Functional Programming for rubyists

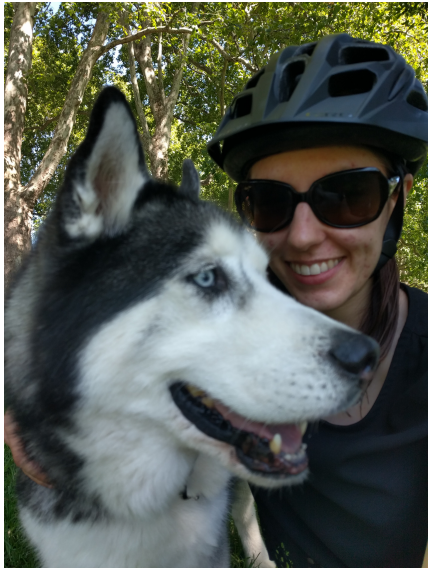
Bianca Gibson

Ruby Conf AU 2016

Rampant side effects

Shifting sands of mutable state beneath your feet

Enter Functional Programming



What should you get out of this?

- What is the core of FP?
- How can it help with problems you've experienced?
- How to use it in ruby projects you're already working on
- It's not that hard and complicated
- No neckbeard required

What will we cover

- Immutability
- Putting off side effects
- Referential transparency
- Common traps when explaining to others and avoiding them
- How to learn more

Stop things changing behind my back

Immutability

- Don't change any values, use only constants
- Instead of changing create a new one
- Recursion and accumulators over loops and mutation
- Confusion over passing by value or reference? Gone.
- Ever thought about using a ! or not? That's worrying about mutation

Image from CollegeDegrees360, licensed CC-BY-SA

Immutability example – fibonacci

$F_0 = 0,\ F_1 = 1$
$F_n = F_{n-1} + F_{n-2}$

What are side effects?

- When your code affects the outside world
- Printing
- Writing to disk
- Network calls

What do they have to do with FP?

- Separate your logic from your side effects
- Push your side effects out in to the edge of your system
- In the middle of your system make representations of the side effects
- Makes the core easier to test and reason about
- No side effects to stub $\rightarrow$ simpler tests
- Push hard testing out to the edge

Example

Referential transparency

- The combination of immutability and no side effects
- When you call a function again you get the same result
- Includes no side effects
- You can substitute the saved result of the function for the call
- Everything a function does is represented by the result
- Nothing unexpected!

Example

Teaching FP

- Don't get too mathematical
- Relate to stuff they already know
- Focus on examples
- Avoid the scary words like monad until they understand it
- Focus on advantages to them
- Pick your battles

Learning more

- Functional Programming Principles in Scala on coursera
- Functional programming in Scala by Paul Chiusano and Rúnar Bjarnason
- Refactoring Ruby with Monads by Tom Stuart
- github.com/cwmyers/FunctionalTraining
- github.com/NICTA/course
- If you're after concepts, go for concept oriented resources
- Try multiple sources before you find one for you
- If it seems really hard, probably the resource, not you.

Questions?

Stop things changing behind our backs