University of Pisa - Scuola Superiore Sant'Anna
Department of Computer Science

An Imperative-Functional Programming Language

Technical Report

Module: Advanced Programming (301AA)

Lecturers: Gianluigi Ferrari Antonio Cisternino

MSc Computer Science and Networking
Academic Year 2014/2015

Table of Contents

Abstract

The aim of this report is to offer an overview of the `funW@P` implementation and of the main design choices made during its development. Also, this document is intended both as a reference and a manual for the user wishing to use or extend our simple language.

All the information provided here will also be available at the official page of the project on GitHub (https://github.com/MCSN-project2014/APproject).

Chapter 1: **Introduction**

The report structure follows closely the project's one and focuses on the most important developing phases. After reporting the complete description of the `funW@P` grammar and a brief summary about the tools used for generating the tokenizer, the parser and the type-checker (see Chapter 2), a recapitulation about the chosen intermediate representation (see Chapter 3) is offered. Then, the `F#` code generator (see Chapter 4) and the language interpreter (see Chapter 5) are reviewed. To conclude, Chapter 6 and **??** are an overview of the testing phase and a quick user guide respectively.

Chapter 2: **Designing the Language**

The grammar design has been a complex, delicate and continuous task of the project, since the grammar represents the true definition of a language in formal terms and it revealed to be of crucial importance for all the subsequent phases. Lots of on the fly adjustments and corrections leaded to the definition reported in the following section.

2.1 EBNF Grammar

The whole grammar is described by means of the Extended Backus-Naur Form (EBNF, [2]) which is metasyntax, commonly adopted by parser generators nowadays. We first introduce the used tokens, which are the following ones:

TOKENS
```
ident  = letter {letter | digit}.
url = ap "http://" {UrlInLine} ap.
number = digit {digit}.
string = '"' {AnyButDoubleQuote | "\\\""} '"'.
```

Then we provide all the needed productions:

2.2 Submission of Project Report

Chapter 3: **The Abstract Syntax Tree (AST)**

Chapter 4: **The F# Code Generator**

Chapter 5: **The** `funW@P` **Interpreter**

Chapter 6: **The System Testing**

Bibliography

[1] Christian Dawson. *The Essence of Computing Projects – A Student's Guide*. 192 pages. ISBN: 013021972X. Pearson Education, 2000.

[2] *EBNF on Wikipedia*. http:en.wikipedia.orgwikiExtended_Backus%E2%80%93Naur_Form