A Featherweight Java Interpreter Implemented in a Functional Programming Language

Despite the fact that the functional programming languages are very old, they are still getting more familiar even to non-academic developers. This thesis is trying to familiarize the reader with OCaml language and the creation of interpreters for functional languages in general. On the example of simple language, which can be considered as a subset of OCaml, several interpreting techniques are demonstrated. It also includes the topics about type checking and type inference.

In the second part of this thesis a minimal core calculus for Java, that is known as Featherweight Java, is introduced. From the experience gained in the first part of this thesis, the interpreter for this language is created. Finally, small extension of Featherweight Java and new interpreter, which is now based on another approach, are developed.