

# Contents

<b>1. Introduction</b> .....	1
1.1 The goals of the book .....	2
1.2 The contents of the book .....	3
1.3 Decomposing Java and the JVM .....	7
1.4 Sources and literature .....	11
<b>2. Abstract State Machines</b> .....	15
2.1 ASMs in a nutshell .....	15
2.2 Mathematical definition of ASMs .....	18
2.3 Notational conventions .....	27

---

## Part I. Java

---

<b>3. The imperative core <math>\text{Java}_{\mathcal{I}}</math> of Java</b> .....	33
3.1 Static semantics of $\text{Java}_{\mathcal{I}}$ .....	33
3.2 Transition rules for $\text{Java}_{\mathcal{I}}$ .....	39
<b>4. The procedural extension <math>\text{Java}_{\mathcal{C}}</math> of <math>\text{Java}_{\mathcal{I}}</math></b> .....	47
4.1 Static semantics of $\text{Java}_{\mathcal{C}}$ .....	47
4.2 Transition rules for $\text{Java}_{\mathcal{C}}$ .....	63
<b>5. The object-oriented extension <math>\text{Java}_{\mathcal{O}}</math> of <math>\text{Java}_{\mathcal{C}}</math></b> .....	71
5.1 Static semantics of $\text{Java}_{\mathcal{O}}$ .....	71
5.2 Transition rules for $\text{Java}_{\mathcal{O}}$ .....	80
<b>6. The exception-handling extension <math>\text{Java}_{\mathcal{E}}</math> of <math>\text{Java}_{\mathcal{O}}</math></b> .....	87
6.1 Static semantics of $\text{Java}_{\mathcal{E}}$ .....	87
6.2 Transition rules for $\text{Java}_{\mathcal{E}}$ .....	89
<b>7. The concurrent extension <math>\text{Java}_{\mathcal{T}}</math> of <math>\text{Java}_{\mathcal{E}}</math></b> .....	95
7.1 Static semantics of $\text{Java}_{\mathcal{T}}$ .....	96
7.2 Transition rules for $\text{Java}_{\mathcal{T}}$ .....	98
7.3 Thread invariants .....	106

<b>8. Java is type safe</b> .....	111
8.1 Structural properties of Java runs .....	111
8.2 Unreachable statements .....	117
8.3 Rules of definite assignment .....	121
8.4 Java is type safe .....	126

---

## Part II. Compilation of Java: The Trustful JVM

---

<b>9. The JVM<sub><math>\mathcal{I}</math></sub> submachine</b> .....	139
9.1 Dynamic semantics of the JVM <sub><math>\mathcal{I}</math></sub> .....	139
9.2 Compilation of Java <sub><math>\mathcal{I}</math></sub> .....	142
<b>10. The procedural extension JVM<sub><math>\mathcal{C}</math></sub> of JVM<sub><math>\mathcal{I}</math></sub></b> .....	147
10.1 Dynamic semantics of the JVM <sub><math>\mathcal{C}</math></sub> .....	147
10.2 Compilation of Java <sub><math>\mathcal{C}</math></sub> .....	153
<b>11. The object-oriented extension JVM<sub><math>\mathcal{O}</math></sub> of JVM<sub><math>\mathcal{C}</math></sub></b> .....	155
11.1 Dynamic semantics of the JVM <sub><math>\mathcal{O}</math></sub> .....	155
11.2 Compilation of Java <sub><math>\mathcal{O}</math></sub> .....	157
<b>12. The exception-handling extension JVM<sub><math>\mathcal{E}</math></sub> of JVM<sub><math>\mathcal{O}</math></sub></b> .....	159
12.1 Dynamic semantics of the JVM <sub><math>\mathcal{E}</math></sub> .....	159
12.2 Compilation of Java <sub><math>\mathcal{E}</math></sub> .....	163
<b>13. Executing the JVM<sub><math>\mathcal{N}</math></sub></b> .....	165
<b>14. Correctness of the compiler</b> .....	167
14.1 The correctness statement .....	167
14.2 The correctness proof .....	178

---

## Part III. Bytecode Verification: The Secure JVM

---

<b>15. The defensive virtual machine</b> .....	209
15.1 Construction of the defensive JVM .....	210
15.2 Checking JVM <sub><math>\mathcal{I}</math></sub> .....	210
15.3 Checking JVM <sub><math>\mathcal{C}</math></sub> .....	213
15.4 Checking JVM <sub><math>\mathcal{O}</math></sub> .....	214
15.5 Checking JVM <sub><math>\mathcal{E}</math></sub> .....	219
15.6 Checking JVM <sub><math>\mathcal{N}</math></sub> .....	221
15.7 Checks are monotonic .....	222

<b>16. Bytecode type assignments</b> .....	223
16.1 Problems of bytecode verification .....	224
16.2 Successors of bytecode instructions .....	231
16.3 Type assignments without subroutine call stacks .....	236
16.4 Soundness of bytecode type assignments .....	242
16.5 Certifying compilation .....	252
<b>17. The diligent virtual machine</b> .....	273
17.1 Principal bytecode type assignments .....	273
17.2 Verifying $JVM_{\mathcal{I}}$ .....	275
17.3 Verifying $JVM_{\mathcal{C}}$ .....	279
17.4 Verifying $JVM_{\mathcal{O}}$ .....	283
17.5 Verifying $JVM_{\mathcal{E}}$ .....	283
17.6 Verifying $JVM_{\mathcal{N}}$ .....	286
<b>18. The dynamic virtual machine</b> .....	289
18.1 Initiating and defining loaders .....	289
18.2 Loading classes .....	290
18.3 Dynamic semantics of the $JVM_{\mathcal{D}}$ .....	291

---

## Appendix

---

<b>A. Executable Models</b> .....	305
A.1 Overview .....	305
A.2 Java .....	306
A.3 Compiler .....	312
A.4 Java Virtual Machine .....	314
<b>B. Java</b> .....	323
B.1 Rules .....	323
B.2 Arrays .....	331
<b>C. JVM</b> .....	335
C.1 Trustful execution .....	335
C.2 Defensive execution .....	343
C.3 Diligent execution .....	344
C.4 Check functions .....	347
C.5 Successor functions .....	348
C.6 Constraints .....	349
C.7 Arrays .....	351
C.8 Abstract versus real instructions .....	355

<b>D. Compiler</b> .....	361
D.1 Compilation functions.....	361
D.2 maxOpd .....	363
D.3 Arrays .....	364
<b>References</b> .....	365
<b>List of Figures</b> .....	367
<b>List of Tables</b> .....	371
<b>Index</b> .....	373