences & Perspectives

Dmitri Nesteruk
@dnesteruk

F#... meh.

- Tool support
  - VS support very thin
  - No code analysis tools
    - And VS itself is slow
  - Profiling, unit testing
- Market penetration
  - Thin but we don't know how thin
  - Currently (Sept 2016) 29th on TIOBE
- Are functional aspects worth it?
  - Next slide ☺
- Perceived lack of evolution
  - Open sourced *ergo* uncertain future

This dog is skeptical.

Functional Programming

- First class functions
  - C# has this
- Higher order functions
  - C# has this, e.g., LINQ etc.
- Pure functions
- Recursion
  - No tail recursion in C#, see JIT
- Type inference
- Purely functional data structures
  - Of dubious value
- Immutability
- Function 'magic'
  - Currying/partial application
  - Functional composition >>
  - Pipelining <| |>
- Not really functional stuff
  - Workflows (~ monads)
  - ADTs
  - Tuples
  - Pattern matching
  - 118 operators 😐
- C# will eventually steal everything worth stealing

Traditional F#/functional domains

- Domain-Specific Languages (DSLs)
- Mathematics/algorithms
  - Quant finance
  - Probabilistic models
  - Data science, AI, etc…
- Symbolic processing
  - Parsing/lexing
  - Symbolic differentiation
  - Circuit verification

DSLs

- Top-level declarations
  - No need to 'nest' in namespace+class
  - No need for `main()` entrypoint
- (Curried) function calls without braces/commas
  - doFoo a b c
- No need for explicit declarations (var/let)
  - Can have global-appearing state
- Result: you end up writing English
- Mutability may result in (somewhat) ugly code

Math

- Math in print is (typically) immutable
- Type inference means less typing
  - But can lead to real annoyances
    `2.0 * sign x // invalid!`
- Lowercase functions with no namespace prefix
- No braces in function calls
  - `sin x`
- Conditional logic is better
- Continuous compilation (REPL)
- It's all about libs & tools

Parsing/lexing

- VS parses F# with F#
  - http://fsharppowerpack.codeplex.com
- Fslexx/fsyacc
  - Project template online
- Most power gained from
  - Discriminated unions
  - Pattern matching
    - Active patterns
  - List comprehensions

Conclusion

- Niche language for specific domains
- Okay for math but needs libs
  - Most good libs are C++, but many have .NET wrappers
- Good for algorithmic/back-end tasks
- Useful for DSLs
- Good for quickly defining simple data structures
- Not good for
  - Full OOP
  - Highly mutable constructs
  - UI

Thanks!

Questions?
@dnesteruk
dmitrinesteruk@gmail.com