

The Whiley Language Specification

David J. Pearce
School of Engineering and Computer Science
Victoria University of Wellington, New Zealand
djp@ecs.vuw.ac.nz

December 31, 2013

Contents

1	Introduction	2
1.1	Overview	2
1.2	Goals	2
1.3	History	2
2	Lexical Structure	3
2.1	Indentation	3
2.2	Blocks	3
2.3	Whitespace	3
2.4	Identifiers	3
3	Compilation Units	4
3.1	Packages	4
3.2	Imports	4
3.3	Type Declarations	4
3.4	Constant Declarations	4
3.5	Function & Method Declarations	4
3.6	Visibility Modifiers	4
4	Types	5
4.1	Primitives	5
4.2	Collection Types	5
4.3	Union Types	5
4.4	Intersection Types	5
4.5	Negation Types	5
4.6	Reference Types	5
5	Expressions	6
5.1	Binary Expressions	6
6	Statements	8
6.1	Variable Declarations	8
6.2	Assign Statements	8
6.3	Return Statements	8
6.4	If/Else Statements	8
6.5	While Statements	8
6.6	Do/While Statements	8
6.7	For Statements	8
6.8	Switch Statements	8
6.9	Try/Catch Statements	8

Chapter 1

Introduction

1.1 Overview

1.2 Goals

1.3 History

Chapter 2

Lexical Structure

2.1 Indentation

2.2 Blocks

2.3 Whitespace

2.4 Identifiers

Chapter 3

Compilation Units

3.1 Packages

3.2 Imports

3.3 Type Declarations

3.4 Constant Declarations

3.5 Function & Method Declarations

3.6 Visibility Modifiers

Chapter 4

Types

4.1 Primitives

4.2 Collection Types

4.3 Union Types

4.4 Intersection Types

4.5 Negation Types

4.6 Reference Types

Expr	::=	Cond [(&&) Expr]	// Expressions
Cond	::=	Append [Cop Expr]	// Condition Expressions
Append	::=	Range [++ Expr]	// Append Expressions
Range	::=	AddSub [.. Expr]	// Range Expressions
AddSub	::=	MulDiv [(+ -) Expr]	// Additive Expressions
MulDiv	::=	Index [(* / %) Expr]	// Multiplicative Expressions
Index	::=	???	// Index Expressions

Figure 5.1: Syntax for Binary Expressions

Chapter 5

Expressions

5.1 Binary Expressions

Term	::=	<i>// Terms</i>	
	<i>Constant</i>		<i>// Constant expressions</i>
	<i>Identifier</i>		<i>// Identifier expressions</i>
	$Expr_1 (, Expr_i)^+$		<i>// Tuple expressions</i>
	$(Expr)$		<i>// Bracketed expressions</i>
	$ Expr $		<i>// Size expressions</i>
	$Identifier ([Expr_1 (, Expr_i)^+])$		<i>// Invocation expressions</i>
	$(- ! \sim \& *) Expr$		<i>// Unary expressions</i>
	$new Expr$		<i>// Allocation expressions</i>
	$\{ [Expr_1 (, Expr_i)^*] \}$		<i>// Set expressions</i>
	$\{ [Expr_1 \Rightarrow Expr'_1 (, Expr_i \Rightarrow Expr'_i)^*] \}$		<i>// Map expressions</i>
	$[[Expr_1 (, Expr_i)^*]]$		<i>// List expressions</i>
	$\{ [n_1 : Expr_1 (, n_i : Expr_i)^*] \}$		<i>// Record expressions</i>

Figure 5.2: Syntax for Term Expressions

Constant	::=	<i>// Constants</i>	
	$(0 1)^+ b$		<i>// Boolean constants</i>
	$(0-9)^+$		<i>// Integer constants</i>
	$(0-9)^+ . (0-9)^+$		<i>// Decimal constants</i>
	$null$		<i>// Null constant</i>

Figure 5.3: Syntax for Constant Expressions

Identifier	::=	$(- a-z A-Z) (- a-z A-Z 0-9)^*$	<i>// Identifiers</i>
-------------------	------------	-----------------------------------------------	-----------------------

Figure 5.4: Syntax for Identifiers

Chapter 6

Statements

6.1 Variable Declarations

6.2 Assign Statements

6.3 Return Statements

6.4 If/Else Statements

6.5 While Statements

6.6 Do/While Statements

6.7 For Statements

6.8 Switch Statements

6.9 Try/Catch Statements