Type hints from method argument names

OR “JUST PLEASE USE STATIC LANGUAGES FOR GOD’S SAKE”

BY ALEKSEI KOSOZHIKHIN
What’s it all about

• Static language method declaration:

  Type Method(SomeOtherType arg)

• Dynamic language method declaration:

  Method: arg

• Wait, what is arg?
Previous work

• A Case Study on Type Hints in Method Argument Names in Pharo Smalltalk Projects
  by Boris Spasojevic, Mircea Lungu, Oscar Nierstrasz

• Exploiting Type Hints in Method Argument Names to Improve Lightweight Type Inference
  by Nevena Milojkovic, Mohammad Ghafari, Oscar Nierstrasz
The existing algorithm

1. Match full name
2. Remove everything before first uppercase letter and match again
3. If it’s “spec”, it’s probably MetacelloAbstractVersionConstructor
4. ...and some more specific cases
My genius plan

• Tokenize!
  aStupidArray = a, Stupid, Array

• Match tails
  a, Stupid, Array ≈ Array

• Get more results
  a, Stupid, Array ≈ CoolArray too

score1 = matched_length ^ 2 / (arg_length * class_length)
Duck typing

• aStringOrSymbol = String + Symbol
• String = String, Symbol = Symbol
Evaluation

- 202 arguments from various methods of classes in Glamour-Core
- 182 produced some matchings
- 54 (29.7%) provided correct best guess 😞
- 90 (49.5%) guessed something right 😊
Common prefixes

• Glamour (Glamour-Core, Glamour-Browsers, ...):
  • GLMTransmission, GLMPort, ...

• 107 (58.8%) guessed something right 😊
Package vicinity

- Are classes from the same package more likely?

- \[ \text{score2} = \text{score1} \times (\text{arg\_package} = \text{class\_package} ? \text{same\_package\_coef} : 1) \]

- 107 (58.8%) guessed something right 😞

- 112 types guessed in total 😊
Class hierarchy

- aBoolean = True + False
- aCollection = Array + OrderedCollection + ...

- Unfold matchings! Superclass + Subclass * subclass_coef + SubclassesSubclass * subclass_coef^2 + ...

- 114 (62.3%) guessed something right 😃
- 135 types guessed in total 😊
Exciting live demo
Caveats

• Performance
  • ≈4s for Glamour-Core

• Stupid argument names
  • anObject
  • No type hints
APPLAUSE
OR KNOCK OR WHATEVER
Summary

• Guessing types from argument names
• Breaking in tokens by camelCase and matching tails
• Removing package prefixes
• Promoting same-package matchings
• Suggesting subclasses of successful matchings