Your Title Here

Your subtitle here (optional)

Christoph Bühler
OST Eastern Switzerland University of Applied Sciences
MSE Seminar “Programming Languages”
Supervisor: Farhad Mehta
Semester: Autumn/Spring 2020

Abstract

Short abstract about why a developer should understand the lambda calculus. What is is how it works and why it is relevant in software engineering and other fields.

Keywords: lambda calculus, type systems

1 Introduction

Motivation why the lambda calculus is needed in programming languages.

Describe the changes from untyped to simply typed calculus and what it actually does to the system (it narrows the "compilable" results)

[Pie02]

2 Lambda Calculus

Very briefly describe the untyped lambda calculus with reference to the paper of Simon Gubler (which writes the paper for the untyped calculus)

The brief description should provide an overview of terms and elements so that the reader can advance to the transition to the typed system.

1. simple syntax example
2. simple rules
   - variables
   - abstractions
   - applications
3. simple typescript example [Pic19]

3 Simply Typed Lambda-Calculus

Short introduction sentence here. the following sub sections should describe the principles of the simply typed lambda calculus.

- What are types
- What can we do with them
- further elements like base types
- where is it used (lambda calculus)

each section should explain the topic in both ways. first the "math" way to explain what the topic is and then, the developer language version.

The goal should be, that a developer can read those sections and can interpret what the simply typed lambda calculus is. Explained in terms and examples that a normal experienced developer should understand.

3.1 Key Differences

Briefly describe the key differences between the untyped and the simply typed calculus.

further sections will specify those differences.

3.2 Types

Describe what types are and what types mean. In terms of mathematics and programming languages.

Why they are needed for certain tasks.

- example types
- describe in math language
- translate to "developer language"
- why do we need them?

3.3 Types in the Calculus

Describe the different types and their context. What are function types and how are they used. What are the properties of typing.

Ref to Curry/Church

typescript example of church notation with typings.

3.4 Extending types

- What hides behind "base types"
- the unit type (void)
- other typing constructs for the simple calculus

3.5 References

- What is a reference type

3.6 Error states

- What exactly is an exception
- How to throw / handle them

4 Usage and Relevancy

- describe the usefulness of the calculus
- describe where it is used (ref to dev ops / kubernetes maybe?)

- why it is relevant (for system F / system T among others)

5 Conclusion

Give the conclusion over the paper. What was shown, why it was shown.

What should the reader know about the calculus now and how the reader can translate that into typed languages

References

[Pic19] Enrico Piccinin. Let's code the roots of functional programming: Lambda calculus implemented in typescript. https://medium.com/hackernoon/lets-code-the-roots-of-functional-programming-lambda-calculus-implemented-in-typescript-36806ebc2857, 2019. Accessed: 2020-09-19.

[Pie02] Benjamin C. Pierce. *Types and Programming Languages*. The MIT Press, Cambridge, MA, USA, 2002.